Deep Learning for IoT Big Data and Streaming Analytics: A Survey

Mehdi Mohammadi, *Graduate Student Member, IEEE*, Ala Al-Fuqaha, *Senior Member, IEEE*, Sameh Sorour, *Senior Member, IEEE*, Mohsen Guizani, *Fellow, IEEE*

***Abstract*—In the era of the Internet of Things (IoT), an enormous amount of sensing devices collect and/or generate various sensory data over time for a wide range of fields and applications. Based on the nature of the application, these devices will result in big or fast/real-time data streams. Applying analytics over such data streams to discover new information, predict future insights, and make control decisions is a crucial process that makes IoT a worthy paradigm for businesses and a quality-of-life improving technology. In this paper, we provide a thorough overview on using a class of advanced machine learning techniques, namely deep learning (DL), to facilitate the analytics and learning in the IoT domain. We start by articulating IoT data characteristics and identifying two major treatments for IoT data from a machine learning perspective, namely IoT big data analytics and IoT streaming data analytics. We also discuss why DL is a promising approach to achieve the desired analytics in these types of data and applications. The potential of using emerging DL techniques for IoT data analytics are then discussed, and its promises and challenges are introduced. We present a comprehensive background on different DL architectures and algorithms. We also analyze and summarize major reported research attempts that leveraged DL in the IoT domain. The smart IoT devices that have incorporated DL in their intelligence background are also discussed. DL implementation approaches on the fog and cloud centers in support of IoT applications are also surveyed. Finally, we shed light on some challenges and potential directions for future research. At the end of each section, we highlight the lessons learned based on our experiments and review of the recent literature.**

***Index Terms*—Deep learning, deep neural network, Internet of Things, on-device intelligence, IoT big data, fast data analytics, cloud-based analytics.**

I. Introduction

THE VISION of the Internet of Things (IoT) is to transform traditional objects to being smart by exploiting a wide range of advanced technologies, from embedded devices and communication technologies to Internet protocols, data analytics, and so forth [1]. The potential economic impact of IoT is expected to bring many business opportunities and to accelerate the economic growth of IoT-based services. Based on McKinsey's report on the global economic impact of IoT [2], the annual economic impact of IoT in 2025 would be in the range of $2.7 to $6.2 trillion. Healthcare constitutes the major part, about 41% of this market, followed by industry and energy with 33% and 7% of the IoT market, respectively. Other domains such as transportation, agriculture, urban infrastructure, security, and retail have about 15% of the IoT market totally. These expectations imply the tremendous and steep growth of the IoT services, their generated data and consequently their related market in the years ahead.

Indeed, machine learning (ML) will have effects on jobs and the workforce, since parts of many jobs may be "suitable for ML applications" [3]. This will lead to increase in demand for some ML products and the derived demand for the tasks, platforms, and experts needed to produce such products. The economic impact of machine learning in McKinsey's report [2] is defined under knowledge work automation; "the use of computers to perform tasks that rely on complex analyses, subtle judgments, and creative problem solving". The report mentions that advances in ML techniques, such as deep learning and neural networks, are the main enablers of knowledge work automation. Natural user interfaces, such as speech and gesture recognition are other enablers that are highly benefiting from ML technologies. The estimated potential economic impact of knowledge work automation could reach $5.2 trillion to $6.7 trillion per year by 2025. Figure shows the break down of this estimate in different occupations. Compared to the economic impact of IoT, this estimation asserts the more attention toward the extraction of value out of data and the potential impacts of ML on the economic situation of individuals and societies. These economic impacts have serious consequences on individuals and countries, since people need to adapt to new means of earning income suitable for them to maintain their desired living standard.

In recent years, many IoT applications arose in different vertical domains, i.e., health, transportation, smart home, smart city, agriculture, education, etc. The main element of most of these applications is an intelligent learning mechanism for prediction (i.e., regression, classification, and clustering), data mining and pattern recognition or data analytics in general. Among the many machine learning approaches, Deep Learning (DL) has been actively utilized in many IoT applications in recent years. These two technologies (i.e., DL and IoT) are among the top three strategic technology trends for 2017 that were announced at Gartner Symposium/ITxpo 2016 [4]. The

Manuscript received September 19, 2017; revised March 30, 2018; accepted May 23, 2018. Date of publication June 6, 2018; date of current version November 19, 2018. *(Corresponding author: Mohsen Guizani.)*

M. Mohammadi and A. Al-Fuqaha are with the Department of Computer Science, Western Michigan University, Kalamazoo, MI 49008 USA (e-mail: mehdi.mohammadi@wmich.edu; ala.al-fuqaha@wmich.edu).

S. Sorour and M. Guizani are with the Department of Electrical and Computer Engineering, University of Idaho, Moscow, ID 83844 USA (e-mail: samehsorour@uidaho.edu; mguizani@ieee.org).

Digital Object Identifier 10.1109/COMST.2018.2844341

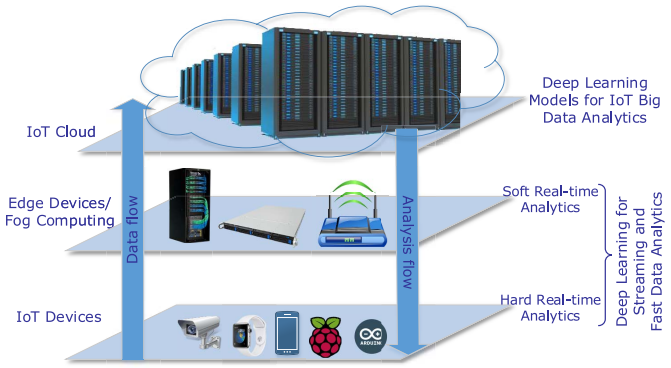


Fig. 1. IoT data generation at different levels and deep learning models to address their knowledge abstraction.

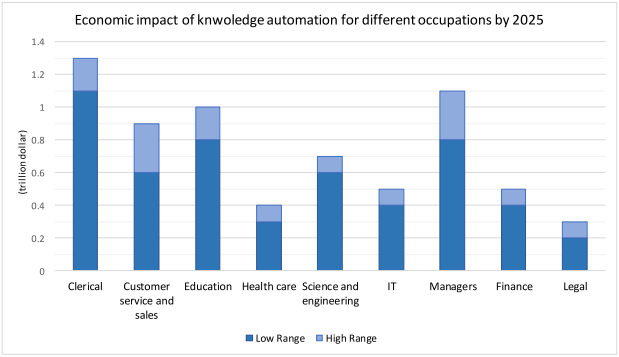


Fig. 2. The break down of estimated economic impact of $5.2 trillion to $6.7 trillion per year for machine learning in 2025.

cause of this intensive publicity for DL refers to the fact that traditional machine learning approaches do not address the emerging analytic needs of IoT systems. Instead, IoT systems need different modern data analytic approaches and artificial intelligence (AI) methods according to the hierarchy of IoT data generation and management as illustrated in Figure 1.

The growing interest in the Internet of Things (IoT) and its derivative big data need stakeholders to clearly understand their definition, building blocks, potentials and challenges. IoT and big data have a two way relationship. On one hand, IoT is a main producer of big data, and on the other hand, it is an important target for big data analytics to improve the processes and services of IoT [5]. Moreover, IoT big data analytics have proven to bring value to the society. For example, it is reported that, by detecting damaged pipes and fixing them, the Department of Park Management in Miami has saved about one million USD on their water bills [6].

IoT data are different than the general big data. To better understand the requirements for IoT data analytics, we need to explore the properties of IoT data and how they are different from those of general big data. IoT data exhibits the following characteristics [6]:

- *Large-Scale Streaming Data:* A myriad of data capturing devices are distributed and deployed for IoT applications, and generate streams of data continuously. This leads to a huge volume of continuous data.
- *Heterogeneity:* Various IoT data acquisition devices gather different information resulting in data heterogeneity.
- *Time and space correlation:* In most of IoT applications, sensor devices are attached to a specific location, and thus have a location and time-stamp for each of the data items.
- *High noise data:* Due to tiny pieces of data in IoT applications, many of such data may be subject to errors and noise during acquisition and transmission.

Although obtaining hidden knowledge and information out of big data is promising to enhance the quality of our lives, it is not an easy and straightforward task. For such a complex and challenging task that goes beyond the capabilities of the traditional inference and learning approaches, new technologies, algorithms, and infrastructures are needed [7]. Luckily, the recent progresses in both fast computing and advanced machine learning techniques are opening the doors for big data analytics and knowledge extraction that is suitable for IoT applications.

Beyond the big data analytics, IoT data calls for another new class of analytics, namely fast and streaming data analytics, to support applications with high-speed data streams and requiring time-sensitive (i.e., real-time or near real-time) actions. Indeed, applications such as autonomous driving, fire prediction, driver/elderly posture (and thus consciousness and/or health condition) recognition demands for fast processing of incoming data and quick actions to achieve their target. Several researchers have proposed approaches and frameworks for fast streaming data analytics that leverage the capabilities of cloud infrastructures and services [8], [9]. However, for the aforementioned IoT applications among others, we need fast analytics in smaller scale platforms (i.e., at the system edge) or even on the IoT devices themselves. For example, autonomous cars need to make fast decisions on driving actions such as lane or speed change. Indeed, this kind of decisions should be supported by fast analytics of possibly multi-modal data streaming from several sources, including the multiple vehicle sensors (e.g., cameras, radars, LIDARs, speedometer, left/right signals, etc.), communications from other vehicles, and traffic entities (e.g., traffic light, traffic signs). In this case, transferring data to a cloud server for analysis and returning back the response is subject to latency that could cause traffic violations or accidents. A more critical scenario would be detecting pedestrians by such vehicles. Accurate recognition should be performed in strict real-time to prevent fatal accidents. These scenarios imply that fast data analytics for IoT have to be close to or at the source of data to remove unnecessary and prohibitive communication delays.

A. Survey Scope

DL models in general bring two important improvements over the traditional machine learning approaches in the two phases of training and prediction. First, they reduce the need for hand crafted and engineered feature sets to be used for the training [10]. Consequently, some features that might not

be apparent to a human view can be extracted easily by DL models. In addition, DL models improve the accuracy.[1]

In this paper, we review a wide range of deep neural network (DNN) architectures and explore the IoT applications that have benefited from DL algorithms. The paper identifies five main foundational IoT services that can be used in different vertical domains beyond the specific services in each domain. It will also discuss the characteristics of IoT applications and the guide to matching them with the most appropriate DL model. This survey focuses on the confluence of two emerging technologies, one in communication networks, i.e., IoT and the other in artificial intelligence, i.e., DL, detailing their potential applications and open issues. The survey does not cover traditional machine learning algorithms for IoT data analytics as there are some other attempts, mentioned in Section I-B, that have covered such approaches. Moreover, this survey also does not go into the details of the IoT infrastructure from a communications and networking perspective.

B. Related Work

To the best of our knowledge, there does not exist an article in the literature that is dedicated to surveying the specific relation between IoT data and DL as well as applications of DL methods in IoT. There are few works presenting common data mining and machine learning methods that have been used in IoT environments. The work presented in [11] by Tsai *et al.* focused on data mining approaches in IoT. It addressed different classification, clustering, and frequent pattern mining algorithms for the IoT infrastructure and services. However, that work did not consider DL approaches, which is the focus of our survey. Moreover, their focus is mainly on offline data mining, while we also consider learning and mining for both real-time (i.e., fast) and big data analytics.

Perera *et al.* [12] have reviewed different classes of machine learning approaches (supervised and unsupervised, rules, fuzzy logic, etc.) in the reasoning phase of a context-aware computing system, and have discussed the potentials of applying those methods in IoT systems. Nonetheless, they also did not study the role of DL on the context reasoning.

The work in [13] by Alsheikh *et al.* provides a survey of machine learning methods for wireless sensor networks (WSNs). In that work, the authors studied machine learning methods in the functional aspects of WSNs, such as routing, localization, and clustering, as well as non-functional requirements, such as security and quality of service. They reviewed several algorithms in supervised, unsupervised, and reinforcement learning approaches. This work focuses on the infrastructure of WSN (which is one potential infrastructure for implementing IoT applications), while our work is not dependent on the sources of data (i.e., IoT infrastructures) and covers a wide range of IoT applications and services. Moreover, the focus of [13] was on traditional machine learning methods, whereas this article focuses on advanced and DL techniques.

Finally, Fadlullah *et al.* [14] addressed DL approaches in network traffic control systems. While this work primarily focuses on the infrastructure of network, it differs from our work that focuses on the usage of DL in IoT applications.

Beyond the specific works on the IoT, Qiu *et al.* [15] reviewed several traditional machine learning techniques along with several advanced techniques including DL for processing general big data. In specific, they highlighted the connection of different machine learning techniques with signal processing technologies to process and analyze timely big data applications.

C. Contributions

This paper is intended for IoT researchers and developers who want to build analytics, AI systems, and learning solutions on top of their IoT infrastructure, using the emerging DL machine learning approaches. The contributions of this paper can be summarized as follows:

- In order to adopt DL approaches in the IoT ecosystems, we identify the key characteristics and issues of IoT data.
- Compared to some related work in the literature that have addressed machine learning for IoT, we review the state-of-the-art DL methods and their applicability in the IoT domain both for big data and streaming data analytics.
- We review a wide range of IoT applications that have used DL in their context. We also provide a comparison and a guideline for using different types of DNN in the various IoT domains and applications.
- We review the recent approaches and technologies for deploying DL on all levels of IoT hierarchy from resource constrained devices to the fog and the cloud.
- We highlight the challenges and future research directions for the successful and fruitful merging of DL and IoT applications.

The rest of this paper is organized as follows. In Section II, we highlight the IoT data characteristics and describe what IoT big data as well as fast and streaming data are, and how they are different from the general big data. Section III presents several common and successful architectures of DNNs. It also includes a brief description of advancements toward real-time and fast DL architectures as well as state-of-the-art algorithms that are joint with DL. A succinct review of several frameworks and tools with different capabilities and algorithms that support DNNs is also presented. IoT applications in different domains (e.g., healthcare, agriculture, ITS, etc.) that have used DL will be surveyed in Section IV. Section V reviews the attempts to bring DNN to the resource constraint devices. Section VI explains the works that investigated bringing the DNN models to the scale of fog and cloud computing. Future research direction and open challenges are presented

[1]Accuracy in this work in general refers to the degree to which the result of the prediction conforms to the ground truth values. Readers may also face top-2 or top-3 accuracy in the text. In general, top-N accuracies refers to considering the N highest-probability answers of the prediction model and checking whether that set contains the expected value or not. Therefore, top-1 accuracy refers to the output with the highest probability. Likewise, top-3 accuracy refers to the three most probable predictions. For example, if we feed a picture of a tiger to a model that recognizes animal images, and it returns the list of possible outputs as dog:0.72, tiger:0.69, and cat:0.58, the top-1 accuracy will output the answer set containing only "dog", which is counted as wrong. On the other hand, the top-2 and top-3 accuracies will result in output sets containing "tiger" as an answer, and are thus counted as correct.

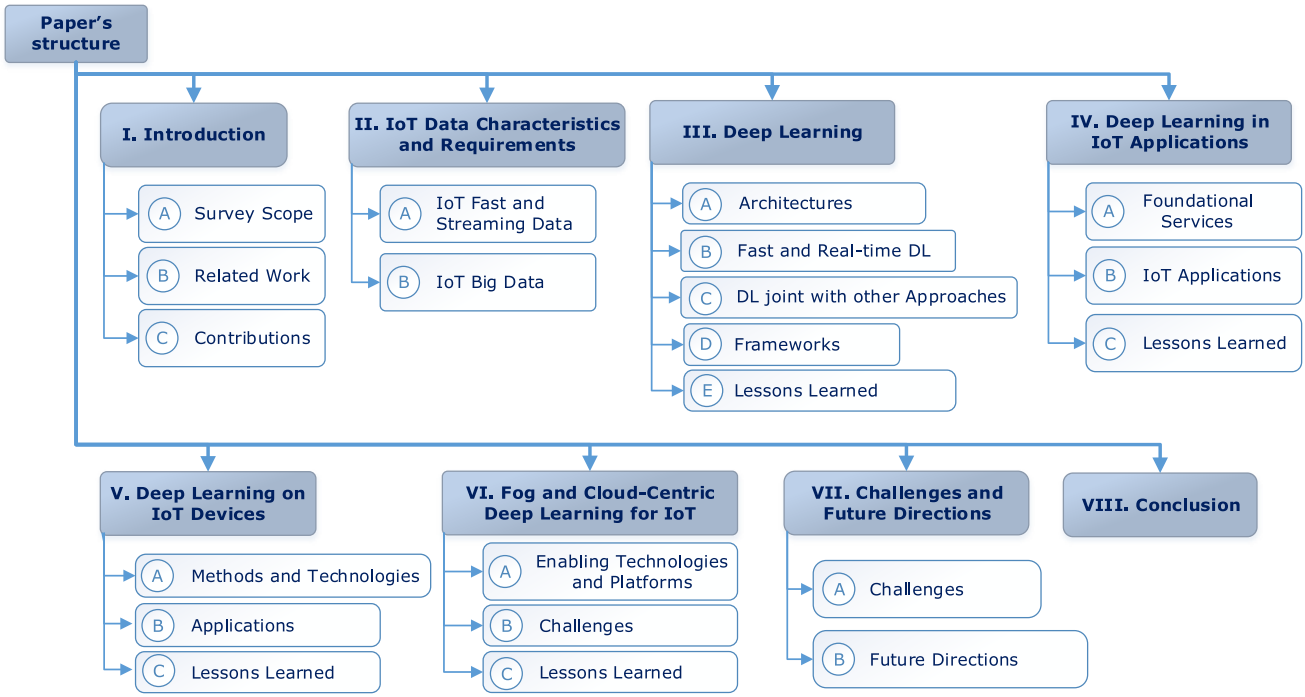


Fig. 3. Structure of the survey.

in Section VII. The paper is concluded in Section VIII with a summary of its main take-away messages. Figure 3 depicts the structure of the paper.

II. IoT Data Characteristics and Requirements for Analytics

IoT data can be streamed continuously or accumulated as a source of big data. Streaming data refers to the data generated or captured within tiny intervals of time and need to be promptly analyzed to extract immediate insights and/or make fast decisions. Big data refers to huge datasets that the commonly used hardware and software platforms are not able to store, manage, process, and analyze. These two approaches should be treated differently since their requirements for analytic response are not the same. Insight from big data analytics can be delivered after several days of data generation, but insight from streaming data analytics should be ready in a range of few hundreds of milliseconds to few seconds.

Data fusion and sharing play a critical role in developing ubiquitous environments based on IoT data. This role is more critical for time-sensitive IoT applications where a timely fusion of data is needed to bring all pieces of data together for analysis and consequently providing reliable and accurate actionable insights. Alam *et al.* [16] presented a survey paper in which data fusion techniques for IoT environments are reviewed followed by several opportunities and challenges.

A. IoT Fast and Streaming Data

Many research attempts suggested streaming data analytics that can be mainly deployed on high-performance computing systems or cloud platforms. The streaming data analytics on such frameworks is based on data parallelism and incremental processing [17]. By data parallelism, a large dataset is partitioned into several smaller datasets, on which parallel analytics are performed simultaneously. Incremental processing refers to fetching a small batch of data to be processed quickly in a pipeline of computation tasks. Although these techniques reduce time latency to return a response from the streaming data analytic framework, they are not the best possible solution for time-stringent IoT applications. By bringing streaming data analytics closer to the source of data (i.e., IoT devices or edge devices) the need for data parallelism and incremental processing is less sensible as the size of the data in the source allows it to be processed rapidly. However, bringing fast analytics on IoT devices introduces its own challenges such as limitation of computing, storage, and power resources at the source of data.

B. IoT Big Data

IoT is well-known to be one of the major sources of big data, as it is based on connecting a huge number of smart devices to the Internet to report their frequently captured status of their environments. Recognizing and extracting meaningful patterns from enormous raw input data is the core utility of big data analytics as it results in higher levels of insights for decision-making and trend prediction. Therefore, extracting these insights and knowledge from the big data is of extreme importance to many businesses, since it enables them to gain competitive advantages. In social sciences, Hilbert [18] compares the impact of big data analytics to that of the invention of the telescope and microscope for astronomy and biology, respectively.

Several works have described the general features of big data from different aspects [18]–[21] in terms of volume,

velocity, and variety. However, we adopt the general definition of big data to characterize the IoT big data through the following "6V's" features:

- *Volume:* Data volume is a determining factor to consider a dataset as big data or traditional massive/ very large data. The quantity of generated data using IoT devices is much more than before and clearly fits this feature.
- *Velocity:* The rate of IoT big data production and processing is high enough to support the availability of big data in real-time. This justifies the needs for advanced tools and technologies for analytics to efficiently operate given this high rate of data production.
- *Variety:* Generally, big data comes in different forms and types. It may consist of structured, semi-structured, and unstructured data. A wide variety of data types may be produced by IoT such as text, audio, video, sensory data and so on.
- *Veracity:* Veracity refers to the quality, consistency, and trustworthiness of the data, which in turn leads to accurate analytics. This property needs special attention to hold for IoT applications, especially those with crowd-sensing data.
- *Variability:* This property refers to the different rates of data flow. Depending on the nature of IoT applications, different data generating components may have inconsistent data flows. Moreover, it is possible for a data source to have different rates of data load based on specific times. For example, a parking service application that utilizes IoT sensors may have a peak data load in rush hours.
- *Value:* Value is the transformation of big data to useful information and insights that bring competitive advantage to organizations. A data value highly depends on both the underlying processes/services and the way that data is treated. For example, a certain application (e.g., medical vital sign monitoring) may need to capture all sensor data, while a weather forecast service may need just random samples of data from its sensors. As another example, a credit card provider may need to keep data for a specific period of time and discard them thereafter.

Beyond the aforementioned properties, researchers [18], [20] have identified other characteristics such as:

- Big data can be a byproduct or footprint of a digital activity or IoT interplay. The use of Google's most common search terms to predict seasonal flu is a good example of such digital byproduct [22].
- Big data systems should be horizontally scalable, that is, big data sources should be able to be expanded to multiple datasets. This attribute also leads to the complexity attribute of big data, which in turn imposes other challenges like transferring and cleansing data.

Performing analytics over continuous data flows are typically referred to as stream processing or sometimes complex event processing (CEP) in the literature. Strohbach *et al.* [23] proposed a big data analytics framework for IoT to support the volume and velocity attributes of IoT data analytics. The integration of IoT big data and streaming data analytics, an open issue that needs more investigation, has been also studied as part of that work. However, their proposed framework is designed to be deployed on cloud infrastructures. Moreover, their focus is on the data management aspect of the framework and did not use advanced machine learning models such as DL. Other off-the-shelf products such as Apache Storm are also available for real-time analytics on the cloud. A big gap in this area is the lack of frameworks and algorithms that can be deployed on the fog (i.e., system edge) or even on the IoT devices. When DL comes to play in such cases, a trade-off between the depth and performance of the DNN should be considered.

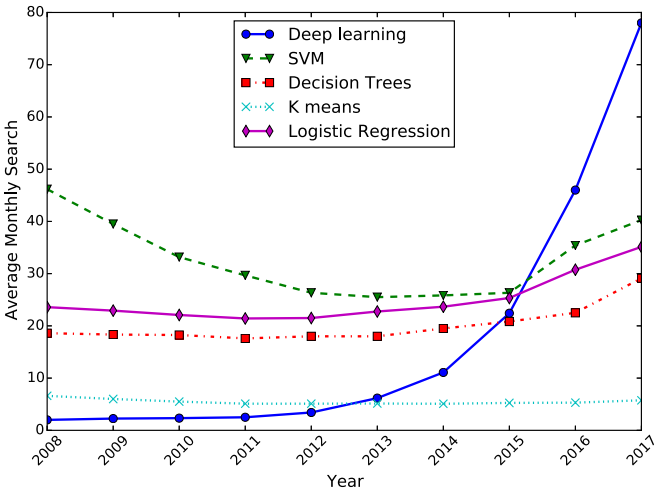


Fig. 4. Google Trend showing more attention toward deep learning in recent years.

III. Deep Learning

DL consists of supervised or unsupervised learning techniques based on many layers of Artificial Neural Networks (ANNs) that are able to learn hierarchical representations in deep architectures. DL architectures consist of multiple processing layers. Each layer is able to produce non-linear responses based on the data from its input layer. The functionality of DL is imitated from the mechanisms of human brain and neurons for processing of signals.

DL architectures have gained more attention in recent years compared to the other traditional machine learning approaches. Such approaches are considered as being shallow-structured learning architectures versions (i.e., a limited subset) of DL. Figure 4 shows the searching trend of five popular machine learning algorithms in Google trends, in which DL is becoming more popular among the others. Although ANNs have been introduced in the past decades, the growing trend for DNNs started in 2006 when Hinton and Salakhutdinov presented the concept of deep belief networks [24]. Thereafter, the state-of-the-art performance of this technology has been observed in different fields of AI including image recognition, image retrieval, search engines and information retrieval, and natural language processing.

DL techniques have been developed on top of traditional ANNs. Feed-forward Neural Networks (FNNs) [25] (a.k.a Multilayer Perceptrons - MLPs) have been used in the past decades to train systems, but when the number of layers is increased, they become difficult to train [26]. The small size of training data was another factor that results in overfitted models. Moreover, the limitation in computational capabilities in those days prohibited the implementation of efficient deeper FNNs. These computational limitations have been resolved lately due to hardware advances in general and the development of Graphics Processing Units (GPUs) and hardware accelerators specifically. Beyond the structural aspects and significance of depth of DL architectures, as well as hardware advances, DL techniques have benefited from advancements in effective training algorithms of deep networks including:

- Using Rectified Linear Units (ReLUs) as activation function [27],
- Introducing dropout methods [28],
- Random initialization for the weights of the network [29],
- Addressing the degradation of training accuracy by residual learning networks [30],
- Solving vanishing gradient problem as well as exploding gradient problem by introducing and enhancing Long Short-Term Memory networks [31], [32].

One advantage of DL architectures, compared to the traditional ANNs, is that DL techniques can learn hidden features from the raw data [10]. Each layer trains on a set of features based on the previous layer's outputs. The inner-most layers can recognize more complex features, since they aggregate and recombine features from the previous layers. This is called the hierarchy of features. For example, in case of a face recognition model, raw image data of portraits as vector of pixels are fed to a model in its input layer. Each hidden layer can then learn more abstract features from the previous layer's outputs, e.g., the first hidden layer identifies the lines and edges, the second layer identifies face parts such as nose, eyes, etc., and the third layer combines all the previous features to generate a face.

However, the reported improvements of DL models are based on empirical evaluations, and there is still no concrete analytical foundation to answer why DL techniques outperform their shallow counterparts. Moreover, there is no clear boundary between deep and shallow networks based on the number of hidden layers. Generally, neural networks with two or more hidden layers that incorporate the recent advanced training algorithms are considered as deep models. Also, recurrent neural networks with one hidden layer are considered as deep since they have a cycle on the units of the hidden layer, which can be unrolled to an equivalent deep network.

A. Architectures

In this section, we present a brief overview of several common DL models as well as the most cutting-edge architectures that have been introduced in recent years. Interested readers can refer to other literature that surveyed the models and architectures of DL in more details, such as [33]. Table I summarizes these models, their attributes, characteristics, and some sample applications.

A DNN consists of an input layer, several hidden layers, and an output layer. Each layer includes several units called neurons. A neuron receives several inputs, performs a weighted summation over its inputs, then the resulting sum goes through an activation function to produce an output. Each neuron has a vector of weights associated to its input size as well as a bias that should be optimized during the training process. Figure 5 depicts the structure of a neuron.

In the training process, the input layer assigns (usually randomly) weights to the input training data and passes it to the next layer. Each subsequent layer also assigns weights to their input and produces their output, which serves as the input for the following layer. At the last layer, the final output representing the model prediction is produced. A loss function determines the correctness of this prediction by computing the error rate between the predicted and true values. An optimization algorithm such as Stochastic Gradient Descent (SGD) [34] is used to adjust the weight of neurons by calculating the gradient of the loss function. The error rate is propagated back across the network to the input layer (known as backpropagation algorithm [35], [36]). The network then repeats this training cycle, after balancing the weights on each neuron in each cycle, until the error rate falls below a desired threshold. At this point, the DNN is trained and is ready for inference. In Figure 6, the high level mechanism of training for DL models is illustrated.

In a broad categorization, DL models fall into three categories, namely generative, discriminative, and hybrid models. Though not being a firm boundary, discriminative models usually provide supervised learning approaches, while generative models are used for unsupervised learning. Hybrid models incorporate the benefits of both discriminative and generative models.

1) Convolutional Neural Networks (CNNs): For vision-based tasks, DNNs with a dense connection between layers are hard to train and do not scale well. One important reason is the translation-invariance property of such models. They thus do not learn the features that might transform in the image (e.g., rotation of hand in pose detection). CNNs have solved this problem by supporting translation-equivariance computations. A CNN receives a 2-D input (e.g., an image or speech signal) and extracts high level features through a series of hidden layers. The hidden layers consist of convolution layers as well as fully connected layers at the end. The convolution layer is at the core of a CNN and consists of a set of learnable parameters, called filters, that have the same shape as the input's shape but with smaller dimensions. In the training process, the filter of each convolutional layer goes through the whole input volume (e.g., in case of an image, it goes across the width and length of the image) and calculates an inner product of the input and the filter. This computation over the whole input leads to a feature map of the filter.

Another building block of a CNN is the pooling layers, which operate on the feature maps. The objective of having pooling layers is to reduce the spatial size of the representation, in order to both cut down the number of parameters and computation times and to reduce the chance of overfitting. *Max pooling* is a common approach that partitions the input space

TABLE I
SUMMARY OF DEEP LEARNING MODELS

Model	Category	Learning model	Typical input data	Characteristics	Sample IoT Applications
AE	Generative	Unsupervised	Various	• Suitable for feature extraction, dimensionality reduction • Same number of input and output units • The output reconstructs input data • Works with unlabeled data	• Machinery fault diagnosis • Emotion recognition
RNN	Discriminative	Supervised	Serial, time-series	• Processes sequences of data through internal memory • Useful in IoT applications with time-dependent data	• Identify movement pattern • Behavior detection
RBM	Generative	Unsupervised, Supervised	Various	• Suitable for feature extraction, dimensionality reduction, and classification • Expensive training procedure	• Indoor localization • Energy consumption prediction
DBN	Generative	Unsupervised, Supervised	Various	• Suitable for hierarchical features discovery • Greedy training of the network layer by layer	• Fault detection classification • Security threat identification
LSTM	Discriminative	Supervised	Serial, time-series, long time dependent data	• Good performance with data of long time lag • Access to memory cell is protected by gates	• Human activity recognition • Mobility prediction
CNN	Discriminative	Supervised	2-D (image, sound, etc.)	• Convolution layers take biggest part of computations • Less connection compared to DNNs. • Needs a large training dataset for visual tasks.	• Plant disease detection • Traffic sign detection
VAE	Generative	Semi-supervised	Various	• A class of Auto-encoders • Suitable for scarcity of labeled data	• Intrusion detection • Failure detection
GAN	Hybrid	Semi-supervised	Various	• Suitable for noisy data • Composed of two networks: a generator and a discriminator	• Localization and wayfinding • Image to text
Ladder Net	Hybrid	Semi-supervised	Various	• Suitable for noisy data • Composed of three networks: two encoders and one decoder	• Face recognition • Authentication

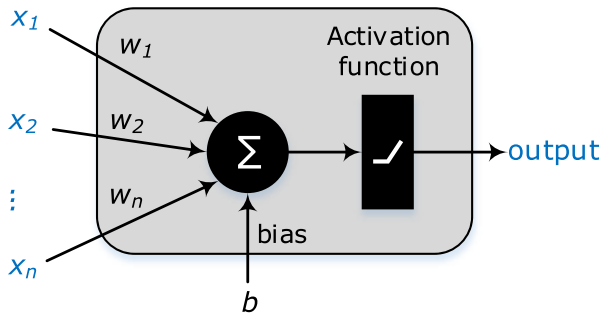


Fig. 5. A neuron is a unit of artificial neural networks, with several inputs and trainable weights and bias.

into non-overlapping regions and picks the maximum value for each region.

The last important component in CNN is ReLU, which consist of neurons with activation function in the form of $f(x) = \max(0, x)$. The introduction of this activation function in CNN results in a faster training time without affecting the generalization of the network in a sensible negative way [37]. Figure 7 depicts the structure of a CNN.

A main difference between CNNs and fully connected networks is that each neuron in CNNs is connected only to a small subset of the input. This decreases the total number of parameters in the network and enhances the time complexity of the training process. This property is called local connectivity.

Many IoT devices, such as drones, smart phones, and smart connected cars, are equipped with cameras. The CNN architecture and its variations have been investigated for a variety of application scenarios that involve these devices. Some typical applications include flood or landslide prediction through drone images, plant disease detection using plant pictures on smart phones, and traffic sign detection using vehicles' cameras.

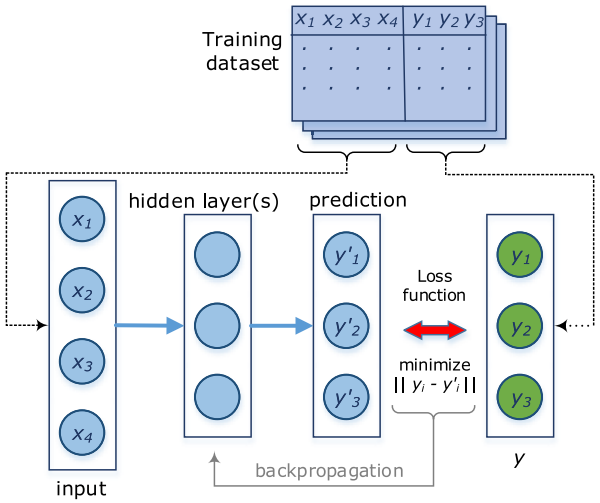


Fig. 6. The overall mechanism of training of a DL model.

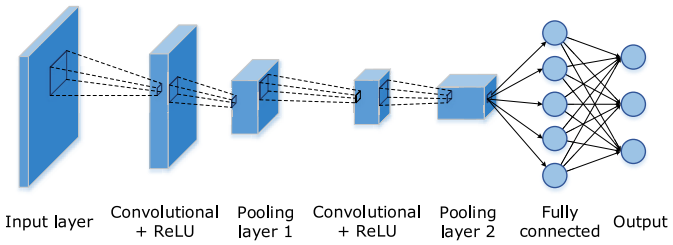


Fig. 7. Architecture of a CNN.

2) Recurrent Neural Networks (RNNs): In many tasks, prediction is dependent on several previous samples such that, in addition to classifying individual samples, we also need to analyze the sequences of inputs. In such applications, a feed-forward neural network is not applicable since it assumes no dependency between input and output layers. RNNs have been developed to address this issue in sequential (e.g., speech or text) or time-series problems (sensor's data) with various length. Detecting drivers' behaviors in smart vehicles, identifying individual's movement patterns, and estimating energy consumption of a household are some examples where RNNs can be applied. The input to an RNN consists of both the current sample and the previous observed sample. In other words, the output of an RNN at time step $t-1$ affects the output at time step t. Each neuron is equipped with a feedback loop that returns the current output as an input for the next step. This structure can be expresses such that each neuron in an RNN has an internal memory that keeps the information of the computations from the previous input.

To train the network, an extension of the backpropagation algorithm, called Backpropagation Through Time (BPTT) [38], is used. Due to the existence of cycles on the neurons, we cannot use the original backpropagation here, since it works based on error derivation with respect to the weight in their upper layer, while we do not have a stacked layer model in RNNs. The core of BPTT algorithm is a technique called unrolling the RNN, such that we come up with a feed-forward network over time spans. Figure 8 depicts the structure of an RNN and unrolled concept.

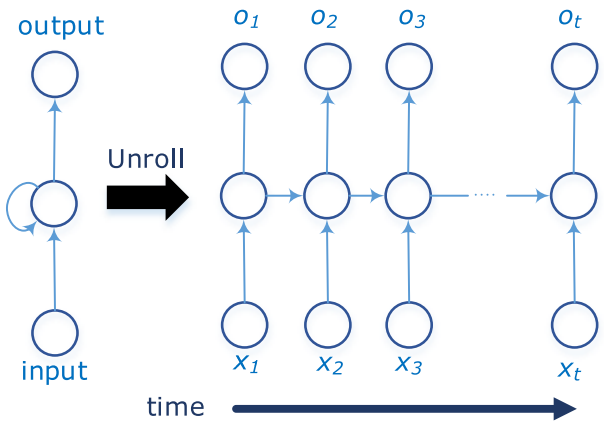


Fig. 8. Structure of a recurrent neural network.

Traditional RNNs can be considered as deep models since they can be seen as several non-linear layers of neurons between the input layer and the output layer when they are unfolded in time [39]. However, considering the architecture and the functionality of RNNs, the hidden layers in RNNs are supposed to provide a memory instead of a hierarchical representation of features [40]. There are several approaches to make RNNs deeper, including adding more layers between the input and hidden layers, stacking more hidden layers, and adding more layers between hidden layers and the output layer [39].

3) Long Short Term Memory (LSTM): LSTM is an extension of RNNs. Different variations of LSTM have been proposed, though most of them have followed the same design of the original network [31]. LSTM uses the concept of gates for its units, each computing a value between 0 and 1 based on their input. In addition to a feedback loop to store the information, each neuron in LSTM (also called a memory cell) has a multiplicative forget gate, read gate, and write gate. These gates are introduced to control the access to memory cells and to prevent them from perturbation by irrelevant inputs. When the forget gate is active, the neuron writes its data into itself. When the forget gate is turned off by sending a 0, the neuron forgets its last content. When the write gate is set to 1, other connected neurons can write to that neuron. If the read gate is set to 1, the connected neurons can read the content of the neuron. Figure 9 depicts this structure.

An important difference of LSTMs compared to RNNs is that LSTM units utilize forget gates to actively control the cell states and ensure they do not degrade. The gates can use *sigmoid* or *tanh* as their activation function. In fact, these activation functions cause the problem of vanishing gradient during backpropagation in the training phase of other models using them. By learning what data to remember in LSTMs, stored computations in the memory cells are not distorted over time. BPTT is a common method for training the network to minimize the error.

When data is characterized by a long dependency in time, LSTM models perform better than RNN models [41]. This long lag of dependency can be observed in IoT applications such as human activity recognition, predicting educational

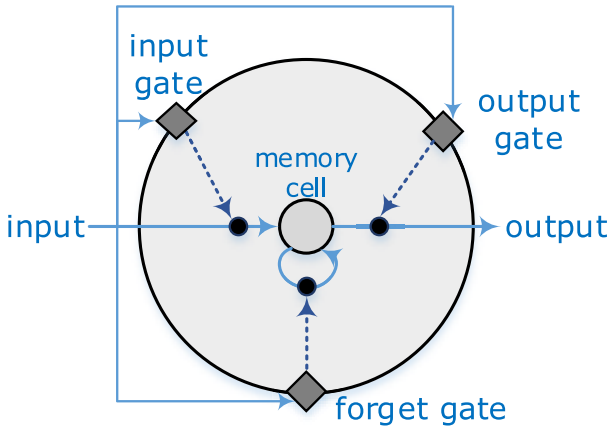


Fig. 9. Structure of a LSTM memory cell. Solid arrow lines show the flow of data and dashed arrow lines show the signals coming from gates.

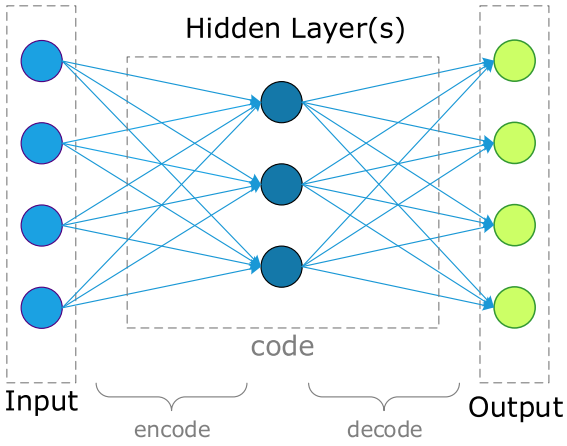


Fig. 10. Structure of an autoencoder network.

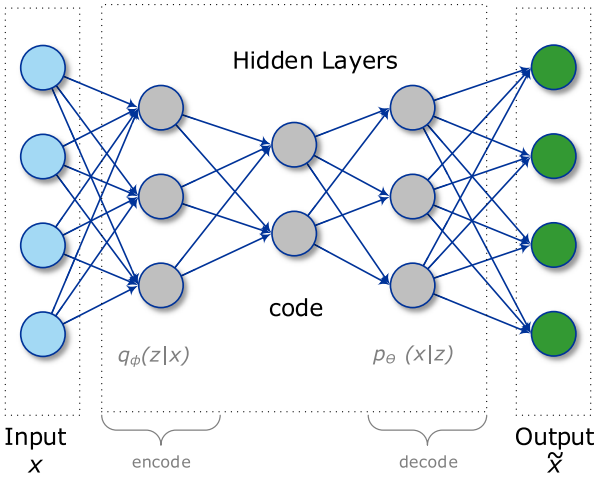


Fig. 11. Structure of a variational autoencoder network.

performance in online programs, and disaster prediction based on environmental monitoring, to name a few.

4) Autoencoders (AEs): AEs consist of an input layer and an output layer that are connected through one or more hidden layers. AEs have the same number of input and output units. This network aims to reconstruct the input by transforming inputs into outputs with the simplest possible way, such that it does not distort the input very much. This kind of neural networks has been used mainly for solving unsupervised learning problems as well as transfer learning [42]. Due to their behavior of constructing the input at the output layer, AEs are mainly used for diagnosis and fault detection tasks. This is of great interest for industrial IoT to serve many applications such as fault diagnosis in hardware devices and machines, and anomaly detection in the performance of assembly lines.

AEs have two main components: An encoder and a decoder. The encoder receives the input and transforms it to a new representation, which is usually called a code or latent variable. The decoder receives the generated code at the encoder, and transforms it to a reconstruction of the original input. The training procedure in AEs involves minimizing reconstruction error, i.e., the output and input showing minimal difference. Figure 10 illustrates the structure of a typical AE. There are several variations and extensions of AEs like denoising AE, contractive AE, stacked AE, sparse AE, and variational AE.

5) Variational Autoencoders (VAEs): VAEs, introduced in 2013, are a popular generative model framework whose assumptions on the structure of the data is not strong, while having a fast training process through backpropagation [43]. Moreover, this model has been used for semi-supervised learning [44]. Therefore, it is a good fit for IoT solutions that deal with diverse data and the scarcity of labeled data. Such applications include failure detection in sensing or actuating levels and intrusion detection in security systems. For each data point $\mathbf{x}$, there is a vector of corresponding latent variables denoted by $\mathbf{z}$. The training architecture of a VAE consists of an encoder and a decoder with parameters ϕ and θ, respectively. A fixed form distribution $q_\phi(\mathbf{z}|\mathbf{x})$ helps the encoder in estimating the posterior distribution $p_\theta(\mathbf{z}|\mathbf{x})$. The model consists of two networks: One generating samples and the other performing approximate inference. A schematic of the VAE is depicted in Figure 11.

6) Generative Adversarial Networks (GANs): GANs, introduced by Goodfellow *et al.* [45], consist of two neural networks, namely the generative and discriminative networks, which work together to produce synthetic and high-quality data. The former network (a.k.a. the generator) is in charge of generating new data after it learns the data distribution from a training dataset. The latter network (a.k.a. the discriminator) performs discrimination between real data (coming from training data) and fake input data (coming from the generator). The generative network is optimized to produce input data that is deceiving for the discriminator (i.e., data that the discriminator cannot easily distinguish whether it is fake or real). In other words, the generative network is competing with an adversary discriminative network. Figure 12 depicts the concept of GANs.

The objective function in GANs is based on minimax games, such that one network tries to maximize the value function and the other network wants to minimize it. In each step of this imaginary game, the generator, willing to fool the discriminator, plays by producing a sample data from random noise. On the other hand, the discriminator receives several

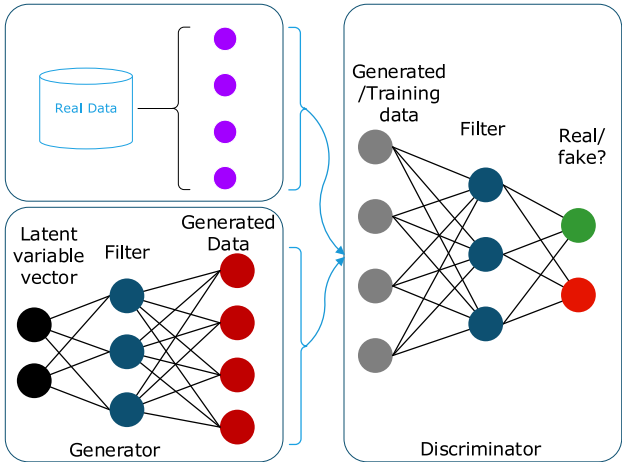


Fig. 12. Concept of a generative adversarial network.

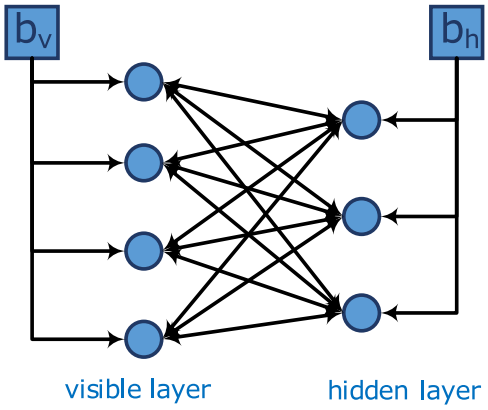


Fig. 13. Structure of a restricted Boltzmann machine. The visible and hidden layers have separate bias.

real data examples from the training set along with the samples from the generator. Its task is then to discriminate real and fake data. The discriminator is considered to perform satisfactorily if its classifications are correct. The generator also is performing well if its examples have fooled the discriminator. Both discriminator and generator parameters then are updated to be ready for the next round of the game. The discriminator's output helps the generator to optimize its generated data for the next round.

In IoT applications, GANs can be applied for scenarios that require the creation of something new out of the available data. This can include applications in localization and way-finding, where a generator network in GAN produces potential paths between two points, while the discriminator identifies which paths look viable. GANs are also very helpful for developing services for visually impaired people, such as images-to sound-converters using both GANs to generate descriptive texts from a given image [46] and another DL model to perform text-to-speech conversion. In an image processing research using GANs, a large number of real celebrity snapshots have been analyzed to create new fake images such that a human cannot identify if they are real images or not [47].

7) Restricted Boltzmann Machine (RBMs): An RBM is a stochastic ANN that consists of two layers: A visible layer that contains the input that we know, and a hidden layer that contains the latent variables. The restriction in RBMs is applied to the connectivity of neurons compared to Boltzmann machine. RBMs should build a bipartite graph, such that each visible neuron should be connected to all hidden neurons and vice versa, but there is no connection between any two units in a same layer. Moreover, the bias unit is connected to all of the visible and hidden neurons. RBMs can be stacked to form DNNs. They are also the building block of deep belief networks.

The training data is assigned to visible units. The training procedure can use backpropagation and gradient descent algorithms to optimize the weights of the network. The objective of training RBM is to maximize the product of all probabilities of the visible units. The functionality of RBM is similar to the AEs as it uses forward feeding to compute the latent variables, which are in turn used to reconstruct the input using backward feeding. The structure of an RBM is shown in Figure 13.

RBMs can perform feature extraction out of input data. This happens through modeling a probability distribution over a set of inputs that is represented in a set of hidden units. For example, having a set of favorite movies of individuals, an RBM model can have a visible layer consisting of as many neurons as the number of available movies, and a hidden layer consisting of three neurons to represent three different genres such as drama, action and comedy. So, based on the application, the hidden layer can be considered as the output layer. Or it can be complemented with an additional classifier layer to perform classification based on extracted features.

From the types of potential applications where RBMs can be used, we name indoor localization, energy consumption prediction, traffic congestion prediction, posture analysis, and generally any application that benefits from extracting the most important features out of the available ones.

8) Deep Belief Network (DBNs): DBNs are a type of generative ANNs that consist of a visible layer (corresponding to the inputs) and several hidden layers (corresponding to latent variables). They can extract hierarchical representation of the training data as well as reconstruct their input data. By adding a classifier layer like softmax, it can be used for prediction tasks.

The training of a DBN is performed layer by layer, such that each layer is treated as an RBM trained on top of the previous trained layer. This mechanism makes a DBN an efficient and fast algorithm in DL [48]. For a given hidden layer in DBN, the hidden layer of previous RBM acts as the input layer. Figure 14 shows the structure of a typical DBN.

Several applications can benefit from the structure of DBNs, such as fault detection classification in industrial environments, threat identification in security alert systems, and emotional feature extraction out of images.

9) Ladder Networks: Ladder networks were proposed in 2015 by Valpola *et al.* [49] to support unsupervised learning. Later, they were extended to work in semi-supervised settings [50] and have shown state-of-the-art performance for several tasks, such as handwritten digits recognition and image classification. The architecture of a ladder network consists

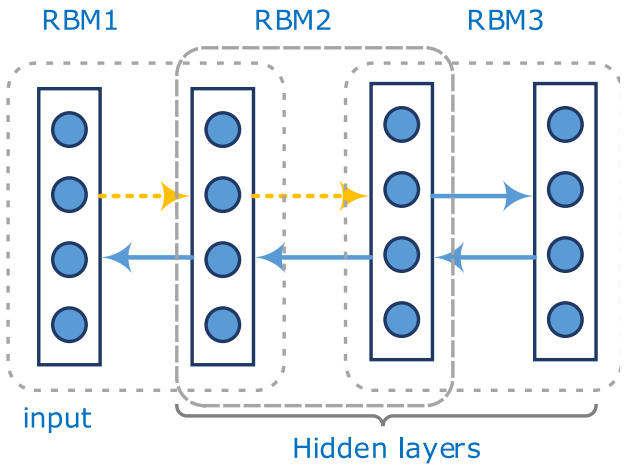


Fig. 14. Structure of a deep belief network. The dash arrows show the feature extraction path and solid arrows show the generative path.

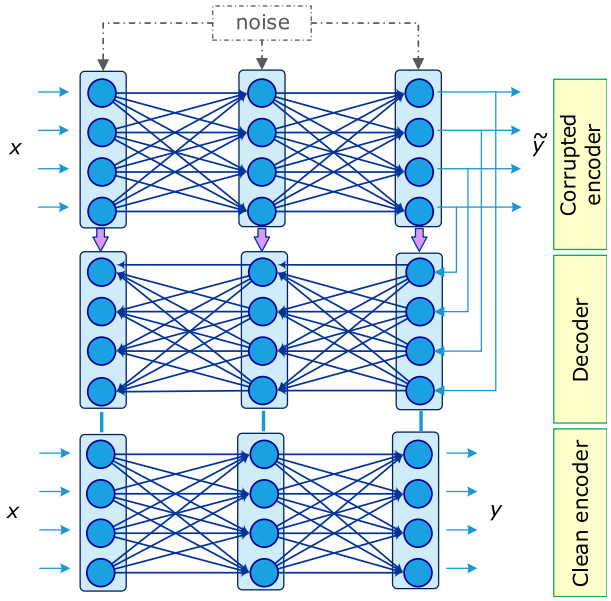


Fig. 15. Ladder network structure with two layers.

of two encoders and one decoder. The encoders act as the supervised part of the network and the decoder performs unsupervised learning. One of the encoders, called clean encoder, produces the normal computations while the other encoder, called corrupted encoder, adds Gaussian noise to all layers.

Using a denoising function, the decoder can reconstruct the representations at each layer given the corresponding corrupted data. The difference between the reconstructed and clean data at each layer is used for computing the denoising cost of that layer. In the encoder side, the cost function uses the difference between the corrupted output of encoder layers and the corresponding clean outputs. The training objective is to minimize the sum of cost in the supervised part and unsupervised network.

The initial experimental evaluations of ladder networks [49] are limited to some standard tasks, such as handwritten digits classification over the Modified National Institute of Standards and Technology (MNIST) datasets [51] or image recognition tasks on the datasets of the Canadian Institute for Advanced Research (CIFAR)-10 [52]. Though it has not been used widely in IoT scenarios, ladder networks have the potential to be used in many vision-based IoT analytics where semi-supervision is a great bonus. Figure 15 shows the structure of a ladder network.

B. Fast and Real-Time DL Architectures

The research works for fast and real-time analytics using DL models over the stream of data are still in their infancy. An initial work in this area that utilizes ANNs is done by Liang *et al.* [53]. It has extended the extreme learning machine (ELM) networks to apply an online sequential learning algorithm to single hidden layer feed-forward networks. Their framework, called OS-ELM, learns the training data one-by-one as well as chunk-by-chunk, and only newly arrived data go through the training process. This architecture is the base for the real-time manufacturing execution system that is proposed in [54]. In this work, OS-ELM has been used for shop floor object localization using RFID technology. Zou *et al.* [55] have also reported using this architecture for an indoor localization algorithm based on WiFi fingerprinting, in which the OS-ELM model can bear well the dynamic environmental changes while still showing good accuracy.

For convolutional networks, the architecture proposed by Ren *et al.*, called Faster R-CNN [56] (based on Fast R-CNN [57]), aims to detect objects in images in real-time. Object detection in images needs more computations and hence consumes more energy compared to the image classification tasks, since the system has a large number of potential object suggestions that need to be evaluated. The proposed architecture is based on applying region proposal algorithms in full CNNs that perform object bounds prediction and objectness score computation at each position at the same time. Their evaluation of the proposed object detection architecture indicates that the run time of the system is between 5-17 frames per second (fps) given that the original input frames are re-scaled such that the shortest side of the image would be 600 pixels. Mao *et al.* [58] also used Fast R-CNN for embedded platforms reporting a run time of 1.85 fps with frames scaled to 600 pixels in the shortest side in embedded CPU+GPU platform, which have been shown to be energy-efficient with a close-to-real-time performance. However, for image processing tasks, we can consider an approach to be truly real-time when it can process and analyze 30 fps or better. Redmon *et al.* [59] developed You Only Look Once (YOLO) that has reached the performance of 45 fps for input images resized to 448 $\times$ 448, and even a smaller version of it, Fast YOLO, achieving 155 fps, which are suitable for smart cameras.

C. Joint DL With Other Approaches

DL architectures also have been used jointly in other machine learning approaches to make them more efficient. The nonlinear function approximation of DL models that

can support thousands or even billions of parameters is a strong motivation to use this method in other machine learning approaches in need of such functions. Moreover, the automatic feature extraction in deep models is another motivating reason to exploit these models jointly with other approaches. In the following subsections, a summary of such approaches that are suitable for IoT scenarios is provided.

1) Deep Reinforcement Learning: Deep Reinforcement Learning (DRL) [60] is a combination of reinforcement learning (RL) with DNNs. It aims to create software agents that can learn by themselves to establish successful policies for gaining maximum long-term rewards. In this approach, RL finds the best policy of actions over the set of states in an environment from a DNN model. The need for a DNN in an RL model becomes evident when the underlying environment can be represented by a large number of states. In such situation, traditional RL is not efficient enough. Instead, a DL model can be used to approximate the action values in order to estimate the quality of an action in a given state. Systems that use DRL in their context are in their infancy, but already have showed very promising results. In the field of IoT, the work presented in [61] uses DRL in a semi-supervised setting for localization in smart campus environments. The aim of this work is to localize the users based on received signals from multiple Bluetooth Low Energy (BLE) iBeacons. The learning agent uses DRL to find the best action to perform (i.e., moving in a direction like North, North-West, etc. from a starting point). The reward function is the reciprocal of the distance error to a predefined target, such that the learning agent receives more rewards when it gets closer to its intended target and vice versa. Figure 16 shows a sample result of such method when a DNN model helps for gaining more rewards in a semi-supervised setting (left sub-figure in Figure 16) and its reward interpretation to the accuracy (right sub-figure).

2) Transfer Learning With Deep Models: Transfer learning, which falls in the area of domain adaptation and multi-task learning, involves the adaptation and improvement of learning in a new domain by transferring the knowledge representation that has been learned from data of a related domain [62]. Transfer learning is an interesting potential solution for many IoT applications where gathering training data is not an easy task. For example, considering the training of a localization system through BLE or WiFi fingerprinting using smart phones, the RSSI values at a same time and location for different platforms (e.g., iOS and Android) vary. If we have a trained model for one platform, the model can be transferred to the other platform without re-collecting another set of training data for the new platform.

DL models are well matched to transfer learning due to their ability to learn both low-level and abstract representations from input data. Specifically, Stacked denoising AEs [62] and other variations of AEs [63] have been shown to perform very well in this area. Transfer learning with DNNs is still an ongoing and active research area in AI community, and we have not seen reported real-world applications in IoT.

3) Online Learning Algorithms Joint With DL: As the stream of data generated from IoT applications goes through the cloud platforms for analysis, the role of online machine learning algorithms becomes more highlighted, as the training model needs to be updated by the incremental volume of data. This is opposed to what the current technologies support, which is based on batch learning techniques, where the whole training data set should be available for training and, thereafter, the trained model cannot evolve by new data. Several research works report applying online learning techniques on various DL models, including stacked denoising AEs [64], sum-product networks [65], and RBMs [66].

D. Frameworks

The rapid growth of interest to use DL architectures in different domains has been supported by introducing several DL frameworks in recent years. Each framework has its own strength based on its supported DL architectures, optimization algorithms, and ease of development and deployment [67]. Several of these frameworks have been used widely in research for efficient training of DNNs. In this section, we review some of these frameworks.

H2O: H2O is a machine learning framework that provides interfaces for R, Python, Scala, Java, JSON, and CoffeeScript/JavaScript [68]. H2O can be run in different modes including standalone mode, on Hadoop, or in a Spark Cluster. In addition to common machine learning algorithms, H2O includes an implementation of a DL algorithm, which is based on feed-forward neural networks that can be trained by SGD with backpropagation. H2O's DL AE is based on the standard deep (multi-layer) neural net architecture, where the entire network is learned together, instead of being stacked layer-by-layer.

Tensorflow: Initially developed for Google Brain project, Tensorflow is an open source library for machine learning systems using various kinds of DNNs [69]. It is used by many Google products including Google Search, Google Maps and Street View, Google Translate, YouTube and some other products. Tensorflow uses graph representations to build neural network models. Developers can also take advantage of TensorBoard, which is a package to visualize neural network models and observe the learning process including updating parameters. Keras[2] also provides a high level of programming abstraction for Tensorflow.

Torch: Torch is an open source framework for machine learning containing a wide range of DL algorithms for easy development of DNN models [70]. It has been developed upon Lua programming language to be light-weight and fast for training DL algorithms. It is used by several companies and research labs like Google, Facebook, and Twitter. It supports developing machine learning models for both CPUs and GPUs, and provides powerful parallelization packages for training DNNs.

Theano: Theano is an open source Python-based framework for efficient machine learning algorithms, which supports compiling for CPUs and GPUs [71]. It uses the CUDA library in optimizing the complicated codes that need to be run on GPUs. It also allows parallelism on CPUs. Theano uses graph representations for symbolic mathematical expressions. Through

[2]https://keras.io/

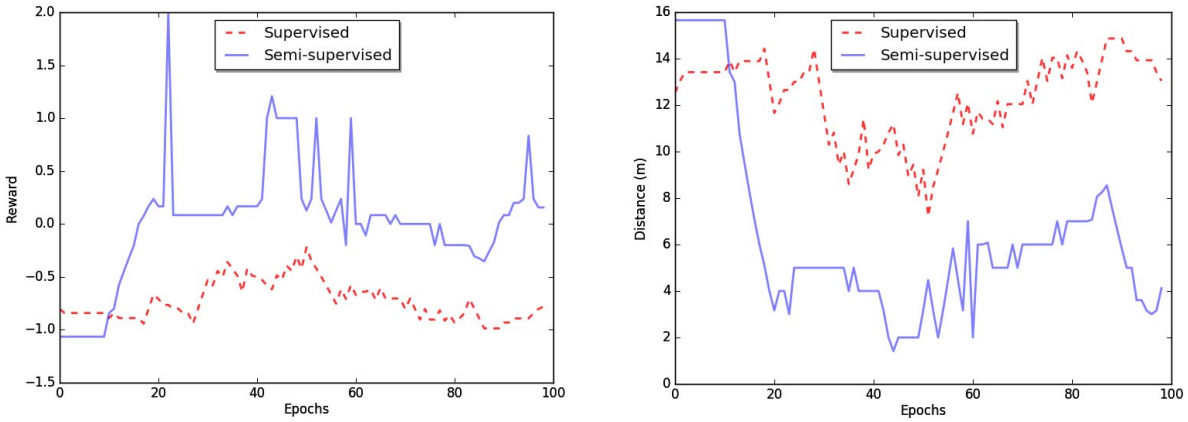


Fig. 16. Deep reinforcement learning (supervised and semi-supervised): Obtaining rewards (left) and their corresponding accuracy measurement (right) [61].

this representation, symbolic differentiation of mathematical expressions is supported in Theano. Several wrappers including Pylearn2, Keras, and Lasagne provide easier programming experience on top of Theano [72].

Caffe: Caffe [73] is an open source framework for DL algorithms and a collection of reference models. It is based on C++, supports CUDA for GPU computations, and provides interfaces for Python and MATLAB. Caffe separates model representation from its implementation. This has been made possible by defining models by configurations without hard-coding them in the source code. Switching between platforms (e.g., CPU to GPU or mobile devices) is easy by only changing a flag. Its speed on GPU is reported to be 1 ms/image for prediction and 4 ms/image for training.

Neon: Neon[3] is another open source DL framework based on Python with high performance for modern DNNs, such as AlexNet [37], Visual Geometry Group (VGG) [74], and GoogleNet [75]. It supports developing several commonly used models, such as CNNs, RNNs, LSTMs, and AEs, on both CPUs and GPUs. The list is being extended as they implemented GANs for semi-supervised learning using DL models. It also supports easy changing of the hardware platform back-ends.

Bahrampour *et al.* [67] have provided a comparative study for four of the aforementioned tools namely, Caffe, Neon, Theano and Torch. Although the performance of each tool varies in different scenarios, Torch and Theano showed the overall best performance in most of the scenarios. Another benchmarking is provided in [76], comparing the running performance of Caffe, TensorFlow, Torch, CNTK, and MXNet. Table II summarizes and compares different DL frameworks.

E. Lessons Learned

In this section, we reviewed several common DL architectures that can serve in the analytics component of various IoT applications. Most of these architectures work with various types of input data generated by IoT applications. However, to get better performance for serial or time-series data, RNNs and their variations are recommended. In particular, for long term dependencies among data points, LSTM is more favorable due to its concept of gates for memory cells. For cases where the input data is more than one-dimensional, variations of CNNs work better. RBM, DBN, and variations of AE perform well in handling high-dimensionality reduction, and hierarchical feature extraction. Combined with a classification layer, they can be used for a variety of detection and forecasting scenarios. More recent architectures including VAE, GAN, and Ladder Networks are expected to have a great impact on IoT applications since they cover semi-supervised learning. Those are more favorable for IoT applications where a huge amount of data is generated while a small fraction of that can be annotated for machine learning. The role of these architectures can be emphasized by knowing that only about 3% of all universe data by 2012 was annotated, and is hence useful for supervised machine learning [89]. Table I summarizes DL architectures.

A few attempts toward making DL architectures fast and real-time responsive were also discussed. This avenue needs more exploration and research to be applicable in many time-sensitive IoT applications. Emerging machine learning architectures and techniques that both benefit from DL and address the specific IoT application requirements were also highlighted. Indeed, DRL can support autonomousness of IoT applications, transfer learning can fill the gap of lack of training data sets, and online learning matches the need for stream analysis of IoT data.

We also reviewed several common and powerful frameworks for the development of DL models. For IoT applications, training times, run times, and dynamic update of the trained models are determining factors for a reliable and efficient analytic module. Most of the current frameworks follow the pattern of "define-and-run" instead of "define-by-run" [85]. The former does not allow dynamic updates of the model while the latter supports such modifications. Chainer [85] is a framework that follows the latter pattern and can handle dynamic changes of the model.

IV. DL Applications in IoT

DL methods have been shown promising with state-of-the-art results in several areas, such as signal processing, natural language processing, and image recognition. The trend is

[3]http://neon.nervanasys.com

TABLE II
PROPERTIES OF FRAMEWORKS FOR DEVELOPING DEEP LEARNING (AS OF SEPTEMBER 2017)

Frameworks	Core Language	Interface	Pros	Cons	Used in IoT Application
H2O	Java	R, Python, Scala, REST API	• Wide range of interfaces	• Limited number of models are supported • Is not flexible	[77]
Tensorflow	C++	Python, Java, C, C++, Go	• Fast on LSTM training • Support to visualize networks	• Slower training compared to other Python-based frameworks	[78]
Theano	Python	Python	• Supports various models • Fast on LSTM training on GPU	• Many low level APIs	[79]
Torch	Lua	C, C++	• Supports various models • Good documentation • Helpful error debugging messages	• Learning a new language	[78] [80]
Caffe	C++	Python, Matlab	• Provides a collection of reference models • Easy platform switching • Very good at convolutional networks	• Not very good for recurrent networks	[81]–[83]
Neon	Python	Python	• Fast training time • Easy platform switching • Supports modern architectures like GAN	• Not supporting CPU multi-threading	[84]
Chainer [85]	Python	Python	• Supports modern architectures • Easier to implement complex architectures • Dynamic change of model	• Slower forward computation in some scenarios	[86]
Deeplearning4j	Java	Python, Scala, Clojure	• Distributed training • Imports models from major frameworks (e.g., TensorFlow, Caffe, Torch and Theano) • Visualization tools	• Longer training time compared to other tools	[87], [88]

going up in IoT verticals. Some neural network models work better in special domains. For example, convolutional networks provide better performance in applications related to vision, while AEs perform very well with anomaly detection, data denoising, and dimensionality reduction for data visualization. It is important to make this link between the kind of neural network model that best fits each of the different application domains.

In this section, we review successful applications of DL in IoT domains. Based on our observation, many IoT related applications utilize vision and image classification (like traffic sign recognition, or plant disease detection that we will discuss in Section IV-B) as their base intelligent service. There are other services, such as human pose detection, which are utilized for smart home applications or intelligent car assistance. We identify several kinds of these services as foundational services on which other IoT applications can be built. The common property of these services is that they should be treated in a fast analytic mode instead of piling their data for later analytics. Indeed, each domain may have specific services beyond these foundational services. Figure 17 shows the foundational services and the IoT applications on top of them.

In the following subsections, we first review foundational services of IoT that use DL as their intelligence engine, then highlight the IoT applications and domains where a combination of foundational services as well as specific ones may be utilized.

A. Foundational Services

1) Image Recognition: A large portion of IoT applications address scenarios in which the input data for DL is in the form of videos or images. Ubiquitous mobile devices equipped with high resolution cameras facilitate generating images and videos by everyone, everywhere. Moreover, intelligent video cameras are used in many places like smart homes, campuses, and manufacturers for different applications. Image recognition/classification and object detection are among the fundamental usages of such devices.

One issue with the IoT-related systems that have addressed image recognition is the use of specific source datasets for evaluation of their performance. Most of these systems employ the available common image datasets such as the MNIST dataset of handwritten digits [51], VGG face dataset [74], CIFAR-10 and CIFAR-100 tiny images dataset, etc. Though being good for comparison with other approaches, those datasets do not show the specific characteristics of IoT systems. For example, the input for the task of vehicle detection in smart connected cars would not be always a clear image, and there are cases where the input image is at night, or in a rainy or foggy weather. These cases are not handled through the available datasets and hence the models trained based on these datasets are not comprehensive enough.

2) Speech/Voice Recognition: With the massive proliferation of smart mobile devices and wearables, automatic speech recognition is becoming a more natural and convenient way for

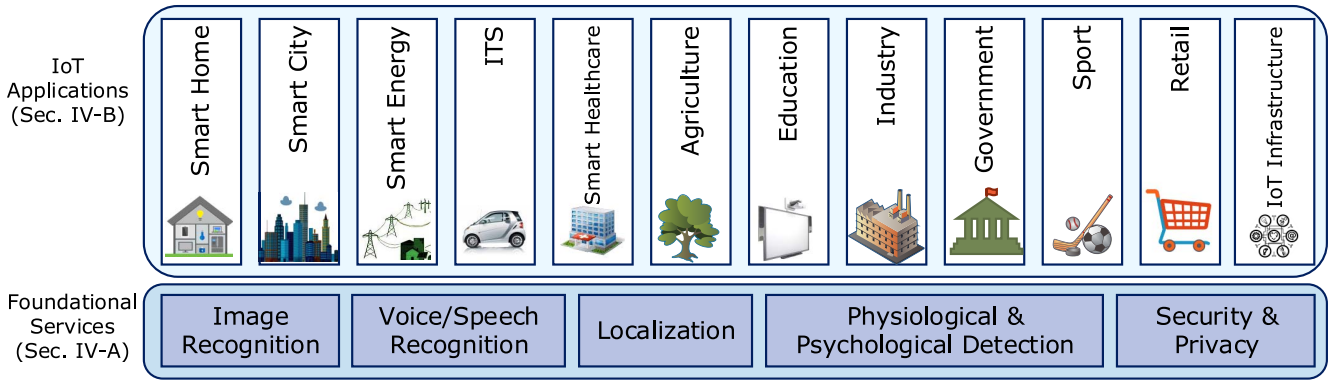


Fig. 17. IoT applications and the foundational services.

people to interact with their devices [90]. Also, the small size of mobile devices and wearables nowadays lower the possibility of having touch screens and keyboards as a means of input and interaction with these devices. However, the main concern for providing speech/voice recognition functionality on resource-constrained devices is its energy-intensiveness, especially when the data is processed through neural networks. In a typical speech recognition neural network model, voice data is represented as the raw input to the network. The data is processed through the hidden layers, and the likelihood of the voice data to a particular speech sound is presented at the output layer.

Price *et al.* [91] have reported that they have built a special-purpose low-power DL chip for automatic speech recognition. The new specialized chip consumes a tiny amount of energy between 0.2 and 10 milliwatts, 100 times lesser than the energy consumption for running a speech recognition tool in current mobile phones. In the new chip, DNNs for speech recognition have been implemented. For the sake of energy saving, three levels of voice activity recognition are designed with three separate neural networks, each of which having a different level of complexity. A lowest complexity network, thus consuming the lowest amount of energy, detects voice activity by monitoring the noise in the environment. If this network identifies a voice, the chip runs the next complexity level recognition network whose task is acoustic modeling to identify if the voice looks like speech. If the output of this network is a high likelihood, then the third network, having the highest energy consumption, is triggered to run to identify individual words.

3) Indoor Localization: Providing location aware services, such as indoor navigation and location aware marketing in retailers, are becoming prevalent in indoor environments. Indoor localization may also have applications in other sectors of IoT, such as in smart homes, smart campuses, or hospitals. The input data generated from such applications usually comes from different technologies, such as vision, visible light communication (VLC), infrared, ultrasound, WiFi, RFID, ultrawide band, and Bluetooth. For the approaches based on WiFi or Bluetooth, most of the literature have used mobile phones for receiving signals from the fixed transmitters (i.e., access points or iBeacons), which are called fingerprints. Among these fingerprinting approaches, several attempts reported the use of DL models to predict the location [92]–[94].

DL has been used successfully to locate indoor positions with high accuracy. In a system called DeepFi [92], a DL method over fingerprinting WiFi channel state information data has been utilized to identify user positions. This system consists of offline training and online localization phases. In the offline training phase, DL is exploited to train all the weights based on the previously stored channel state information fingerprints. Other works [93], [94] report using variations of DL models in conjunction with other learning methods to extract features and estimate positions. These experiments assert that the number of hidden layers and units in DL models has a direct effect on the localization accuracy. In [95], a CNN is used for indoor localization by fusion of both magnetic and visual sensing data. Moreover, a CNN has been trained in [96] to determine the indoor positions of users by analyzing an image from their surrounding scene.

Lu *et al.* [97] have also used LSTM networks for localizing soccer robots. In this application, data collected from several sensors, namely Inertia Navigation System (INS) and vision perceptions, are analyzed to predict the position of the robot. The authors reported improved accuracy and efficiency compared to two baseline methods, namely standard Extended Kalman Filtering (EKF) and the static Particle Filtering.

4) Physiological and Psychological State Detection: IoT combined with DL techniques has been also employed to detect various physiological and psychological states of humans, such as pose, activity, and emotions. Many IoT applications incorporate a module for human pose estimation or activity recognition to deliver their services, e.g., smart homes, smart cars, entertainment (e.g., XBox), education, rehabilitation and health support, sports, and industrial manufacturing. For example, convenient applications in smart homes are built based on the analysis of occupant's poses. The cameras transfer the video of the occupant to a DNN to find out the pose of the person and take the most appropriate action accordingly. Toshev and Szegedy [98] report a system employing a CNN model to achieve this goal. This sort of services can also be used in education to monitor the attention of students, and in retail stores to predict the shopping behavior of customers [99].

Ordóñez and Roggen [100] have proposed a DL framework that combines the strength of CNN and LSTM neural networks for human activity recognition from wearable sensor data (accelerometer, gyroscope, etc.). Their model consists of

four convolutional layers with ReLUs followed by two LSTM layers and a softmax layer. They showed that this combination outperformed a baseline model that is just based on convolutional layers by 4% on average. The work of Tao *et al.* [101] also used LSTM architecture for human activity recognition based on mobile phone sensor's data. Li *et al.* [102] also report the usage of raw data from passive FRID tags for detecting medical activities in a trauma room (e.g., blood pressure measurement, mouth exam, cardiac lead placement, etc.) based on a deep CNN.

In [103], a combined model of CNN and RNN was proposed for gesture recognition in video frames. This model showed better results compared to the models without such combination, and asserted the importance of the recurrence component for such task. Fragkiadaki *et al.* [104] proposed a DNN model called Encoder-Recurrent-Decoder (ERD) for human body pose recognition and motion prediction in videos and motion capture data sets. The proposed model consisted of an RNN with an encoder before the recurrent layers and a decoder after them. This architecture was shown to outperform Conditional Restricted Boltzmann Machines (CRBMs) for this application.

Beyond the physical movements, emotion estimation of humans from video frames has been also investigated in [105] using a model that consists of a CNN, DBN, and AE. Furthermore, the work in [106] used mobile inertial sensor data for motion detection. It confirmed that human motion patterns can be used as a source of user identification and authentication. The employed model in this system is a combination of convolutional layers and clockwork RNN.

5) Security and Privacy: Security and privacy is a major concern in all IoT domains and applications. Smart homes, ITS, Industry, smart grid, and many other sectors consider security as a critical requirement. Indeed, the validity of the functionality of the systems depends on protecting their machine learning tools and processes from attackers.

False Data Injection (FDI) is a common type of attack on data-driven systems. He *et al.* [107] proposed a Conditional DBN to extract FDI attack features from the historical data of smart grids, and use these features for attack detection in real-time. The work in [108] is also related to anomaly detection that may occur in vehicle networks.

Smart phones as great contributers to IoT data and applications are also under serious threats of hacker attacks. Consequently, protecting these devices from a variety of security issues is necessary for IoT perspectives beyond the users' concerns. Yuan *et al.* [109] proposed a DL framework to identify malwares in Android apps. Their architecture is based on a DBN by which they reported accuracy of 96.5% to detect malware apps.

The security and privacy preservation of deep machine learning approaches are the most important factors for the acceptance of using these methods in IoT sectors. Shokri and Shmatikov [110] proposed a method to address the privacy preservation issues in DL models when they are subject to distributed learning. Their approach was able to preserve both the privacy of participants' training data and the accuracy of the models at the same time. The core of their approach is based on the fact that stochastic gradient descent optimization algorithms, used in many DL architectures, can be performed in a parallel and asynchronous way. Individual participants can thus independently train the model on their own data and share a portion of their model parameters with other participants. Abadi *et al.* [111] also proposed a method for privacy guarantee in DL models using differentially private stochastic gradient descent algorithm.

B. Applications

1) Smart Homes: The concept of smart homes involve a broad range of applications based on IoT, which can contribute to enhancing homes' energy usage and efficiency, as well as the convenience, productivity, and life-quality of their occupants. Nowadays, home appliances can connect to the Internet and provide intelligent services. For example, Microsoft and Liebherr in a collaborative project are applying Cortana DL to the information gathered from inside the refrigerator [112]. These analytics and predictions can help the household to have a better control on their home supplies and expenses, and, in conjunction with other external data, can be used for monitoring and predicting health trends.

Over one third of the generated electricity in the U.S. is consumed by the residential sector [113], with HVAC and lighting devices consisting the largest source of such consumption in buildings. This demand is expected to grow in a slower pace by smart management of energy as well as the efficiency improvements in appliances. Hence, the ability to control and improve energy efficiency and predict the future need is a must for smart home systems. In the smart home applications, electricity load prediction are the most common applications that employ different DL networks to figure out the task. Manic *et al.* [113] performed a comparison analysis of load forecasting for home energy consumption using three DL architectures, including LSTM, LSTM Sequence-to-Sequence (S2S) and CNN. Their results show that LSTM S2S predicts the future usage better than the other architectures, followed by CNN, and then LSTM. They also compared the same dataset over a conventional ANN, and all of the aforementioned models outperformed the ANN model.

2) Smart City: Smart city services span over several IoT domains, such as transportation, energy, agriculture [114], etc. However, this area is more interesting from a machine learning perspective as the heterogeneous data coming from different domains lead to big data, which can result in high-quality output when analyzed using DL models. Smart city benefits from advances in other domain to achieve efficient resource management for the whole city. For example, to improve public transportation infrastructure and offer new improved services, getting analytics and patterns out of public transportation behaviors is of interest for local authorities.

Toshiba has recently developed a DL testbed jointly with Dell Technologies, and used this testbed in a Smart Community Center in Kawasaki, Japan, to evaluate the data collected in the Center [115]. The aim of running the testbed is to measure the effectiveness of using DL architectures in IoT ecosystems, and identify the best practices for service improvement including increasing machines' availability,

optimizing monitoring sensors, and lowering maintenance expenses. The big data that feeds the testbed were gathered from building management, air conditioning and building security.

One important issue for smart city is predicting crowd movements patterns, and their use in public transportation. Song *et al.* [116] developed a system based on DNN models to achieve this goal on a city level. Their system is built upon a four-layer LSTM neural network to learn from human mobility data (GPS data) joint with their transportation transition modes (e.g., stay, walk, bicycle, car, train). They treated the prediction of people's mobility and transportation mode as two separated tasks instead of joining all these features together. Consequently, their learning system is based on a multi-task deep LSTM architecture to jointly learn from the two sets of features. The choice of LSTM was driven by the spatio-temporal nature of human mobility patterns. The authors assert that their approach based on multi-task deep LSTM achieves better performance compared to both shallow LSTMs having only one single LSTM layer as well as deep LSTMs without multi-tasking.

Liang *et al.* [117] presented a real-time crowd density prediction system in transportation stations that leverages the mobile phone users' telecommunication data known as caller detail record (CDR). CDR data are gathered when a user takes a telecommunication action (i.e., call, SMS, MMS, and Internet access) on the phone, which usually includes data about user ID, time, location, and the telecommunication action of the user. They built their system based on an RNN model for metro stations, and reported more accurate predictions compared to nonlinear autoregressive neural network models.

Waste management and garbage classification is another related task for smart cities. A straightforward method to perform this automation is through vision-based classifications using deep CNNs as it has been done in [81]. Monitoring air quality and predicting the pollution is another direction for city management. Li *et al.* [118] developed a DL-based air quality prediction model using a stacked AE for unsupervised feature extraction and a logistic regression model for regression of the final predictions.

Amato *et al.* [119] developed a decentralized system to identify the occupied and empty spots in parking lots using smart cameras and deep CNNs. The authors deployed a small architecture of a CNN on smart cameras, which are equipped with Raspberry Pi 2 model. These embedded devices in smart cameras can thus run the CNN on each device to classify images of individual parking spaces as occupied or empty. The cameras then send only the classification output to a central server. Valipour *et al.* [120] also developed a system for detecting parking spots using CNN, which has shown improved results compared to SVM baselines. Table III summarizes the aforementioned attempts.

TABLE III
TYPICAL IoT-BASED SERVICES IN SMART CITY

Service	Reference	Input data	DL model
Crowd density/ transportation prediction	[116]	GPS/ transition mode	LSTM
	[117]	Telecommunication data/CDR	RNN
Waste management	[81]	Garbage images	CNN
Parking lot management	[119], [120]	Images of parking spaces	CNN

3) Energy: The two way communication between energy consumers and the smart grid is a source of IoT big data. In this context, smart meters are in the role of data generation and acquisition for the fine grained level of energy consumption measurement. Energy providers are interested to learn the local energy consumption patterns, predict the needs, and make appropriate decisions based on real-time analytics. Mocanu *et al.* [121] have developed a kind of RBM to identify and predict the buildings' energy flexibility in real-time. Energy flexibility is about modifying a household's electricity consumption while minimizing the impact on the occupants and operations. In the mentioned work, time-of-use and consumption of individual appliances are predicted to achieve flexible energy control. The advantage of this model beyond showing good performance and accuracy is that flexibility identification can be performed with flexibility prediction concurrently. In [122], two variations of RBMs are used to forecast energy consumption for short term intervals in residential houses. The model includes a Conditional RBM (CRBM) and a Factored Conditional RBM (FCRBM). Their results indicate that FCRBM performs better that CRBM, RNN and ANN. Moreover, by extending the forecasting horizon, FCRBM and CRBM show more accurate predictions than the RBM and ANN.

On the smart grid side, forecasting the power from solar, wind, or other types of natural sustainable sources of energy is an active research field. DL is increasingly used in many applications in this domain. For example, Gensler *et al.* [123] investigate the performance of several DL models, such as DBNs, AEs, and LSTMs, as well as MLP for predicting the solar power of 21 photovoltaic plants. For solar power prediction, a main element of the input is a numeric value for weather forecasting in a given time horizon. From their evaluation, the combination of AEs and LSTMs (Auto-LSTM) has been shown to produce the best results compared to other models, followed by DBN. The reason for obtaining a good prediction score by Auto-LSTM is that they are able to extract features from raw data, which is not the case for ANN and MLP. In [86], an online forecasting system based on LSTM is proposed to predict the solar flare power 24 hours ahead.

4) Intelligent Transportation Systems: Data from Intelligent Transportation Systems (ITS) is another source of big data that is becoming ubiquitous every day. Ma *et al.* [79] presented a system of transportation network analysis based on DL. They have employed RBM and RNN architectures as their models in a parallel computing environment, and GPS data from participating taxies as the input of the models. The accuracy of their system to predict traffic congestion evolution over one hour of aggregated data is reported to be as high as 88% which was computed within less than 6 minutes.

Reference [124] also reported the investigation on short-term traffic flow prediction. They used LSTM as their learning model and reported better accuracy for LSTM compared to other methods including SVM, simple feed forward neural networks, and stacked AEs. For different intervals (15, 30, 45, and 60 min) LSTM showed the lowest mean absolute percentage error (MAPE) rate. However, for short intervals of 15 minutes, the error rate of SVM is slightly higher than the LSTM model. This result can be interpreted by the fact that the small number of data points in short intervals does not make stronger discrimination boundaries for the classification task in the LSTM model compared to the SVM model. In another study [108], ITS data are exposed to an intrusion detection system based on DNN to improve the security of in-vehicular network communications.

ITS also motivate the development of methods for traffic sign detection and recognition. Applications such as autonomous driving, driver assistance systems, and mobile mapping need such sort of mechanisms to provide reliable services. Cireşan *et al.* [125] presented a traffic sign recognition system based on DNNs of convolutional and max-pooling layers. They introduced a multi-column DNN architecture that includes several columns of separate DNNs, and reported increased accuracy with this approach. The input is preprocessed by several different preprocessors, and a random number of columns receives the preprocessed input to proceed with training. The final prediction output is the average of all the DNNs' outputs. Their results show that this proposed method, achieving a recognition rate of 99.46%, has been able to recognize traffic signs better than the humans on the task with 0.62% more accuracy.

In order to be applicable in real scenarios, these analytics need to be performed in real-time. Lim *et al.* [126] proposed a real-time traffic sign detection based on CNN that has been integrated with a general purpose GPU. They reported F1 measure of at least 0.89 in their results with data having illumination changes. To have a faster inference engine, they used a CNN with two convolutional layers.

Furthermore, self-driving cars use DNNs in performing many tasks, such as detecting pedestrians, traffic signs, obstacles, etc. There are several startups that use DL in their self-driving cars to perform different tasks when driving in the streets [127].

5) Healthcare and Wellbeing: IoT combined with DL has been also employed in providing healthcare and wellbeing solutions for individuals and communities. For instance, developing solutions based on mobile apps to accurately measure dietary intakes is a track of research that can help control the health and wellbeing of individuals. Liu *et al.* [82], [128] developed a system to recognize food images and their relevant information, such as types and portion sizes. Their image recognition algorithm is based on CNNs that achieved competitive results compared to the baseline systems.

DL for classification and analysis of medical images is a hot topic in the healthcare domain. For example, Pereira *et al.* [129] used the idea of recognizing handwritten images by CNNs to help identifying Parkinson's disease in its early stages. Their model learns features from the signals of a smart pen that uses sensors to measure handwritten dynamics during the individual's exam. Muhammad *et al.* [130] propose a voice pathology detection system using IoT and cloud frameworks, in which patients' voice signals are captured through sensor devices and are sent to a cloud server for analytics. They used an extreme learning machine trained by voice signals to diagnose the pathology. In [131], DL was employed for detection of cardiovascular diseases from mammograms. In their study, Wang *et al.* used breast arterial calcification (BAC) revealed in mammograms as a sign of coronary artery disease. They developed a CNN with twelve layers to identify the existence of BAC in a patient. Their results show that the accuracy of their DL model is as good as the human experts. Although this work has been done offline, it shows the potential of developing or extending mammogram devices in IoT contexts for online and early detection of such diseases.

Feng *et al.* [132] report the use of RBMs and DBNs for fall detection in a home care environment. Normal postures in such environment are standing, sitting, bending, and lying. Lying on the floor longer than a threshold is considered as a fallen posture. Their evaluation shows that RBM outperforms DBN in terms of classification accuracy. The lack of large datasets and performing offline detection are the restrictions of their method.

Researchers also used time series medical data in conjunction with RNN based models for early diagnosis and prediction of diseases. Lipton *et al.* [133] investigated the performance of LSTM networks to analyze and recognize patterns in multivariate time series of medical measurements in intensive care units (ICUs). The input data in their system consist of sensor data of vital signs as well as lab test results. Their performance results show that an LSTM model trained on raw time-series data outperforms a MLP network. A survey of DL in health informatics is provided in [134].

6) Agriculture: Producing healthy crops and developing efficient ways of growing plants is a requirement for a healthy society and sustainable environment. Disease recognition in plants using DNNs is a direction that has shown to be a viable solution. In a study that is reported by Sladojevic *et al.* [83], the authors developed a plant disease recognition system based on the classification of leave images. They have used a deep convolutional network model implemented using the Caffe framework. In this model, diseased leaves in 13 categories can be identified from the healthy ones with an accuracy of about 96%. Such recognition model can be exploited as a smart mobile applications for farmers to identify the fruit, vegetable, or plant disease based on their leaf images captured by their mobile devices. It can also allow them to select remedies or pesticides in conjunction with complementary data.

DL also has been used in remote sensing for land and crop detection and classification [135]–[137]. The direction established in these works enabled the automated monitoring and management of the agricultural lands in large scales. In most of such works, deep convolutional networks are used to learn from images of the land or crops. In [135], it is reported that

using CNN has yielded an accuracy of 85% in detecting major crops, including wheat, sunflower, soybeans, and maize, while outperforming other approaches such as MLP and random forest (RF).

Furthermore, DL has been reported to be utilized for prediction and detection tasks for automatic farming. For example, [138] has used a DL model based on deep CNNs for obstacle detection in agricultural fields, which enables autonomous machines to operate safely in them. The proposed system was able to detect a standardized object with an accuracy between 90.8% to 99.9%, based on the field (e.g., row crops or grass mowing).

Moreover, fruit detection and finding out the stage of fruit (raw or ripe) is critical for automated harvesting. Sa *et al.* [139] used a variation of CNN, called Region-based CNN, for image analysis of fruits. The input image of the system comes in two modes: one containing RGB colors and the other is near-infrared. The information of these images are combined in the model and has achieved detection improvement compared to pixel-based training models.

7) Education: IoT and DL contribute to the efficiency of education systems, from kindergarten to higher education. Mobile devices can gather learners' data and deep analytical methods can be used for prediction and interpretation of learners progress and achievements. Augmented reality technology combined with wearables and mobile devices are also potential applications for DL methods in this area to make students motivated, lessons and studies to be interesting, and make educational learning methods to be efficient [140], [141]. Moreover, DL can be used as a personalized recommendation module [142] to recommend more relevant content to the educator. The applications of DL in other domains, such as natural language translation and text summarization, would be of help for smart education when it comes to online learning on mobile devices.

Furthermore, the advent of Massive Open Online Courses (MOOCs) and their popularity among the students has led to generating a huge amount of data from the learners' behavior in such courses. MOOCs analysis can help identify struggling students in early sessions of a course, and provide sufficient support and attention from instructors to those students to achieve a better performance. Yang *et al.* [143] proposed a method for predicting student grades in MOOCs. They use clickstream data collected from lecture videos when students are watching the video and interacting with it. Clickstream data are fed to a time series RNN that learns from both prior performance and clickstream data. In addition, Piech *et al.* applied RNN and LSTM networks to model the prediction of educator answers to exercises and quizzes, based on their past activities and interactions in MOOCs [144]. Results showed improvement over Bayesian Knowledge Tracing (BKT) methods, which employ a Hidden Markov Model (HMM) for updating probabilities of single concepts. Mehmood *et al.* [77] also used DNNs for a personalized ubiquitous e-teaching and e-learning framework, based on IoT technologies, aiming for the development and delivery of educational content in smart cities. Their proposed framework is built on top of an IoT infrastructure (e.g., smart phone sensors, smart watch sensors, virtual reality technologies) connecting the users in order to optimize the teaching and learning processes. They used DNN for human activity recognition to deliver adaptive educational content to the students.

Classroom occupancy monitoring is another application that has been investigated by Conti *et al.* [145]. In this work, the authors propose two methods for head detection and density estimation, both based on CNN architecture for counting students in a classroom. The algorithms have been deployed on off-the-shelf embedded mobile ARM platform. Their algorithm receives the images that are taken from the cameras in three classrooms with a rate of three pictures every 10 minutes. They report that the root-mean-square (RMS) error of their algorithms is at most 8.55.

8) Industry: For the industry sector, IoT and cyber-physical systems (CPS) are the core elements to advance manufacturing technologies toward smart manufacturing (a.k.a Industry 4.0). Providing high-accuracy intelligent systems is critical in such applications, as it directly leads to increased efficiency and productivity in assembly/product lines, as well as decreased maintenance expenses and operation costs. Therefore, DL can play a key role in this field. Indeed, a wide range of applications in industry (such as visual inspection of product lines, object detection and tracking, controlling robots, fault diagnosis, etc.) can benefit from introduction of DL models.

In [78], visual inspection is investigated using CNN architectures including AlexNet and GoogLeNet over different platforms (Caffe, Tensorflow, and Torch). In this work, several images of produced vehicles in the assembly line along with their annotation are submitted to a DL system. It has been found that the best performance is achieved using Tensorflow with accuracy of 94%. Moreover, Tensorflow was the fastest framework in terms of training time, where the model reached its peak accuracy in a shorter time, followed by Torch and then Caffe.

Shao *et al.* [146] used DNNs for feature extraction in a fault diagnosis (also referred as fault detection and classification (FDC)) system for rotating devices. Models using denoising auto-encoder (DAE) and contractive auto-encoder (CAE) were developed. The learned features from these models were both refined using a method called locality preserving projection (LPP), and fed to a softmax classifier for fault diagnosis. The input to the system is vibration data of a rotating device. In their system, seven operating conditions were considered, including normal operation, rubbing fault, four degrees of unbalance faults and compound faults (rub and unbalance). Given the vibration data, the diagnosis system identifies whether the device is in normal condition or in one of the fault conditions. Based on their experiments for fault diagnosis of rotor and locomotive bearing devices, the proposed approach is reported to outperform CNN and shallow learning methods.

In another study reported by Lee [147], a DBN model was proposed in conjunction with an IoT deployment and cloud platform to support fault detection of defect types in cars' headlight modules in a vehicle manufacturer setting. Their results confirmed the superior performance of the DBN model over two baseline methods, using SVM and radial basis

function (RBF), in terms of error rate in test dataset. However, the reported error rate for their training dataset in the DBN model is comparable to that of the SVM model.

For the problem of fault detection and classification (FDC) in noisy settings, [148] employed stacked denoising AEs (SdA) to both reduce the noise of sensory data caused by mechanical and electrical disturbances, and perform fault classification. Their system was applied for fault detection in wafer samples of a photolithography process. Results show that SdA leads to 14% more accuracy in noisy situations compared to several baseline methods including K-Nearest Neighbors and SVM. Yan and Yu [149] have also used SdA joint with extreme learning machines for anomaly detection in the behavior of gas turbine combustion system. Based on their results, the use of learned features by SdA leads to a better classification accuracy compared to the use of hand crafted features in their system.

9) Government: Governments can gain great potential advantages through enhanced and intelligent connectivity that comes from the convergence of IoT and DL. Indeed, a wide variety of tasks that pertains to the governments or city authorities require precise analysis and prediction. For instance, the recognition and prediction of natural disasters (landslide, hurricane, forest fires, etc.) and environmental monitoring is of high importance for governments to take appropriate actions. Optical remote sensing images that are fed to a deep AEs network and softmax classifiers were proposed by Liu and Wu [150] to predict geological landslides. An accuracy of 97.4% was reported for the proposed method, thus outperforming SVM and ANN models. In another study [151], an LSTM network is used for the prediction of earthquakes. They used the historical data from U.S. Geological Survey website for training. Their system was shown to achieve an accuracy of 63% and 74% with 1-D and 2-D input data, respectively. In another study by Liu *et al.* [84], a CNN architecture is used for detection of extreme climate events, such as tropical cyclones, atmospheric rivers and weather fronts. Training data in their system included image patterns of climate events. The authors developed their system in Neon framework and achieved accuracy of 89%-99%.

In addition, damage detection in the infrastructures of the cities, such as roads, water pipelines, etc., is another area where IoT and DL can provide benefits to governments. In [152], the problem of road damage detection was addressed using DNNs that gets its data through crowd-sourcing, which can be enabled by IoT devices. Citizens can report the damage through a mobile app to a platform. However, these citizens have no expert knowledge to accurately assess the status of road damage, which may lead to uncertain and/or wrong assessments. To eliminate these instances, the app can determine the status of the road damage by analyzing the image of the scene. The analysis is performed by a deep CNN that is trained by citizen reports as well as road manager inspection results. Since the training phase is out of the capability of mobile phones, the DL model is created on a server and trained everyday. An Android application can then download the latest model from the server upon each launch, and identify the status of road damages reported by images. Evaluations showed a damage classification accuracy of 81.4% in 1 second of analysis on the mobile devices.

10) Sport and Entertainment: Sports analytics have been evolving rapidly during the recent years and plays an important role to bring a competitive advantage for a team or player. Professional sport teams nowadays have dedicated departments or employees for their analytics [153]. Analytics and predictions in this field can be used to track the players' behavior, performance, score capturing, etc. DL is new to this area and only few works have used DNNs in different sports.

In [154], a DL method has been proposed for making an intelligent basketball arena. This system makes use of SVM to choose the best camera for real-time broadcasting from among the available cameras around the court. They also fed basketball energy images[4] to a CNN to capture the shoot and scoring clips from the non-scoring ones, hence providing accurate online score reporting and interesting highlight clips. This system was shown to achieve an accuracy of 94.59% in capturing score clips with 45 ms of processing time for each frame.

In another work by Wang and Zemel [155], an RNN has been used for classification of offensive basketball plays in NBA games. The authors used video clips of the games from SportVU[5] dataset. This dataset provides videos of the rate of 25 frames per second to detect players' unique ID, their location on the court, and the position of the ball. Their model is shown to achieve accuracy of 66% and 80% for top-1 and top-3 accuracies, respectively. Similarly, [156] used an RNN with LSTM units over the same dataset to predict the success rates of three-point shots, and reported better classification accuracy compared to gradient boosted machine (GBM) and generalized linear model (GLM).

Kautz *et al.* [157] investigated players' activity recognition in volleyball. Wearable sensor data and CNN were employed to achieve this task, and a classification accuracy of 83.2% to identify players activities was observed.

Group activity recognition is another interesting direction for sport teams. Ibrahim *et al.* [158] investigated this option in a volleyball team using a hierarchical LSTM model. In this work, a single LSTM model was built to derive the activities of each player, and a top-level LSTM model was designed to aggregate the individual models to identify the overall behavior of the team. A CNN model was utilized to extract features from video frames, and feed them to the individual LSTM models. Compared to several baseline models, the proposed hierarchical model obtained better classification results.

11) Retail: Due to the proliferation of mobile devices, online shopping has increased greatly. A recent shift toward product image retrieval through visual search techniques was noticed [159]. CNNs have been used for this visual search of clothes and fashion market, to find items in online shops that are identical or similar to what you have seen in a movie [160] or in the street [161].

[4]Basketball energy image is the spatial-temporal tracks of basketballs in the hotspot area. Hotspot area includes basketball backboard, hoop, and basket.

[5]http://go.stats.com/sportvu

Moreover, shopping for visually impaired people needs to be made convenient. A combination of IoT technologies, including smart carts, integrated with DL methods can be a solution to this problem. In [162], a visual grocery assistance system that includes smart glasses, gloves, and shopping carts was designed to help visually impaired people in shopping. This system also used a CNN to detect items in the aisles.

Moreover, check-out counters in retail stores are usually the bottlenecks where people queue up to pay their shoppings. The development of smart carts can enable real-time self check-out and enhancing such system with prediction capabilities can offer an item that a customer may need based on his/her past shopping.

Furthermore, recommending items to shoppers is a popular application of IoT for retails that uses different technologies, like BLE signals or visual cameras. The latter approach can be done through identifying the shop items or shoppers actions (e.g., reach to a shelf, retract from a shelf, etc.) [163] and providing a list of related items for the detected action.

To analyze the customer interest in merchandise, Liu *et al.* [99] proposed a customer pose and orientation estimation system based on a DNN consisting of a CNN and RNN. The input data comes from surveillance cameras. The CNN network is used to extract image features. The image features and the last predicted orientation features are then fed to an RNN to get the output pose and orientation.

12) Smart IoT Infrastructure: IoT environments consist of a large number of sensors, actuators, media and many other devices that generate big M2M and network traffic data. Therefore, the management, monitoring, and coordination of these devices are subject to processing such big data, with advanced machine learning techniques, to identify bottlenecks, improve the performance, and guarantee the quality of service.

One popular task for infrastructure management would be anomaly detection. For example, spectrum anomaly detection in wireless communications using AE was proposed by Feng *et al.* [164]. In this work, an AE model was developed to detect the anomaly that may happen due to sudden change in signal-to-noise ratio of the communications channel. The model is trained on features based on the time-frequency diagram of input signals. Their result showed that a deeper AE performs better than the conventional shallow networks. Lopez-Martin *et al.* [165] and Shone *et al.* [166] have used conditional VAE and deep AEs, respectively, for network intrusion detection. In the conditional VAE, the labels of the samples are used in addition to the latent variables as extra inputs to the decoder network.

The tiny traces of IoT traffic may not lead to congestion at the backbone. However, the need to access the channel simultaneously by a large number of IoT devices can lead to contention during the channel access phase. The contention in channel access turns to a severe problem when the access delays are increased [167]. Therefore, load balancing is a viable solution that can be performed by DL models to predict the traffic metrics and propose alternate routes. Kim and Kim [168] used DBNs to perform load balancing in IoT. Their DL model is trained on a large amount of user data and network loads. Interference identification can also be handled by DNNs as demonstrated by Schmidt *et al.* [169], where a wireless interference identification systems based on CNN was proposed. Ahad *et al.* [170] provided a survey focused on the application of neural networks to wireless networks. They reviewed related literature on quality of service and quality of experience, load balancing, improved security, etc.

Since the emerging 5th Generation (5G) cellular networks constitute one of the main pillars of IoT infrastructure, it is necessary to utilize cutting-edge technologies to enhance the different aspects of cellular networks, including radio resource management, mobility management, service provisioning management, self-organization, and to find an efficient and accurate solution for complicated configuration problems [171]. As part of these efforts, using crowd-sourced cellular networks data (e.g., signal strength) can help to come up with reliable solutions. For example, such big data can be used to create more precise coverage maps for cellular networks to improve the performance of the network [172], as was performed by the OpenSignal mobile application [173].

C. Lessons Learned

In this section, we have identified five classes of IoT services as the foundational services that can be used in a wide range of IoT applications. We discussed how DL has been used to achieve these services. Moreover, we went through a wide range of IoT domains to find out how they exploit DL to deliver an intelligent service. Table IV shows the works that utilized foundational services in IoT domains.

Many IoT domains and applications have greatly benefited from image recognition. The interest is expected to grow faster as the high-resolution cameras embedded in smart phones will result in easier generation of image and video data. The usage of other fundamental applications, especially physiological and psychological detections as well as localization, can be seen in different fields. However, the utilization of security and privacy services is shown to be limited. This is the gap in developing intelligent IoT applications, where the potential activities of hackers and attackers are ignored. Also, voice recognition with DL has not been used widely in IoT applications belonging to several domains, such as smart homes, education, ITS, and industry. There are works that use voice recognition with traditional machine learning approaches. Voice recognition has shown remarkable advancement with DL. One reason for the few appearance of this technique in IoT applications is the lack of comprehensive training datasets for each domain, as there is a need for large training datasets to train voice recognition DNNs.

Foundational services need fast data analytics to be efficient in their context. Despite several works in this direction, IoT fast data analytics based on DL has many spaces for development of algorithms and architectures.

Table V summarizes the research in each domain, and their DL model. Figure 18 also depicts the frequency of different models that have been used in the different research works. About 43% of the papers have used CNN in building their

TABLE IV
USAGE OF FOUNDATIONAL SERVICES IN IOT DOMAINS

		IoT Foundational Services				
		Image Recognition	**Voice Recognition**	**Physiological & Psychological Detection**	**Localization**	**Security & Privacy**
IoT Domains	Smart Home					
	Smart City	[81], [119], [120]			[116], [117]	
	Energy					[107]
	ITS	[125], [126]				[108]
	Healthcare	[82], [128], [129], [131]	[130]	[132]		
	Agriculture	[83], [135]–[139]				
	Education	[145]		[77]		
	Industry	[78], [147]			[54]	
	Government	[84], [150], [152]				
	Sport	[154]–[156]		[157], [158]	[97]	
	Retail	[159]–[162]		[99]	[163]	

TABLE V
THE COMMON USE OF DIFFERENT DNN MODELS IN IOT DOMAINS

Domain	**Usage of DNNs**					
	AE	**CNN**	**DBN**	**LSTM**	**RBM**	**RNN**
Image Recognition	[150]	[81]–[83], [119], [120] [125], [126], [129], [131] [135], [138], [145] [154], [158], [160]	[147]	[156]		[155], [156]
Physiological & Psychological Detection	[104], [105]	[98], [100], [102] [103], [105], [106]	[105]	[100], [101]		[103], [104], [106]
Localization	[93], [94]	[95], [96]		[97]	[92]	
Privacy and Security		[110]	[107], [109]			
Smart home		[113]		[113]		
Smart city		[81], [119], [120]		[116]		[117]
Energy	[123]		[123]	[123] [86]	[121] [122]	[122]
ITS		[125], [126]	[108]	[124]	[79]	[79]
Healthcare		[82], [128], [129], [131]	[132]	[133]	[132]	
Agriculture		[83], [135]–[139]				
Education		[145]		[144]		[143], [144]
Industry	[146], [148], [149]	[78]	[147]			
Government	[150]	[84], [152]		[151]		
Sport		[154], [157], [158]		[156], [158]		[155]
Retail		[159]–[162]				[163]
IoT Infrastructure	[164]–[166]	[169]	[168]			

proposed systems while DBN are less used compared to other models (about 7%). RNNs and LSTMs together, as time-series models, have been used in 30% of the works. The table also emphasizes the great impact of works related to image recognition on IoT applications. Moreover, one third of the IoT applications are related to time-series or serial data, in which employing RNNs is a helpful approach.

1) Complexity vs. Performance: Canziani *et al.* [174] analyzed several state-of-the-art DNN models to find out the relationship between their accuracy, memory usage, operations count, inference time, and power consumption. They found out that the accuracy and inference time show a hyperbolic relationship such that a minor increase in accuracy leads to a long computational time. They also illustrated that the number of operations in a network model have a linear relationship with the inference time. Their results also indicated that imposing an energy constraint would limit the maximum achievable accuracy. Regarding the memory footprint and batch size, the results showed that the maximum memory usage is constant during the initial memory allocation of the model and then linearly increases with the batch size. Given that the neurons are the main building blocks of a model that performs the operations, the number of operations is proportional to the number of neurons. So, the complexity can be expressed as the number of neurons in the network, such that increasing the number of neurons directly impacts the run-time.

However, there is not a clear relationship between the accuracy and number of layers (i.e., depth) or number of neurons. There are reports indicating degradation of accuracy after increasing the number of layers beyond some point. For example, Zhang *et al.* [94] assert that the number of hidden layers and neurons have a direct effect on the accuracy of the localization system. Increasing the layers initially leads to better results, but at some point when the network is made deeper, the

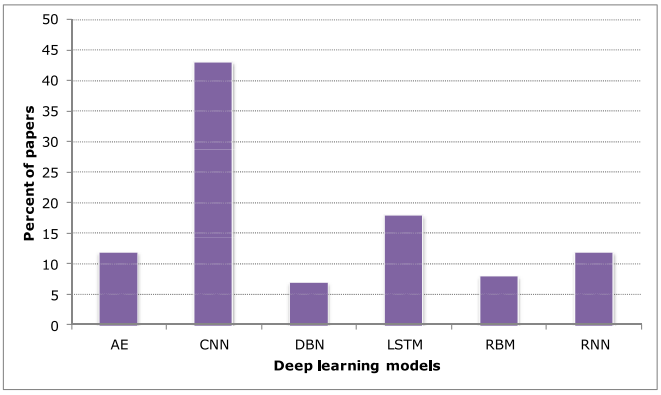


Fig. 18. The percentage of surveyed papers that have used DL models.

results start degrading. Their best result was obtained with a network of three hidden layers. On the other hand, the depth of representations has been shown to be most beneficial for many image recognition tasks [30]. The high accuracy of vision-based tasks, in part, is due to introducing deeper networks with larger number of parameters (e.g., over 1000 layers as presented in [30] and [175]). There are many hyper-parameters to optimize (e.g., epoch count, loss function, activation function, optimization function, learning rate, etc.) that complicate the process of developing good and accurate DL models. Table VI summarizes the characteristics of DNN models in several applications. In the table, the test time is for one sample unless specified otherwise.

2) Pitfalls and Criticisms: DL models were demonstrated to be as a great step toward creating powerful AI systems, but they are not a single solution for all problems. DL techniques are known as black boxes that show high predictability but low interpretability. While powerful prediction capability is most desired from the scientific perspective, the interpretability and explicability of the models are preferred from a business perspective [176]. Chollet [177] argued that problems requiring reasoning, long-term planning, and algorithmic-like data manipulation, cannot be solved by deep learning models. This is because of the nature of DL techniques that only transform one vector space into another, no matter how much data you feed them.

Moreover, there are criticisms on the performance of DNN models, suggesting that the traditional models may achieve comparable results or even outperform deep models [178]. According to Chatfield *et al.* [179], the dimensionality of the convolutional layers in CNNs can be shrunk without adversely affecting the performance. They also discussed that the shallow techniques can reach a performance analogous to that of deep CNN models if the former models use the data augmentation techniques that are commonly applied to CNN-based methods. Ba and Caruana [180] performed several empirical tests asserting that shallow FNNs can learn the complex functions and achieve accuracies that were previously possible only by deep models. In their work on optical communication systems, Eriksson *et al.* [181] showed that using pseudo random bit sequences or short repeated sequences can lead to overestimating the signal-to-noise ratio.

Generally, DL models are sensitive to the structure, and size of the data. Compared to shallow models, they work better when there is a large body of training data with a wide range of attributes. Otherwise, shallow models typically lead to better results.

V. DL on IoT Devices

Prior to the era of IoT, most research on DL targeted the improvement of its models and algorithms to efficiently operate when the scale of the problem grows to the big data, by trying to deploy efficient models on cloud platforms. The emergence of IoT has then opened up a totally different direction when the scale of the problems shrank down to resource-constrained devices and to the need for real-time analytics.

Smart objects need to support some sort of light-weight intelligence. Due to DL's successful results in speech and video applications, which are among the fundamental services and common uses of IoT, adapting its models and approaches for deployment on resource-constrained devices became a very crucial point of study. So far, DL methods can hardly be used in IoT and resource-constrained devices for training purposes since DL models require a large portion of resources, such as the processors, battery energy, and memory. In some cases, the available resources are even not sufficient for running a pre-trained DL algorithm for inference tasks [80]. Luckily, it has been recently shown that many parameters that are stored in DNNs may be redundant [182]. It is also sometimes unnecessary to use a large number of hidden layers to get a high accuracy [180]. Consequently, efficiently removing these parameters and/or layers will considerably reduce the complexity of these DNNs without significant degradation of the output [180], [182] and make them IoT-friendly. In the remaining of this section, we will discuss methods and technologies to achieve this results, and illustrate their applications in different domains.

A. Methods and Technologies

DL models may consist of millions or even billions of parameters which need sophisticated computing and large storage resources. In this section, we discuss several state-of-the-art approaches that bring DL models to IoT embedded and resource constrained devices.

1) Network Compression: One way of adopting DNNs to resource-constrained devices is through the use of network compression, in which a dense network is converted to a sparse network. This approach helps in reducing the storage and computational requirements of DNNs when they are used for classification or other sorts of inference on IoT devices. The main limitation of this approach is that they are not general enough to support all kinds of networks. It is only applicable to specific network models that can exhibit such sparsity.

Another interesting study to adopt compressed DL models on IoT devices is the one performed by Lane *et al.* [80]. In this study, the authors measure different factors that embedded, mobile, and wearable devices can bear for running DL algorithms. These factors included measurements of the running

TABLE VI
DNN SIZES AND COMPLEXITIES IN DIFFERENT APPLICATIONS. (NA: NOT AVAILABLE, C: CONVOLUTIONAL LAYER, F: FULLY CONNECTED LAYER, FF: FEEDFORWARD LAYER, SG: SUMMATION (COLLAPSE) LAYER, P: POOLING LAYER, LRN: LOCAL RESPONSE NORMALIZATION LAYER, SM: SOFTMAX LAYER, L: LSTM LAYER, R: RNN LAYER, RBM: RBM HIDDEN LAYER)

Work	Application	Type of DNN	Depth	Layers Sizes	Training Time	Test Time
[79]	Transportation analysis	RNN+RBM	2	R(100)-RBM(150)	NA	354 (s), whole test set
[92]	Localization	RBM DBN	4 4	500-300-150-50 300-150-100-50	NA	NA
[93]	Localization	SdA DBN ML-ELM SDELM	4 3 5 5	26-200-200-71 26-300-71 26-300-300-1500-71 26-300-300-1500-71	451 (s) 110 (s) 14 (s) 24 (s)	NA
[94]	Localization	SdA	5	163-200-200-200-91	NA	0.25 (s)
[98]	Pose detection	CNN	12	C(55×55×96)-LRN-P- C(27×27×256)-LRN-P- C(13×13×384)- C(13×13×384)- C(13×13×256)-P- F(4096)-F(4096)-SM	NA	0.1 (s)
[100]	Human activity detection	CNN+LSTM	7	C(384)-C(20544)-C(20544)- C(20544)-L(942592)- L(33280)-SM	340 (min)	7 (s), whole test set
[101]	Human activity detection	LSTM	5	L(4)-FF(6)-L(10)-SG-SM	NA	2.7 (ms)
[107]	FDI detection	DBN	4	50-50-50-50	NA	1.01 (ms)
[109]	Malware detection	DBN	3	150-150-150	NA	NA
[120]	Parking space	CNN	8	C(64×11×11)-C(256×5×5)- C(256×3×3)-C(256×3×3)- C(256×3×3)-F(4096)-F(2)-SM	NA	0.22 (s)
[126]	Traffic sign detection	CNN	6	C(36×36×20)-P- C(14×14×50)-P- FC(250)-SM	NA	29.6 (ms) on GPU 4264 (ms) on CPU
[128]	food recognition	CNN	22	Used GoogLeNet [75]	NA	50 (s)
[135]	Crop recognition	CNN	6	C(96×7×7)-P- C(96×4×4)-P-F(96)-F(96)	12 (h)	NA
[145]	Classroom Occupancy	CNN	5	C(6×28×28)-P- C(16×10×10)-P-F(120)	2.5 (h)	2 (s) (4 thread)
[146]	Fault diagnosis	AE	4	300-300-300-300	NA	91 (s)
[152]	Road damage detection	CNN	8	Used AlexNet [37]	NA	1.08 (s)
[155]	Classifying offensive plays	RNN	3	10-10-11	NA	10 (ms)

time, energy consumption, and memory footprint. The study focused on investigating the behavior of CNNs and DNNs in three hardware platforms that are used in IoT, mobile, and wearable applications, namely *Snapdragon 800* used in some models of smart phones and tablets, *Intel Edison* used in wearable and form-factor sensitive IoT, and *Nvidia Tegra K1* employed in smart phones as well as IoT-enabled vehicles. Torch has been used for developing and training DNNs, and AlexNet [37] was the dominant model used in these platforms. Their measurement of energy usage indicated that all the platforms, including Intel Edison (which is the weakest one), were able to run the compressed models. In terms of execution time for CNNs, it has been shown that the later convolutional layers tend to consume less time as their dimensions decrease.

Moreover, it is known that feed-forward layers are much faster than the convolutional layers in CNNs. Consequently, a good approach for improving CNN models on resource-constrained devices is to replace convolutional layers with feed-forward layers whenever possible. In addition, choosing the employed activation function in DNNs can have a great effect on time efficiency. For instance, several tests have shown that ReLU functions are more time-efficient followed by Tanh, and then Sigmoid. However, the overall runtime reduction of such selection is not significant (less than 7%) compared to the execution time of layers (at least 25%). In terms of memory usage, CNNs use less storage than DNNs due to the fewer stored parameters in convolutional layers compared to their counterpart in DNNs.

As previously stated, reducing the number of employed parameters in DNNs, by pruning redundant and less-important ones, is another important approach to make DNNs implementable on resource-constrained devices. One of the first works on this approach is Optimal Brain Damage [183] in 1989. At the core of this method, the second order derivative of the parameters are used to compute the importance of the parameters and prune unimportant parameters from the network. The method presented in [184] also works based on pruning redundant and unnecessary connections and neurons as well as using weight sharing mechanisms. Weight sharing replaces each weight with an n bit index from a shared table

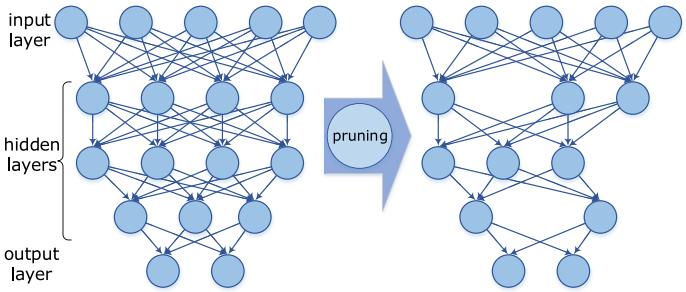


Fig. 19. The overall concept of pruning a DNN.

that has 2^n possible values. The steps to prune the network as describe by Han *et al.* [184] consist of:

- Train the network to find the connections with high weights.
- Prune the unimportant connections that have a weight less than a threshold.
- After pruning, there may remain some neurons with no input nor output connections. The pruning process identifies these neurons and removes them as well as all their remaining connections.
- Retrain the network to fine-tune the weight of the updated model. The weights should be transferred from the previous trained steps instead of initializing them, otherwise the performance will degrade to some extent.

The authors evaluated this approach on four models related to vision, namely AlexNet, VGG-16, LeNet-300-100, and LeNet-5. The models were compressed at least 9 times for AlexNet and 13 times at VGG-16, while the accuracy of the models were almost preserved. One limitation of this approach is that it cannot be used for other types of DNN models. Moreover, the resulting compressed networks are not efficient enough on all hardware platforms and CPU architectures, and thus need new kinds of accelerators that can handle dynamic activation sparsity and weight sharing. Figure 19 illustrates the concept of pruning a DNN.

In [185], an inference engine, called EIE, was designed with a special hardware architecture and SRAMs instead of DRAMs, and was shown to work well with compressed network models. In this architecture, customized sparse matrix vector multiplication and weight sharing are handled without losing the efficiency of the network. The engine consists of a scalable array of processing elements (PEs), each of which keeping a part of the network in an SRAM and performing its corresponding computations. Since most of the energy that is used by neural networks is consumed for accessing the memory, the energy usage is reported to be 120 times fewer with this designed accelerator than the energy consumption of the corresponding original network.

In HashedNets [186], the neural network connection weights are randomly grouped into hash buckets using a hash function. All connections that fall into a same bucket are represented by a single parameter. Backpropagation is used to fine-tune the parameters during the training. Testing results show that the accuracy of this hash-based compressed model outperforms all other compression baseline methods.

The work in [187] by Courbariaux and Bengio proposed to binarize network weights and neurons at both the inference phase and the entire training phase, in order to reduce the memory footprint and accesses. The network can also perform most of the arithmetic operations through bit-wise operations, leading to a decreased power consumption. MNIST, CIFAR-10 and Street View House Numbers (SVHN) [195] datasets were tested over Torch7 and Theano frameworks using this approach, and results were found to be promising.

2) Approximate Computing: Approximate computing is another approach to both implement machine learning tools on IoT devices and contribute to the energy saving of their hosting devices [188], [189]. The validity of this approach arises from the fact that, in many IoT applications, machine learning outputs (e.g., predictions) need not to be exact, but rather to be in an acceptable range providing the desired quality. Indeed, these approaches need to define quality thresholds that the output must not pass. Integrating DL models with approximate computing can lead to more efficient DL models for resource-constrained devices. Venkataramani *et al.* [188] proposed the extension of approximate computing to neural networks, and converted a neural network to an approximate neural network. In their approach, the authors extend back-propagation to identify the neurons with the least effect on output accuracy. Then, the approximate NN is formed by substituting less important neurons in the original network with their approximate counterparts. Making approximate neurons is performed by an approximate design technique called precision scaling. Instead of using a typical fixed number of bits (16-bit or 32-bit format) to present computations, various number of bits (4 - 10 bits) are used in this technique. After forming the approximate network, the precisions of the inputs and weights of neurons are adjusted to come up with an optimal trade-off between accuracy and energy. There are also other attempts that have reported applying approximate computing with precision scaling on CNNs [189] and DBNs [196]. However, the current practice requires that the process of training the model and converting it to approximate DL takes place on a resource rich platform and then the converted model is deployed on a resource-constrained device.

3) Accelerators: Designing specific hardware and circuits is another active research direction aiming to optimize the energy efficiency [191] and memory footprint [192] of DL models in IoT devices. The focus of such research works is on the inference time of DL models, since the training procedure of complex models is time and energy intensive. In [197], several approaches for improving the intelligence of IoT devices are identified including designing accelerators for DNNs, and using Post-CMOS technologies such as spintronics that employs electron spinning mechanism [198]. This latter technology suggests a direction toward the development of hybrid devices that can store data, perform computations and communications within the same material technology.

The works in [190] and [191] have reported an investigation of developing accelerators for DNNs and CNNs, respectively. Beyond the hardware accelerators, the work in [193] proposed the use of a software accelerator for the inference phase of DL models on mobile devices. It employs two

resource control algorithms at run-time, one that compresses layers and the other that decomposes deep architecture models across available processors. This software accelerator can be a complementary solution for the hardware accelerator designs.

4) Tinymotes: In addition to all prior solutions, developing tiny size processors (micromotes) with strong DL capabilities is on the rise [194], [199]. Designed within the range of one cubic millimeter, such processors can be operated by batteries, consuming only about 300 microwatts while performing on-board analysis and prediction using deep network accelerators. By this technology, many time-critical IoT applications can perform decision-making on the device instead of sending data to high performance computers and waiting for their response. For applications where data security and privacy are the main concerns, this integration of hardware and DL alleviates these concerns to some extent, as no or only limited data needs to be sent to the cloud for analysis. Moons and Verhelst [200] also developed a tiny processor for CNNs (total active area of 1.2×2 mm^2) that is power efficient (power consumption is from 25 to 288 mW).

B. Applications

There are existing mobile apps that employ pre-trained DNNs to perform their analytic and prediction tasks, such as using a CNN to identify garbage in images [81]. However, resource consumption on these apps is still very high. Indeed, [81] reports about 5.6 seconds for returning a prediction response, while consuming 83% of the CPU and 67 MB of memory. Howard *et al.* [201] proposed MobileNets architecture for use in mobile and embedded vision applications. By restructuring a complex model through factorizing a standard convolution layer into a depthwise convolution and a 1×1 convolution, they were able to reach a smaller and computationally efficient models for GoogleNet and VGG16. They also demonstrated several use cases of their model including object detection, image classification, and identifying face attributes.

Amato *et al.* [119] run a CNN on Raspberry Pi boards that were incorporated in smart cameras to find out the empty parking slots. Ravi *et al.* [202] have reported the development of a fitness app for mobile devices that uses DL for classifying human activities. The DL model is trained on a standard machine and then transferred to the mobile platform for activity recognition. However, the input of the DL model is mixed with several engineered features to improve the accuracy. As the authors describe, the small number of layers in the DNN models dedicated for a resource-constrained device is a potential reason for them achieving a poor performance. In addition, the performance would not be satisfactorily if the training data is not well representing the entire ecosystem.

Nguyen-Thanh *et al.* [203] proposed a conceptual software-hardware framework to support IoT smart applications. Their framework consists of a cognitive engine and a smart connectivity component. The cognitive engine, which provides cognitive functionality to smart objects, utilizes both DL algorithms and game-theoretic decision analytics. To be suitable for IoT, these algorithms must be deployed on low-power application-specific processors. The smart connectivity component integrates with cognitive radio transceivers and baseband processors to cater flexible and reliable connections to IoT smart objects.

C. Lessons Learned

In this section, the need to move toward supporting DL on IoT embedded and resource constrained devices were discussed. The adversary characteristics of IoT devices and DL techniques make this direction more challenging since IoT devices can rarely host DL models even to only perform predictions due to their resource constraints. To tackle these challenges, several methods were introduced in the recent literature including:

- DNN compression
- Approximate computing for DL
- Accelerators
- Tinymotes with DL.

These approaches focus on the inference functionality of available or pre-trained DL models. So, training DL models on resource-constrained and embedded devices is still an open challenge. Shifting the training process to IoT devices is desired for scalable and distributed deployment of IoT devices. For example, having hundreds of smart security cameras deployed in a community for face-based authentication, the training process for each camera can be done on site. Network compression involves identifying unimportant connections and neurons in a DNN through several rounds of training. While this is a promising approach for getting close to real-time analytics on IoT devices, more investigations need to be performed to handle several challenges such as:

- It is not clear whether network compression approaches are suitable for data streaming, especially when the DL model is dynamic and may evolve over time.
- The compression methods for time-series architectures, such as RNN and LSTM, have not been well investigated, and there is a gap to see if the existing compression methods are applicable to these DL architectures.
- There is a need to specify the trade-off between the rate of compression and accuracy of a DNN, as more compression leads to degraded accuracy.

More recently, approximate computing approaches have also been utilized in making DL models simpler and more energy-efficient, in order to operate them on resource constrained devices. Similar to network compression techniques, these methods also take advantage of insignificant neurons. However, instead of manipulating the network structure, they preserve the structure but change the computation representations through bit-length reduction. For that reason, they seem applicable to a variety of DL architectures and can even cover the dynamic evolution of network models during run-time. Keeping a balance between accuracy and energy usage is their common goal. Nonetheless, more works are needed to find out the superiority of one of these approaches for embedding DL models in IoT devices.

Moreover, we discussed the emergence of special and small form-factor hardware that is designed to efficiently run DL

TABLE VII
METHODS AND TECHNOLOGIES TO BRING DL ON IoT DEVICES

Method / Technology	Reference	Pros	Cons
Network Compression	[184] [186] [187]	• Reduce storage and computation	• Not general for all DL models • Need specific hardware • The pruning process bring overload to training
Approximate Computing	[188], [189]	• Makes fast DL models • Save energy	• Not suitable for precise systems
Accelerators	[91] [185] [190] [191] [192] [193]	• Integrates DL model with the hardware • Efficient computations	• Does not work with the traditional hardware platforms
Tinymote with DL	[194]	• Good for time-critical IoT apps • Energy-efficient • Provides more security and privacy for data	• Special-purpose networks

models on embedded and resource constrained devices. These architectures can be utilized in wearable, mobile, and IoT devices, due to their reduced resource demands and their applicability to time-sensitive IoT applications. However, their generality to support any kind of DNN as well as their interoperability and compatibility with existing hardware platforms remain as clear challenges.

Table VII summarizes the methods and technologies utilized in the recent literature to host DL analytics on IoT devices along with their pros and cons.

We also reviewed some applications that have implemented DL on resource constrained devices. Due to the aforementioned challenges, there are not many well developed applications in this category. However, by resolving these challenges and barriers, we will see the rise of many IoT applications where their core DL model is embedded into the sensors, actuators, and IoT smart objects.

VI. FOG AND CLOUD-CENTRIC DL FOR IOT

Cloud computing is considered a promising solution for IoT big data analytics. However, it may not be ideal for IoT data with security, legal/policy restrictions (e.g., data should not be transferred into cloud centers that are hosted outside of national territory), or time constraints. On the other hand, the high-level abstraction of data for some analytics purposes should be acquired by aggregating several sources of IoT data; hence, it is insufficient to deploy analytic solutions on individual IoT nodes in these cases.

Instead of being only on the cloud, the idea of bringing computing and analytics closer to the end-users/devices has been recently proposed under the name of fog computing. Relying on fog-based analytics, we can benefit from the advantages of cloud computing while reducing/avoiding its drawbacks, such as network latency and security risks. It has been shown that, by hosting data analytics on fog computing nodes, the overall performance can be improved due to the avoidance of transmitting large amounts of raw data to distant cloud nodes [204]. It is also possible to perform real-time analytics to some extent since the fog is hosted locally close to the source of data. Smart application gateways are the core elements in this new fog technology, performing some of the tasks currently done by cloud computing such as data aggregation, classification, integration, and interpretation, thus facilitating the use of IoT local computing resources.

The work in [205] proposed an intelligent IoT gateway that supports mechanisms by which the end users can control the application protocols in order to optimize the performance. The intelligent gateway primarily supports the inter-operation of different types of both IoT and resource-rich devices, causing them to be treated similarly. In the proposed intelligent gateway, a lightweight analytic tool is embedded to increase the performance at the application level. Equipping IoT gateways and edge nodes with efficient DL algorithms can localize many complex analytical tasks that are currently performed on the cloud. Table VIII summarizes several products that have incorporated DL in their intelligent core, and can serve IoT domains in the fog or cloud.

In the following subsection, we review several state-of-the-art enabling technologies that facilitate deep learning on the fog and cloud platforms.

A. Enabling Technologies and Platforms

Despite introducing DL analytics on fog infrastructure, cloud computing remains the only viable solution for analytics in many IoT applications that cannot be handled by fog computing. For example, complex tasks such as video analysis require large and complex models with a lot of computing resources. Thus, designing scalable and high performance cloud-centric DNN models and algorithms, which can perform analytics on massive IoT data, is still an important research area. Coates *et al.* [206] proposed a large-scale system, based on a cluster of GPU servers, which can perform the training of neural networks with 1 billion parameters on 3 machines in few days. The system can be also scaled to train networks with 11 billion parameters on 16 machines.

Project Adam [207] is another attempt to develop a scalable and efficient DL model. The system is based on distributed DL, where the computation and communication of the whole system are optimized for high scalability and efficiency. The evaluation of this system using a cluster of 120 machines shows that training a large DNN with 2 billion connection achieves two times higher accuracy compared to a baseline system, while using 30 times fewer machines.

Google's Tensor Processing Unit (TPU) [208] is a specialized co-processor for DNNs in Google's data centers. It was designed in 2015 with the aim to accelerate the inference phase of DNNs that are written by TensorFlow framework. From 95% of DNN representatives in their data centers, CNNs

TABLE VIII
SOME PRODUCTS THAT USED DEEP LEARNING AND SERVING IOT DOMAINS ON THE FOG OR CLOUD

Product	Description	Application	Platform
Amazon Alexa	Intelligent personal assistant (IPA)	Smart home	Fog
Microsoft Cortana	IPA	Smart Car, XBox	Fog
Google Assistant	IPA	Smart Car, Smart home	Fog
IBM Watson	Cognitive framework	IoT domains	Cloud

only constitute about 5% of the workload, while MLPs and LSTMs cover the other 90%. Performance evaluation showed that TPU outperforms its contemporary GPUs or CPUs on average, by achieving 15 to 30 times faster operation execution, while consuming 30 to 80 times fewer energy per TeraOps/second.

Beyond the infrastructural advancements to host scalable DL models on cloud platforms, there is a need for mechanisms and approaches to make DL models accessible through APIs, in order to be easily integrated into IoT applications. This aspect has not been investigated much, and only a few products are available, such as Amazon's AWS DL AMIs,[6] Google cloud ML,[7] and IBM Watson.[8] This creates opportunities for cloud providers to offer "*DL models as a service*" as a new sub-category of Software as a Service (SaaS). However, this imposes several challenges for cloud providers, since DL tasks are computationally intensive and may starve other cloud services. Moreover, due to the data thirstiness of DL models, data transfers to the cloud may become a bottleneck. In order to deliver DL analytics on the cloud, Figure 20 presents a general stack for DL models as a service. Different providers may use their customized intelligence stack [209], [210].

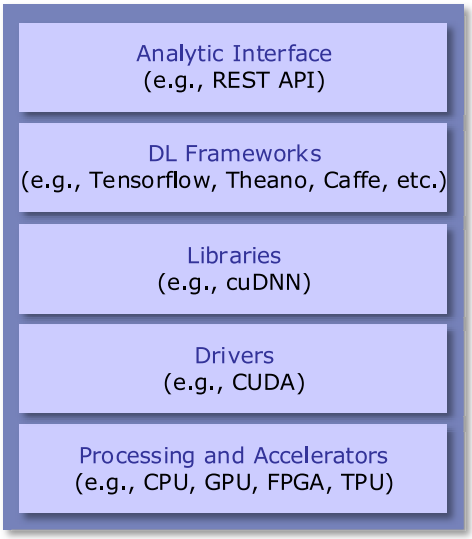


Fig. 20. A general stack of DL models as a service in the cloud platforms.

B. Challenges

When DL analytics come to fog nodes, several challenges need to be addressed, including:

- *DL service discovery:* Fog nodes are densely distributed in geographical regions, and each node may have specific analytics capabilities (e.g., one node runs CNN models for image detection, another node runs RNNs for time-series data prediction, etc.). So, the devices need to identify the sources of appropriate analytic providers through some sort of extended service discovery protocols for DL analytics.
- *DL model and task distribution:* Partitioning the execution of DL models and tasks among the fog nodes, and optimally distributing of the data stream among the available nodes are critical for time-sensitive applications [211]. Aggregating the final results from the computing nodes and returning the action with the least latency are the other side of the coin.
- *Design factors:* Since fog computing environments are in their infancy and are expected to evolve, it is worthwhile to investigate how the design factors of the fog environment (e.g., architectures, resource management, etc.) and the deployment of DL models in this environment can impact the quality of analytic services. Alternatively, it would be also interesting to see how far these design factors can be tailored/extended to improve the operation and quality of DL analytics.
- *Mobile edge:* Through the ubiquity of mobile edge computing environments and their contribution to the IoT analytics, it is important to consider the dynamicity of such environments for designing edge-assisted DL analytics since mobile devices may join and leave the system. Also, the energy management of mobile edge devices should be accurate when analytic tasks are delegated to them.

A few attempts reported the integration of DL on fog nodes in the IoT ecosystems. For example, a proof of concept for deploying CNN models on fog nodes for machine health prognosis was proposed by Qaisar and Usman [212]. In their work, a thorough search among fog nodes is done to find free nodes to delegate analytic tasks to. Also, Li *et al.* [213] proposed a system that leverages the collaboration of mobile and edge devices running CNN models for object recognition.

C. Lessons Learned

In this section, we highlighted the role of cloud and fog computing and their enabling technologies, platforms and challenges to deliver DL analytics to IoT applications. The great success of cloud computing in support of DL is backed by the advancement and employment of optimized processors for neural networks as well as scalable distributed DL algorithms. Deploying DL models on fog platforms for IoT applications, such as smart homes and smart grids, would draw the attention of the end users due to the ease of accessibility and fast response time. Nevertheless, cloud-based DL analytics would

[6]https://aws.amazon.com/amazon-ai/amis/
[7]https://cloud.google.com/products/machine-learning/
[8]https://www.ibm.com/watson/

be of great importance for long-term and complex data analytics that exceed the capabilities of fog computing. Some smart city applications, government sector, and nation-wide IoT deployments need to utilize cloud-based DL infrastructures.

Currently, the integration of DL analytics into IoT applications is limited to RESTful APIs, which are based on the HTTP protocol. While there exist several other application protocols that are extensively used in IoT applications, such as Message Queue Telemetry Transport (MQTT), Constrained Application Protocol (CoAP), Extensible Messaging and Presence Protocol (XMPP), and Advanced Message Queuing Protocol (AMQP), the integration of these protocols with the DL analytic interfaces calls for enhancing their compatibility with the aforementioned protocols to eliminate the need for message conversion proxies, which imposes extra overhead on the analytics response time.

We identified several challenges related to the deployment and usage of DL models in support of analytics on fog nodes. DL service discovery is a necessary requirement due to the dense deployment of fog nodes, which makes brute-force search for available services an inefficient approach. Currently used service discovery protocols in IoT applications, such as multicast DNS (mDNS) or DNS Service Discovery (DNS-SD) [1], need to be extended to support DL service discovery (e.g., declare the type of analytics, DL model, input shape, etc.). Efficient distribution of DL models and tasks, and distribution of data streams on fog nodes and the aggregation of the results are other requirements that need to be addressed.

VII. IoT Challenges for Deep Learning, and Future Directions

In this section we first review several challenges that are important from the machine learning point of view to implement and develop IoT analytics. Then we point out research directions that can fill the existing gaps for IoT analytics based on DL approaches.

A. Challenges

1) Lack of Large IoT Dataset: The lack of availability of large real-world datasets for IoT applications is a main hurdle for incorporating DL models in IoT, as more data is needed for DL to achieve more accuracy. Moreover, more data prevents the overfitting of the models. This shortage is a barrier for deployment and acceptance of IoT analytics based on DL since the empirical validation and evaluation of the system should be shown promising in the natural world. Access to the copyrighted datasets or privacy considerations are another burdens that are more common in domains with human data such as healthcare and education. Also, a portfolio of appropriate datasets would be of a lot of help for developers and researchers. A general list of useful datasets has been compiled in Wikipedia [214]. For the convenience of researchers in machine learning applications in IoT, table IX presents a collection of common datasets suitable to use for DL.

2) Preprocessing: Preparing raw data in an appropriate representation to be fed in DL models is another challenge for IoT applications. Most DL approaches need some sort of preprocessing to yield good results. For example, image processing techniques by CNNs work better when the input data at the pixel level are normalized, scaled into a specific range, or transformed into a standard representation [37], [60]. For IoT applications, preprocessing is more complex since the system deals with data from different sources that may have various formats and distributions while showing missing data.

3) Secure and Privacy Preserving Deep Learning: Ensuring data security and privacy is a main concern in many IoT applications, since IoT big data will be transferred through the Internet for analytics, and can be thus observed around the world. While anonymization is used in many applications, these techniques can be hacked and re-identified as anonymized data. Moreover, DL training models are also subject to malicious attacks, such as False Data Injection or adversarial sample inputs, by which many functional or non-functional (e.g., availability, reliability, validity, trustworthiness, etc.) requirements of the IoT systems may be in jeopardy. Indeed, DL models learn the features from the raw data, and can therefore learn from any invalid data feed to it. In this case, DL models must be enhanced with some mechanism to discover abnormal or invalid data. A data monitoring DL model accompanying the main model should work in such scenarios. McDaniel *et al.* [215] have investigated the vulnerability of DNNs in adversarial settings where an adversary tries to provide some inputs that lead to an incorrect output classification and hence corrupting the integrity of the classification. Developing further techniques to defend and prevent the effect of this sort of attacks on DL models is necessary for reliable IoT applications.

4) Challenges of 6V's: Despite the recent advancement in DL for big data, there are still significant challenges that need to be addressed to mature this technology. Each characteristic of IoT big data imposes a challenge for DL techniques. In the following we highlight these challenges.

The massive volume of data poses a great challenge for DL, especially for time and structure complexity. The voluminous number of input data, their broad number of attributes, and their high degree of classification result in a very complex DL model and affect running time performance. Running DL on distributed frameworks or clusters of CPUs with parallel processing is a viable solution that has been developed [7]. The high volume of IoT big data also brings another challenge, namely the noisy and unlabeled data. Even though DL is very good at tolerating noisy data and learning from unlabeled data, it is not clear to what extent DL models can be accurate in the presence of such abnormal data.

The variety of IoT data formats that come from various sources pops up the challenge of managing conflicts between different data sources. In case of no conflict in data sources, DL has the ability to effectively work on heterogeneous data.

The high velocity of IoT big data, i.e., the high rate of data generation, also brings the challenge of high speed processing and analysis of data. Online learning is a solution for high velocity and has been proposed for DNNs. However, more research is needed to augment DL with online learning and sequential learning techniques.

TABLE IX
COMMON DATA SETS FOR DEEP LEARNING IN IOT

Dataset Name	Domain	Provider	Notes	Address/Link
CGIAR dataset	Agriculture, Climate	CCAFS	High-resolution climate datasets for a variety of fields including agricultural	http://www.ccafs-climate.org/
Educational Process Mining	Education	University of Genova	Recordings of 115 subjects' activities through a logging application while learning with an educational simulator	http://archive.ics.uci.edu/ml/datasets/Educational+Process+Mining+%28EPM%29%3A+A+Learning+Analytics+Data+Set
Commercial Building Energy Dataset	Energy, Smart Building	IIITD	Energy related data set from a commercial building where data is sampled more than once a minute.	http://combed.github.io/
Individual household electric power consumption	Energy, Smart home	EDF R&D, Clamart, France	One-minute sampling rate over a period of almost 4 years	http://archive.ics.uci.edu/ml/datasets/Individual+household+electric+power+consumption
AMPds dataset	Energy, Smart home	S. Makonin	AMPds contains electricity, water, and natural gas measurements at one minute intervals for 2 years of monitoring	http://ampds.org/
UK Domestic Appliance-Level Electricity	Energy, Smart Home	Kelly and Knottenbelt	Power demand from five houses. In each house both the whole-house mains power demand as well as power demand from individual appliances are recorded.	http://www.doc.ic.ac.uk/∼dk3810/data/
PhysioBank databases	Healthcare	PhysioNet	Archive of over 80 physiological datasets.	https://physionet.org/physiobank/database/
Saarbruecken Voice Database	Healthcare	Universität des Saarlandes	A collection of voice recordings from more than 2000 persons for pathological voice detection.	http://www.stimmdatenbank.coli.uni-saarland.de/help_en.php4
T-LESS	Industry	CMP at Czech Technical University	An RGB-D dataset and evaluation methodology for detection and 6D pose estimation of texture-less objects	http://cmp.felk.cvut.cz/t-less/
CityPulse Dataset Collection	Smart City	CityPulse EU FP7 project	Road Traffic Data, Pollution Data, Weather, Parking	http://iot.ee.surrey.ac.uk:8080/datasets.html
Open Data Institute - node Trento	Smart City	Telecom Italia	Weather, Air quality, Electricity, Telecommunication	http://theodi.fbk.eu/openbigdata/
Málaga datasets	Smart City	City of Malaga	A broad range of categories such as energy, ITS, weather, Industry, Sport, etc.	http://datosabiertos.malaga.eu/dataset
Gas sensors for home activity monitoring	Smart home	Univ. of California San Diego	Recordings of 8 gas sensors under three conditions including background, wine and banana presentations.	http://archive.ics.uci.edu/ml/datasets/Gas+sensors+for+home+activity+monitoring
CASAS datasets for activities of daily living	Smart home	Washington State University	Several public datasets related to Activities of Daily Living (ADL) performance in a two-story home, an apartment, and an office settings.	http://ailab.wsu.edu/casas/datasets.html
ARAS Human Activity Dataset	Smart home	Bogazici University	Human activity recognition datasets collected from two real houses with multiple residents during two months.	https://www.cmpe.boun.edu.tr/aras/
MERLSense Data	Smart home, building	Mitsubishi Electric Research Labs	Motion sensor data of residual traces from a network of over 200 sensors for two years, containing over 50 million records.	http://www.merl.com/wmd

(Continued)

The veracity of IoT big data also presents challenges for DL analytics. The IoT big data analytics will not be useful if the input data is not coming from a trustworthy source. Data validation and trustworthiness should be checked at each level of big data analytics, especially when we are dealing with online streams of input data to an analytic engine [216].

Moreover, the variability of IoT big data (variation in the data flow rates) rises challenges for online analytics. In case

TABLE IX
CONTINUED

Dataset Name	Domain	Provider	Notes	Address/Link
SportVU	Sport	Stats LLC	Video of basketball and soccer games captured from 6 cameras.	http:/ /go.stats.com/sportvu
RealDisp	Sport	O. B anos	Includes a wide range of phyiscal activities (warm up, cool down and fitness exercises).	http:/ /orestibanos.com/ datas ets.htm
Taxi Service Trajectory	Transportation	Prediction Challenge, ECML PKDD 2015	Trajectories performed by all the 442 taxis running in the city of Porto, in Portugal.	http://www.geolink.pt/ ecmlpkdd2015-challenge/ dataset.html
GeoLife GPS Trajectories	Transportation	Microsoft	A GPS trajectory by a sequence of time-stamped points	https://www.microsoft.com /en-us/download/details.aspx? id=52367
T-Drive trajectory data	Transportation	Microsoft	Contains a one-week trajectories of 10,357 taxis	https://www.microsoft.com/ en-us/research/publication/ t-drive-trajectory-data-sample/
Chicago Bus Traces data	Transportation	M. Doering	Bus traces from the Chicago Transport Authority for 18 days with a rate between 20 and 40 seconds.	http://www.ibr.cs.tu-bs.de/ users/mdoering/bustraces/
Uber trip data	Transportation	FiveThirty-Eight	About 20 million Uber pickups in New York City during 12 months.	https://github.com/fivethirtyeight/ uber-tlc-foil-response
Traffic Sign Recognition	Transportation	K. Lim	Three datasets: Korean daytime, Korean nighttime, and German daytime traffic signs based on Vienna traffic rules.	https://figshare.com/articles /Traffic_Sign_Recognition_ Testsets/4597795
DDD17	Transportation	J. Binas	End-To-End DAVIS Driving Dataset.	http://sensors.ini.uzh.ch/ databases.html

of immense streams of data, DL techniques, and in particular the online ones, handle them. Data sampling techniques would be beneficial in these scenarios.

Finally, a main challenge for business managers to adopt big data is that it is not clear for them how to use big data analytics to get value out of it and improve their business [217]. Beyond that, the analytic engine may produce abstractions that are not important for the stakeholders, or are not clear enough for them.

5) Deep Learning for IoT Devices: Developing DL on IoT devices poses a new challenge for IoT device designers, to consider the requirements of handling DNNs in resource-constrained devices. These requirements are expected to grow as the datasets sizes are growing every day, and new algorithms arise to be part of the solutions for DL in IoT.

6) Deep Learning Limitations: Despite showing impressive results in many applications, DL models still have several limitations. Nguyen *et al.* [218] reported about the false confidence of DDN for predicting images that are unrecognizable by humans. By producing fooling examples that are totally unrecognizable by humans, the DNN classifies them as familiar objects.

The other limitation is the focus of DL models on classification, while many IoT applications (e.g., electricity load forecasting, temperature forecasting) need a kind of regression at their analytics core. Few works tried to enrich DNNs with regression capabilities, such as the work in [219] proposing the ensemble of DBN and Support Vector Regression (SVR) for regression tasks. However, more investigation is required to clear many aspects of regression with DL.

B. Future Directions

1) IoT Mobile Data: One remarkable part of IoT data comes from mobile devices. Investigating efficient ways to utilize mobile big data in conjunction with DL approaches is a way to come up with better services for IoT domains, especially in smart city scenarios. In [220], the capabilities of DL models in mobile big data analytics were investigated using a distributed learning framework that executes an iterative MapReduce task on several parallel Spark workers.

2) Integrating Contextual Information: The environment's situation cannot be understood by the IoT sensor data alone. Therefore, IoT data needs to be fused with other sources of data, namely context information that complement the understanding of the environment [12]. This integration can also help for fast data analytics and quick reasoning due to the bounded search space for the reasoning engine. For example, a smart camera with capability of face pose recognition can perform its job in various contexts such as security gates in smart homes or government buildings, or in smart cars for driving assistance. In all these situations, complementary contextual information (e.g., time within the day, daily habits, etc.) helps the system to reason about the best action that can be done based on the detected pose of the person.

3) Online Resource Provisioning for IoT Analytics: The deployment of fast DL based data analytics on the fog and cloud would require online provisioning of fog or cloud resources to host the stream of data. Due to the streaming nature of IoT data, knowing the volume of data sequence in advance is not feasible. In this regard, we need a new class of

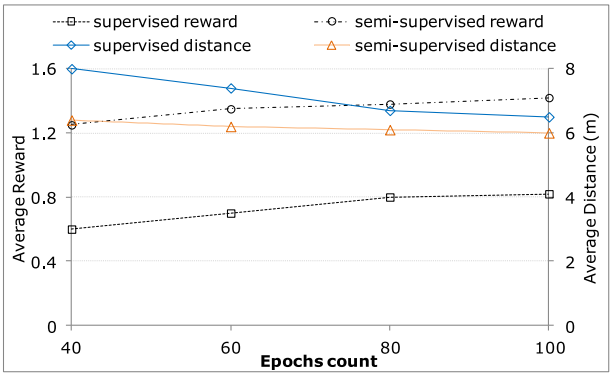


Fig. 21. Deep reinforcement learning with only labeled data (supervised) vs. with labeled and unlabeled data (semisupervised). At each epoch, semi-supervised model outperforms the supervised model both in terms of total received rewards and closeness to the target.

algorithms that work based on the current stream of data and do not rely on the prior knowledge of the data stream. A DL mechanism and an online auctioning algorithm are proposed in [88] and [221], respectively, to support online provisioning of fog and cloud resources for IoT applications.

4) Semi-Supervised Analytic Frameworks: Most of the analytic algorithms are supervised, thus needing a large amount of training labeled data that is either not available or comes at a great expense to prepare. Based on IDC's report [89], it is estimated that by 2012 only about 3% of all data in the digital universe has been annotated, which implies the poor source of training datasets for DL. A combination of advanced machine learning algorithms designed for semi-supervised settings fits well for smart cities systems, where a small training dataset can be used while the learning agent improves its accuracy using a large amount of unlabeled data [5]. Figure 21 illustrates the role of semi-supervised learning in improving the output accuracy for deep reinforcement learning [61] in indoor localization experiments. In their experiments, only 15% of data was labeled but the results were strengthened by utilizing unlabeled data in the algorithm.

5) Dependable and Reliable IoT Analytics: As we rely more on CPS and IoT in large scales, the need for mechanisms to ensure the safety of the system against malicious attacks as well as failures become more crucial [222]. DL approaches can be applied in these directions by analyzing the huge amount of log traces of CPS and IoT systems, in order to identify and predict weak points of the system where attacks may occur or the functionality is defected. This will help the system to prevent or recover from faults and consequently increase the level of dependability of CPS and IoT systems.

6) Self-Organsizing Communication Networks: With a huge number of IoT devices, the configuration and maintenance of their underlying physical M2M communications and networking become harder. Although the large body of network nodes and their relation is a challenge for traditional machine learning approaches, it opens the opportunity for DL architectures to prove their competency in this area by providing a range of self-services such as self-configuration, self-optimization, self-healing, and self-load balancing. Klaine *et al.* [223] have provided a survey of traditional machine learning approaches for self-organizing cellular networks.

7) Emerging IoT Applications (Unmanned Aerial Vehicles): The usage of Unmanned aerial vehicles (UAVs) is a promising application that can improve service delivery in hard-reaching regions or in critical situations. UAVs have been also used for many image analysis in real-time such as surveillance tasks, search-and-rescue operations, and infrastructure inspection [224]. These devices face several challenges for their adoption, including routing, energy saving, avoiding private regions, and obstacle avoidance [225] etc. DL can be of great impact in this domain for the prediction and decision-making of tasks to get the best out of UAVs. Moreover, UAVs can be seen as on-the-fly analytics platforms that potentially can provide temporarily fog computing analytic services as well as distributed analytics.

Virtual/Augmented Reality: Virtual/augmented reality is another application area that can benefit from both IoT and DL. The latter can be used in this field to provide services such as object tracking [226], activity recognition, image classification, and object recognition [227] to name a few. Augmented reality can greatly affect several domains such as education, museums, smart connected cars, etc.

Mobile Robotics: Mobile robots are being used in many commercial and industrial settings for moving materials or performing tasks in hazardous environments. There are many research efforts that have benefited from DL to develop an intelligent control software for mobile robots [228], [229]. Being able to perceive the environment through different kinds of sensors, such as LIDARs and cameras, have made them a top topic to assess the performance of CNN techniques for a variety of vision-based tasks. A strict requirement for mobile robots is that DL models should be able to provide real-time responsiveness.

VIII. Conclusion

DL and IoT have drawn the attention of researchers and commercial verticals in recent years, as these two technology trends have proven to make a positive effect on our lives, cities, and world. IoT and DL constitute a chain of data producer-consumer, in which IoT generates raw data that is analyzed by DL models and DL models produce high-level abstraction and insight that is fed to the IoT systems for fine-tuning and improvement of services.

In this survey, we reviewed the characteristics of IoT data and its challenges for DL methods. In specific, we highlighted IoT fast and streaming data as well as IoT big data as the two main categories of IoT data generation and their requirements for analytics. We also presented several main architectures of DL that is used in the context of IoT applications followed by several open source frameworks for development of DL architectures. Reviewing different applications in various sectors of IoT that have utilized DL was another part of this survey in which we identified five foundational services along with eleven application domains. By distinguishing foundational services, as well as IoT vertical applications, and reviewing

their DL approaches and use cases, the authors provided a basis for other researchers to understand the principle components of IoT smart services and apply the relevant techniques to their problems. The new paradigm of implementing DL on IoT devices was surveyed and several approaches to achieve it were introduced. DL based on fog and cloud infrastructures to support IoT applications was another part of this survey. We also identified the challenges and future research direction in the path of DL for IoT applications.

List of Acronyms

5G	5th Generation (Cellular networks)
AE	Auto-encoder
AI	Artificial Intelligence
AMQP	Advanced Message Queuing Protocol
ANN	Artificial Neural Network
BAC	Breast Arterial Calcification
BLE	Bluetooth Low Energy
BPTT	Backpropagation Through Time
CAE	Contractive Auto-encoder
CDR	Caller Detail Record
CIFAR	Canadian Institute for Advanced Research
CNN	Convolutional Neural Network
CoAP	Constrained Application Protocol
CPS	Cyber-physical System
CRBM	Conditional Restricted Boltzmann Machine
DAE	Denoising Auto-encoder
DBN	Deep Belief Network
DL	Deep Learning
DNN	Deep Neural Network
DNS-SD	DNS Service Discovery
DRL	Deep Reinforcement Learning
ELM	Extreme Learning Machine
FDC	Fault Detection and Classification
FDI	False Data Injection
FNN	Feedforward Neural Network
GAN	Generative Adversarial Network
GBM	Gradient Boosted Machine
GLM	Generalized Linear Model
GPU	Graphics Processing Unit
HMM	Hidden Markov Model
HVAC	Heating, Ventilation and Air Conditioning
INS	Inertia Navigation System
IoT	Internet of Things
IPA	Intelligent Personal Assistant
ITS	Intelligent Transportation System
LSTM	Long Short-term Memory
M2M	Machine-to-Machine
MAPE	Mean Absolute Percentage Error
mDNS	multicast DNS
ML	Machine Learning
MLP	Multi-layer Perceptron
MNIST	Modified National Institute of Standards and Technology
MOOC	Massive Open Online Courses
MQTT	Message Queue Telemetry Transport
RBN	Restricted Boltzmann Machine
ReLU	Rectified Linear Units
RL	Reinforcement Learning
RNN	Recurrent Neural Network
SaaS	Software as a Service
SdA	Stacked denoising Autoencoder
SGD	Stochastic Gradient Descent
SVM	Support Vector Machine
SVR	Support Vector Regression
TPU	Tensor Processing Unit
UAV	Unmanned Aerial Vehicle
VAE	Variational Auto-encoder
VGG	Visual Geometry Group
VLC	Visual Light Communication
WSN	Wireless Sensor Network
XMPP	Extensible Messaging and Presence Protocol.

References

[1] A. Al-Fuqaha, M. Guizani, M. Mohammadi, M. Aledhari, and M. Ayyash, "Internet of Things: A survey on enabling technologies, protocols, and applications," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 2347–2376, 4th Quart., 2015.

[2] J. Manyika *et al.*, *Disruptive Technologies: Advances That Will Transform Life, Business, and the Global Economy*, vol. 180. San Francisco, CA, USA: McKinsey Glob. Inst., 2013.

[3] E. Brynjolfsson and T. Mitchell, "What can machine learning do? workforce implications," *Science*, vol. 358, no. 6370, pp. 1530–1534, 2017.

[4] K. Panetta. (2016). *Gartner's Top 10 Strategic Technology Trends for 2017*. [Online]. Available: http://www.gartner.com/smarterwithgartner/gartners-top-10-technology-trends-2017/

[5] M. Mohammadi and A. Al-Fuqaha, "Enabling cognitive smart cities using big data and machine learning: Approaches and challenges," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 94–101, Feb. 2018.

[6] M. Chen, S. Mao, Y. Zhang, and V. C. Leung, *Big Data: Related Technologies, Challenges and Future Prospects*. Heidelberg, Germany: Springer, 2014.

[7] X.-W. Chen and X. Lin, "Big data deep learning: Challenges and perspectives," *IEEE Access*, vol. 2, pp. 514–525, 2014.

[8] M. Zaharia *et al.*, "Fast and interactive analytics over Hadoop data with spark," *USENIX Login*, vol. 37, no. 4, pp. 45–51, 2012.

[9] C. Engle *et al.*, "Shark: Fast data analysis using coarse-grained distributed memory," in *Proc. ACM SIGMOD Int. Conf. Manag. Data*, ACM, 2012, pp. 689–692.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[11] C.-W. Tsai, C.-F. Lai, M.-C. Chiang, and L. T. Yang, "Data mining for Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 77–97, 1st Quart., 2014.

[12] C. Perera, A. Zaslavsky, P. Christen, and D. Georgakopoulos, "Context aware computing for the Internet of Things: A survey," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 1, pp. 414–454, 1st Quart., 2014.

[13] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, "Machine learning in wireless sensor networks: Algorithms, strategies, and applications," *IEEE Commun. Surveys Tuts.*, vol. 16, no. 4, pp. 1996–2018, 4th Quart., 2014.

[14] Z. Fadlullah *et al.*, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2432–2455, 4th Quart., 2017.

[15] J. Qiu, Q. Wu, G. Ding, Y. Xu, and S. Feng, "A survey of machine learning for big data processing," *EURASIP J. Adv. Signal Process.*, vol. 2016, no. 1, p. 67, 2016.

[16] F. Alam, R. Mehmood, I. Katib, N. N. Albogami, and A. Albeshri, "Data fusion and IoT for smart ubiquitous environments: A survey," *IEEE Access*, vol. 5, pp. 9533–9554, 2017.

[17] B. Li, Y. Diao, and P. Shenoy, "Supporting scalable analytics with latency constraints," *Proc. VLDB Endowment*, vol. 8, no. 11, pp. 1166–1177, 2015.

[18] M. Hilbert, "Big data for development: A review of promises and challenges," *Develop. Policy Rev.*, vol. 34, no. 1, pp. 135–174, 2016.

[19] W. Fan and A. Bifet, "Mining big data: Current status, and forecast to the future," *ACM SIGKDD Explor. Newslett.*, vol. 14, no. 2, pp. 1–5, 2013.

[20] H. Hu, Y. Wen, T.-S. Chua, and X. Li, "Toward scalable systems for big data analytics: A technology tutorial," *IEEE Access*, vol. 2, pp. 652–687, 2014.

[21] Y. Demchenko, P. Grosso, C. De Laat, and P. Membrey, "Addressing big data issues in scientific data infrastructure," in *Proc. Int. Conf. Collaboration Technol. Syst. (CTS)*, IEEE, 2013, pp. 48–55.

[22] J. Ginsberg *et al.*, "Detecting influenza epidemics using search engine query data," *Nature*, vol. 457, no. 7232, pp. 1012–1014, 2009.

[23] M. Strohbach, H. Ziekow, V. Gazis, and N. Akiva, "Towards a big data analytics framework for IoT and smart city applications," in *Modeling and Processing for Next-Generation Big-Data Technologies*. Heidelberg, Germany: Springer, 2015, pp. 257–282.

[24] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, pp. 504–507, 2006.

[25] D. Svozil, V. Kvasnicka, and J. Pospichal, "Introduction to multi-layer feed-forward neural networks," *Chemometrics Intell. Lab. Syst.*, vol. 39, no. 1, pp. 43–62, 1997.

[26] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[27] X. Glorot, A. Bordes, and Y. Bengio, "Deep sparse rectifier neural networks," in *Proc. 14th Int. Conf. Artif. Intell. Stat.*, 2011, pp. 315–323.

[28] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov, "Improving neural networks by preventing co-adaptation of feature detectors," *arXiv preprint arXiv:1207.0580 [cs.NE]*, 2012.

[29] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn.*, 2013, pp. 1139–1147.

[30] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 770–778.

[31] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997.

[32] T. Mikolov, A. Joulin, S. Chopra, M. Mathieu, and M. Ranzato, "Learning longer memory in recurrent neural networks," *arXiv preprint arXiv:1412.7753v2 [cs.NE]*, 2014.

[33] L. Deng, "A tutorial survey of architectures, algorithms, and applications for deep learning," *APSIPA Trans. Signal Inf. Process.*, vol. 3, pp. 1–29, Jan. 2014.

[34] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. COMPSTAT*, Springer, 2010, pp. 177–186.

[35] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[36] Y. Chauvin and D. E. Rumelhart, *Backpropagation: Theory, Architectures, and Applications*. New York, NY, USA: Psychol. Press, 1995.

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

[38] P. J. Werbos, "Backpropagation through time: What it does and how to do it," *Proc. IEEE*, vol. 78, no. 10, pp. 1550–1560, Oct. 1990.

[39] R. Pascanu, C. Gulcehre, K. Cho, and Y. Bengio, "How to construct deep recurrent neural networks," *arXiv preprint arXiv:1312.6026v5 [cs.NE]*, 2013.

[40] M. Hermans and B. Schrauwen, "Training and analysing deep recurrent neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 190–198.

[41] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, "Empirical evaluation of gated recurrent neural networks on sequence modeling," *arXiv preprint arXiv:1412.3555v1 [cs.NE]*, 2014.

[42] P. Baldi, "Autoencoders, unsupervised learning, and deep architectures," in *Proc. ICML Workshop Unsupervised Transf. Learn.*, vol. 27, 2012, pp. 37–50.

[43] C. Doersch, "Tutorial on variational autoencoders," *arXiv preprint arXiv:1606.05908v2 [stat.ML]*, 2016.

[44] D. P. Kingma, S. Mohamed, D. J. Rezende, and M. Welling, "Semi-supervised learning with deep generative models," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3581–3589.

[45] I. Goodfellow *et al.*, "Generative adversarial nets," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 2672–2680.

[46] B. Dai, S. Fidler, R. Urtasun, and D. Lin, "Towards diverse and natural image descriptions via a conditional GAN," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2989–2998.

[47] C. Metz and K. Collins. *How an A.I. 'Cat-and-Mouse Game' Generates Believable Fake Photos*. Accessed: Jan. 2, 2018. [Online]. Available: https://www.nytimes.com/interactive/2018/01/02/technology/ai-generated-photos.html

[48] Y. Bengio *et al.*, "Learning deep architectures for AI," *Found. Trends® Mach. Learn.*, vol. 2, no. 1, pp. 1–127, 2009.

[49] H. Valpola, "From neural PCA to deep unsupervised learning," in *Proc. Adv. Independent Compon. Anal. Learn. Mach.*, 2015, pp. 143–171.

[50] A. Rasmus, M. Berglund, M. Honkala, H. Valpola, and T. Raiko, "Semi-supervised learning with ladder networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 3546–3554.

[51] Y. LeCun, C. Cortes, and C. J. Burges. *The MNIST Database of Handwritten Digits*. Accessed: Feb. 1, 2018. [Online]. Available: http://yann.lecun.com/exdb/mnist/

[52] A. Krizhevsky, "Learning multiple layers of features from tiny images," M.S. thesis, Dept. Comput. Sci., Univ. Toronto, Toronto, ON, Canada, 2009.

[53] N.-Y. Liang, G.-B. Huang, P. Saratchandran, and N. Sundararajan, "A fast and accurate online sequential learning algorithm for feedforward networks," *IEEE Trans. Neural Netw.*, vol. 17, no. 6, pp. 1411–1423, Nov. 2006.

[54] Z. Yang, P. Zhang, and L. Chen, "RFID-enabled indoor positioning method for a real-time manufacturing execution system using OS-ELM," *Neurocomputing*, vol. 174, pp. 121–133, Jan. 2016.

[55] H. Zou, H. Jiang, X. Lu, and L. Xie, "An online sequential extreme learning machine approach to WiFi based indoor positioning," in *Proc. IEEE World Forum Internet Things (WF-IoT)*, IEEE, 2014, pp. 111–116.

[56] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards real-time object detection with region proposal networks," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 6, pp. 1137–1149, Jun. 2017.

[57] R. Girshick, "Fast R-CNN," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1440–1448.

[58] H. Mao *et al.*, "Towards real-time object detection on embedded systems," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[59] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 779–788.

[60] V. Mnih *et al.*, "Playing Atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602v1 [cs.LG]*, 2013.

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

[62] Y. Bengio *et al.*, "Deep learning of representations for unsupervised and transfer learning," in *Proc. ICML Unsupervised Transf. Learn.*, vol. 27, 2012, pp. 17–36.

[63] J. Deng, R. Xia, Z. Zhang, Y. Liu, and B. Schuller, "Introducing shared-hidden-layer autoencoders for transfer learning and their application in acoustic emotion recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy: IEEE, 2014, pp. 4818–4822.

[64] P. Wu *et al.*, "Online multimodal deep similarity learning with application to image retrieval," in *Proc. 21st ACM Int. Conf. Multimedia*, Barcelona, Spain: ACM, 2013, pp. 153–162.

[65] P. Jaini *et al.*, "Online algorithms for sum-product networks with continuous variables," in *Proc. 8th Int. Conf. Probabilistic Graph. Models*, 2016, pp. 228–239.

[66] G. Chen, R. Xu, and S. N. Srihari, "Sequential labeling with online deep learning: Exploring model initialization," in *Proc. Joint Eur. Conf. Mach. Learn. Knowl. Disc. Databases*, Springer, 2016, pp. 772–788.

[67] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative study of deep learning software frameworks," *arXiv preprint arXiv:1511.06435v3 [cs.LG]*, 2016.

[68] A. Candel, V. Parmar, E. LeDell, and A. Arora, *Deep Learning With H2O*, 4th ed. Mountain View, CA, USA: H2O.ai Inc., Oct. 2015.

[69] M. Abadi *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467v2 [cs.DC]*, 2016.

[70] R. Collobert, K. Kavukcuoglu, and C. Farabet, "Torch7: A MATLAB-like environment for machine learning," in *Proc. BigLearn NIPS Workshop*, 2011, pp. 1–6.

[71] F. Bastien *et al.*, "Theano: New features and speed improvements," *arXiv preprint arXiv:1211.5590v1 [cs.SC]*, 2012.

[72] S. Raschka and V. Mirjalili, *Python Machine Learning*, 2nd ed. Birmingham, U.K.: Packt, 2017.

[73] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," in *Proc. 22nd ACM Int. Conf. Multimedia*, Orlando, FL, USA: ACM, 2014, pp. 675–678.

[74] O. M. Parkhi, A. Vedaldi, and A. Zisserman, "Deep face recognition," in *Proc. Brit. Mach. Vis. Conf.*, vol. 1, 2015, p. 6.

[75] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, 2015, pp. 1–9.

[76] S. Shi, Q. Wang, P. Xu, and X. Chu, "Benchmarking state-of-the-art deep learning software tools," *arXiv preprint arXiv:1608.07249v7 [cs.DC]*, 2016.

[77] R. Mehmood *et al.*, "UTiLearn: A personalised ubiquitous teaching and learning system for smart societies," *IEEE Access*, vol. 5, pp. 2615–2635, 2017.

[78] A. Luckow *et al.*, "Deep learning in the automotive industry: Applications and tools," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Washington, DC, USA: IEEE, 2016, pp. 3759–3768.

[79] X. Ma, H. Yu, Y. Wang, and Y. Wang, "Large-scale transportation network congestion evolution prediction using deep learning theory," *PLoS ONE*, vol. 10, no. 3, 2015, Art. no. e0119044.

[80] N. D. Lane, S. Bhattacharya, P. Georgiev, C. Forlivesi, and F. Kawsar, "An early resource characterization of deep learning on wearables, smartphones and Internet-of-Things devices," in *Proc. Int. Workshop Internet Things Towards Appl.*, Seoul, South Korea: ACM, 2015, pp. 7–12.

[81] G. Mittal, K. B. Yagnik, M. Garg, and N. C. Krishnan, "SpotGarbage: Smartphone app to detect garbage using deep learning," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput.*, Heidelberg, Germany: ACM, 2016, pp. 940–945.

[82] C. Liu *et al.*, "A new deep learning-based food recognition system for dietary assessment on an edge computing service infrastructure," *IEEE Trans. Services Comput.*, vol. 11, no. 2, pp. 249–261, Mar./Apr. 2017.

[83] S. Sladojevic, M. Arsenovic, A. Anderla, D. Culibrk, and D. Stefanovic, "Deep neural networks based recognition of plant diseases by leaf image classification," *Comput. Intell. Neurosci.*, vol. 2016, Jun. 2016, Art. no. 6.

[84] Y. Liu *et al.*, "Application of deep convolutional neural networks for detecting extreme weather in climate datasets," *arXiv preprint arXiv:1605.01156 [CS.cv]*, 2016.

[85] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: A next-generation open source framework for deep learning," in *Proc. Workshop Mach. Learn. Syst. (LearningSys) 29th Annu. Conf. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 1–6.

[86] Y. Hada-Muranushi *et al.*, "A deep-learning approach for operation of an automated realtime flare forecast," *arXiv preprint arXiv:1606.01587v1 [astro-ph.SR]*, 2016.

[87] J. A. C. Soto, M. Jentsch, D. Preuveneers, and E. Ilie-Zudor, "CEML: Mixing and moving complex event processing and machine learning to the edge of the network for IoT applications," in *Proc. 6th Int. Conf. Internet Things*, Stuttgart, Germany: ACM, 2016, pp. 103–110.

[88] M. Borkowski, S. Schulte, and C. Hochreiner, "Predicting cloud resource utilization," in *Proc. 9th Int. Conf. Utility Cloud Comput.*, Shanghai, China: ACM, 2016, pp. 37–42.

[89] J. Gantz and D. Reinsel, "The digital universe in 2020: Big data, bigger digital shadows, and biggest growth in the far east," *IDC iView IDC Analyze Future*, vol. 2007, no. 2012, pp. 1–16, 2012.

[90] H. Larry. (2017). *Voice Control Everywhere: Low-Power Special-Purpose Chip Could Make Speech Recognition Ubiquitous in Electronics*. [Online]. Available: http://news.mit.edu/2017/low-power-chip-speech-recognition-electronics-0213

[91] M. Price, J. Glass, and A. P. Chandrakasan, "14.4 A scalable speech recognizer with deep-neural-network acoustic models and voice-activated power gating," in *Proc. IEEE ISSCC*, San Francisco, CA, USA, 2017, pp. 244–245.

[92] X. Wang, L. Gao, S. Mao, and S. Pandey, "DeepFi: Deep learning for indoor fingerprinting using channel state information," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, New Orleans, LA, USA: IEEE, 2015, pp. 1666–1671.

[93] Y. Gu, Y. Chen, J. Liu, and X. Jiang, "Semi-supervised deep extreme learning machine for Wi-Fi based localization," *Neurocomputing*, vol. 166, pp. 282–293, Oct. 2015.

[94] W. Zhang, K. Liu, W. Zhang, Y. Zhang, and J. Gu, "Deep neural networks for wireless localization in indoor and outdoor environments," *Neurocomputing*, vol. 194, pp. 279–287, Jun. 2016.

[95] Z. Liu *et al.*, "Fusion of magnetic and visual sensors for indoor localization: Infrastructure-free and more effective," *IEEE Trans. Multimedia*, vol. 19, no. 4, pp. 874–888, Apr. 2017.

[96] M. Becker, "Indoor positioning solely based on user's sight," in *Proc. Int. Conf. Inf. Sci. Appl.*, Springer, 2017, pp. 76–83.

[97] W. Lu, J. Zhang, X. Zhao, J. Wang, and J. Dang, "Multimodal sensory fusion for soccer robot self-localization based on long short-term memory recurrent neural network," *J. Ambient Intell. Humanized Comput.*, vol. 8, no. 6, pp. 885–893, 2017.

[98] A. Toshev and C. Szegedy, "DeepPose: Human pose estimation via deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Columbus, OH, USA, 2014, pp. 1653–1660.

[99] J. Liu, Y. Gu, and S. Kamijo, "Joint customer pose and orientation estimation using deep neural network from surveillance camera," in *Proc. IEEE Int. Symp. Multimedia (ISM)*, San Jose, CA, USA: IEEE, 2016, pp. 216–221.

[100] F. J. Ordóñez and D. Roggen, "Deep convolutional and LSTM recurrent neural networks for multimodal wearable activity recognition," *Sensors*, vol. 16, no. 1, p. 115, 2016.

[101] D. Tao, Y. Wen, and R. Hong, "Multicolumn bidirectional long short-term memory for mobile devices-based human activity recognition," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1124–1134, Dec. 2016.

[102] X. Li, Y. Zhang, I. Marsic, A. Sarcevic, and R. S. Burd, "Deep learning for RFID-based activity recognition," in *Proc. 14th ACM Conf. Embedded Netw. Sensor Syst.*, Stanford, CA, USA: ACM, 2016, pp. 164–175.

[103] L. Pigou, A. Van Den Oord, S. Dieleman, M. Van Herreweghe, and J. Dambre, "Beyond temporal pooling: Recurrence and temporal convolutions for gesture recognition in video," *Int. J. Comput. Vis.*, vol. 126, nos. 2–4, pp. 430–439, 2018.

[104] K. Fragkiadaki, S. Levine, P. Felsen, and J. Malik, "Recurrent network models for human dynamics," in *Proc. IEEE Int. Conf. Comput. Vis.*, Santiago, Chile, 2015, pp. 4346–4354.

[105] S. E. Kahou *et al.*, "EmoNets: Multimodal deep learning approaches for emotion recognition in video," *J. Multimodal User Interfaces*, vol. 10, no. 2, pp. 99–111, 2016.

[106] N. Neverova *et al.*, "Learning human identity from motion patterns," *IEEE Access*, vol. 4, pp. 1810–1820, 2016.

[107] Y. He, G. J. Mendis, and J. Wei, "Real-time detection of false data injection attacks in smart grid: A deep learning-based intelligent mechanism," *IEEE Trans. Smart Grid*, vol. 8, no. 5, pp. 2505–2516, Sep. 2017.

[108] M.-J. Kang and J.-W. Kang, "Intrusion detection system using deep neural network for in-vehicle network security," *PLoS ONE*, vol. 11, no. 6, 2016, Art. no. e0155781.

[109] Z. Yuan, Y. Lu, Z. Wang, and Y. Xue, "Droid-Sec: Deep learning in android malware detection," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 371–372, 2014.

[110] R. Shokri and V. Shmatikov, "Privacy-preserving deep learning," in *Proc. 22nd ACM SIGSAC Conf. Comput. Commun. Security*, Denver, CO, USA: ACM, 2015, pp. 1310–1321.

[111] M. Abadi *et al.*, "Deep learning with differential privacy," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Vienna, Austria: ACM, 2016, pp. 308–318.

[112] T. J. Hazen. (2016). *Microsoft and Liebherr Collaborating on New Generation of Smart Refrigerators*. [Online]. Available: http://blogs.technet.microsoft.com/machinelearning/2016/09/02/

[113] M. Manic, K. Amarasinghe, J. J. Rodriguez-Andina, and C. Rieger, "Intelligent buildings of the future: Cyberaware, deep learning powered, and human interacting," *IEEE Ind. Electron. Mag.*, vol. 10, no. 4, pp. 32–49, Dec. 2016.

[114] H. Yoshida. (2016). *The Internet of Things That Matter: Integrating Smart Cities With Smart Agriculture. Digitalist Magazine*. [Online]. Available: http://www.digitalistmag.com/iot/2016/05/19/internet-of-things-that-matter-integrating-smart-cities-with-smart-agriculture-04221203

[115] Toshiba. (2016). *Toshiba and Dell Technologies' Deep Learning Testbed for IoT Is First Approved by Industrial Internet Consortium*. [Online]. Available: https://www.toshiba.co.jp/about/press/2016_10/pr1702.htm

[116] X. Song, H. Kanasugi, and R. Shibasaki, "Deeptransport: Prediction and simulation of human mobility and transportation mode at a citywide level," in *Proc. IJCAI*, New York, NY, USA, 2016, pp. 2618–2624.

[117] V. C. Liang *et al.*, "Mercury: Metro density prediction with recurrent neural network on streaming CDR data," in *Proc. IEEE 32nd Int. Conf. Data Eng. (ICDE)*, Helsinki, Finland: IEEE, 2016, pp. 1374–1377.

[118] X. Li, L. Peng, Y. Hu, J. Shao, and T. Chi, "Deep learning architecture for air quality predictions," *Environ. Sci. Pollution Res.*, vol. 23, no. 22, pp. 22408–22417, 2016.

[119] G. Amato *et al.*, "Deep learning for decentralized parking lot occupancy detection," *Exp. Syst. Appl.*, vol. 72, pp. 327–334, Apr. 2017.

[120] S. Valipour, M. Siam, E. Stroulia, and M. Jagersand, "Parking-stall vacancy indicator system, based on deep convolutional neural networks," in *Proc. IEEE 3rd World Forum Internet Things (WF-IoT)*, Reston, VA, USA, 2016, pp. 655–660.

[121] D. C. Mocanu, E. Mocanu, P. H. Nguyen, M. Gibescu, and A. Liotta, "Big IoT data mining for real-time energy disaggregation in buildings," in *Proc. IEEE Int. Conf. Syst. Man Cybern.*, Budapest, Hungary, 2016, pp. 9–12.

[122] E. Mocanu, P. H. Nguyen, M. Gibescu, and W. L. Kling, "Deep learning for estimating building energy consumption," *Sustain. Energy Grids Netw.*, vol. 6, pp. 91–99, Jun. 2016.

[123] A. Gensler, J. Henze, B. Sick, and N. Raabe, "Deep learning for solar power forecasting—An approach using autoencoder and LSTM neural networks," in *Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC)*, Budapest, Hungary: IEEE, 2016, pp. 2858–2865.

[124] Y. Tian and L. Pan, "Predicting short-term traffic flow by long short-term memory recurrent neural network," in *Proc. IEEE Int. Conf. Smart City/SocialCom/SustainCom (SmartCity)*, Chengdu, China: IEEE, 2015, pp. 153–158.

[125] D. Cireşan, U. Meier, J. Masci, and J. Schmidhuber, "Multi-column deep neural network for traffic sign classification," *Neural Netw.*, vol. 32, pp. 333–338, Aug. 2012.

[126] K. Lim, Y. Hong, Y. Choi, and H. Byun, "Real-time traffic sign recognition based on a general purpose GPU and deep-learning," *PLoS ONE*, vol. 12, no. 3, 2017, Art. no. e0173317.

[127] E. Ackerman and A. Pagano, "Deep-learning first: Drive.ai's path to autonomous driving," *IEEE Spectr.*, to be published. Accessed: Mar. 17, 2017. [Online]. Available: https://spectrum.ieee.org/video/transportation/self-driving/driveai-deep-learning-first-autonomous-driving

[128] C. Liu *et al.*, "DeepFood: Deep learning-based food image recognition for computer-aided dietary assessment," in *Proc. Int. Conf. Smart Homes Health Telematics*, Wuhan, China, 2016, pp. 37–48.

[129] C. R. Pereira, D. R. Pereira, J. P. Papa, G. H. Rosa, and X.-S. Yang, "Convolutional neural networks applied for parkinson's disease identification," in *Machine Learning for Health Informatics*. Cham, Switzerland: Springer, 2016, pp. 377–390.

[130] G. Muhammad, S. K. M. M. Rahman, A. Alelaiwi, and A. Alamri, "Smart health solution integrating IoT and cloud: A case study of voice pathology monitoring," *IEEE Commun. Mag.*, vol. 55, no. 1, pp. 69–73, Jan. 2017.

[131] J. Wang *et al.*, "Detecting cardiovascular disease from mammograms with deep learning," *IEEE Trans. Med. Imag.*, vol. 36, no. 5, pp. 1172–1181, May 2017.

[132] P. Feng, M. Yu, S. M. Naqvi, and J. A. Chambers, "Deep learning for posture analysis in fall detection," in *Proc. 19th Int. Conf. Digit. Signal Process. (DSP)*, Hong Kong: IEEE, 2014, pp. 12–17.

[133] Z. C. Lipton, D. C. Kale, C. Elkan, and R. Wetzell, "Learning to diagnose with LSTM recurrent neural networks," in *Proc. ICLR*, 2016, pp. 1–18.

[134] D. Ravì *et al.*, "Deep learning for health informatics," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 1, pp. 4–21, Jan. 2017.

[135] N. Kussul, M. Lavreniuk, S. Skakun, and A. Shelestov, "Deep learning classification of land cover and crop types using remote sensing data," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 5, pp. 778–782, May 2017.

[136] K. Kuwata and R. Shibasaki, "Estimating crop yields with deep learning and remotely sensed data," in *Proc. IEEE Int. Geosci. Remote Sens. Symp. (IGARSS)*, Milan, Italy: IEEE, 2015, pp. 858–861.

[137] G. J. Scott, M. R. England, W. A. Starms, R. A. Marcum, and C. H. Davis, "Training deep convolutional neural networks for land–cover classification of high-resolution imagery," *IEEE Geosci. Remote Sens. Lett.*, vol. 14, no. 4, pp. 549–553, Apr. 2017.

[138] K. A. Steen, P. Christiansen, H. Karstoft, and R. N. Jørgensen, "Using deep learning to challenge safety standard for highly autonomous machines in agriculture," *J. Imag.*, vol. 2, no. 1, p. 6, 2016.

[139] I. Sa *et al.*, "DeepFruits: A fruit detection system using deep neural networks," *Sensors*, vol. 16, no. 8, p. 1222, 2016.

[140] M. B. Ibáñez, Á. Di Serio, D. Villarán, and C. D. Kloos, "Experimenting with electromagnetism using augmented reality: Impact on flow student experience and educational effectiveness," *Comput. Educ.*, vol. 71, pp. 1–13, Feb. 2014.

[141] L.-F. Kwok, "A vision for the development of i-campus," *Smart Learn. Environ.*, vol. 2, pp. 1–12, Dec. 2015.

[142] H. Wang, N. Wang, and D.-Y. Yeung, "Collaborative deep learning for recommender systems," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, Sydney, NSW, Australia: ACM, 2015, pp. 1235–1244.

[143] T.-Y. Yang, C. G. Brinton, C. Joe-Wong, and M. Chiang, "Behavior-based grade prediction for MOOCs via time series neural networks," *IEEE J. Sel. Topics Signal Process.*, vol. 11, no. 5, pp. 716–728, Aug. 2017.

[144] C. Piech *et al.*, "Deep knowledge tracing," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2015, pp. 505–513.

[145] F. Conti, A. Pullini, and L. Benini, "Brain-inspired classroom occupancy monitoring on a low-power mobile platform," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Columbus, OH, USA, 2014, pp. 610–615.

[146] H. Shao, H. Jiang, F. Wang, and H. Zhao, "An enhancement deep feature fusion method for rotating machinery fault diagnosis," *Knowl. Based Syst.*, vol. 119, pp. 200–220, Mar. 2017.

[147] H. Lee, "Framework and development of fault detection classification using IoT device and cloud environment," *J. Manuf. Syst.*, vol. 43, pp. 257–270, Apr. 2017.

[148] H. Lee, Y. Kim, and C. O. Kim, "A deep learning model for robust wafer fault monitoring with sensor measurement noise," *IEEE Trans. Semicond. Manuf.*, vol. 30, no. 1, pp. 23–31, Feb. 2017.

[149] W. Yan and L. Yu, "On accurate and reliable anomaly detection for gas turbine combustors: A deep learning approach," in *Proc. Annu. Conf. Prognostics Health Manag. Soc.*, 2015, p. 8.

[150] Y. Liu and L. Wu, "Geological disaster recognition on optical remote sensing images using deep learning," *Procedia Comput. Sci.*, vol. 91, pp. 566–575, 2016.

[151] Q. Wang, Y. Guo, L. Yu, and P. Li, "Earthquake prediction based on spatio-temporal data mining: An LSTM network approach," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[152] H. Maeda, Y. Sekimoto, and T. Seto, "Lightweight road manager: Smartphone-based automatic determination of road damage status by deep neural network," in *Proc. 5th ACM SIGSPATIAL Int. Workshop Mobile Geograph. Inf. Syst.*, Burlingame, CA, USA: ACM, 2016, pp. 37–45.

[153] L. Steinberg. (2015). *Forbes—Changing the Game: The Rise of Sports Analytics*. [Online]. Available: https://www.forbes.com/sites/leighsteinberg/2015/08/18/changing-the-game-the-rise-of-sports-analytics

[154] W. Liu *et al.*, "Deep learning based intelligent basketball arena with energy image," in *Proc. Int. Conf. Multimedia Model.*, Springer, 2017, pp. 601–613.

[155] K.-C. Wang and R. Zemel, "Classifying NBA offensive plays using neural networks," in *Proc. MIT SLOAN Sports Anal. Conf.*, 2016, pp. 1–9.

[156] R. Shah and R. Romijnders, "Applying deep learning to basketball trajectories," *arXiv preprint arXiv:1608.03793 [cs.NE]*, pp. 1–4, 2016.

[157] T. Kautz *et al.*, "Activity recognition in beach volleyball using a deep convolutional neural network," *Data Min. Knowl. Disc.*, vol. 31, no. 6, pp. 1678–1705, 2017.

[158] M. S. Ibrahim, S. Muralidharan, Z. Deng, A. Vahdat, and G. Mori, "A hierarchical deep temporal model for group activity recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 1971–1980.

[159] S. Bell and K. Bala, "Learning visual similarity for product design with convolutional neural networks," *ACM Trans. Graph.*, vol. 34, no. 4, p. 98, 2015.

[160] L. Xiao and X. Yichao, "Exact clothing retrieval approach based on deep neural network," in *Proc. IEEE Inf. Technol. Netw. Electron. Autom. Control Conf.*, Chongqing, China: IEEE, 2016, pp. 396–400.

[161] M. H. Kiapour, X. Han, S. Lazebnik, A. C. Berg, and T. L. Berg, "Where to buy it: Matching street clothing photos in online shops," in *Proc. IEEE Int. Conf. Comput. Vis.*, Santiago, Chile, 2015, pp. 3343–3351.

[162] S. Advani *et al.*, "A multitask grocery assist system for the visually impaired: Smart glasses, gloves, and shopping carts provide auditory and tactile feedback," *IEEE Consum. Electron. Mag.*, vol. 6, no. 1, pp. 73–81, Jan. 2017.

[163] B. Singh, T. K. Marks, M. Jones, O. Tuzel, and M. Shao, "A multi-stream bi-directional recurrent neural network for fine-grained action detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 1961–1970.

[164] Q. Feng, Y. Zhang, C. Li, Z. Dou, and J. Wang, "Anomaly detection of spectrum in wireless communication via deep auto-encoders," *J. Supercomput.*, vol. 73, no. 7, pp. 3161–3178, 2017.

[165] M. Lopez-Martin, B. Carro, A. Sanchez-Esguevillas, and J. Lloret, "Conditional variational autoencoder for prediction and feature recovery applied to intrusion detection in IoT," *Sensors*, vol. 17, no. 9, p. 1967, 2017.

[166] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.

[167] M. Hasan, E. Hossain, and D. Niyato, "Random access for machine-to-machine communication in LTE-advanced networks: Issues and approaches," *IEEE Commun. Mag.*, vol. 51, no. 6, pp. 86–93, Jun. 2013.

[168] H.-Y. Kim and J.-M. Kim, "A load balancing scheme based on deep-learning in IoT," *Cluster Comput.*, vol. 20, no. 1, pp. 873–878, 2017.

[169] M. Schmidt, D. Block, and U. Meier, "Wireless interference identification with convolutional neural networks," *arXiv preprint arXiv:1703.00737v1 [cs.LG]*, 2017.

[170] N. Ahad, J. Qadir, and N. Ahsan, "Neural networks in wireless networks: Techniques, applications and guidelines," *J. Netw. Comput. Appl.*, vol. 68, pp. 1–27, Jun. 2016.

[171] R. Li *et al.*, "Intelligent 5G: When cellular networks meet artificial intelligence," *IEEE Wireless Commun.*, vol. 24, no. 5, pp. 175–183, Oct. 2017.

[172] M.-R. Fida, A. Lutu, M. K. Marina, and Ö. Alay, "ZipWeave: Towards efficient and reliable measurement based mobile coverage maps," in *Proc. IEEE Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA: IEEE, 2017, pp. 1–9.

[173] OpenSignal. *Global Cell Coverage Maps*. Accessed: Feb. 6, 2018. [Online]. Available: http://opensignal.com/networks

[174] A. Canziani, A. Paszke, and E. Culurciello, "An analysis of deep neural network models for practical applications," *arXiv preprint arXiv:1605.07678v4 [cs.CV]*, 2016.

[175] K. He, X. Zhang, S. Ren, and J. Sun, "Identity mappings in deep residual networks," in *Computer Vision—ECCV 2016*, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds. Cham, Switzerland: Springer Int., 2016, pp. 630–645.

[176] C. Bourguignat. (2014). *Interpretable vs Powerful Predictive Models: Why We Need Them Both*. Accessed: Feb. 15, 2018. [Online]. Available: https://medium.com/@chris_bour/interpretable-vs-powerful-predictive-models-why-we-need-them-both-990340074979

[177] F. Chollet, *Deep Learning With Python*. Shelter Island, NY, USA: Manning, 2018.

[178] V. J. Hellendoorn and P. Devanbu, "Are deep neural networks the best choice for modeling source code?" in *Proc. 11th Joint Meeting Found. Softw. Eng.*, Paderborn, Germany: ACM, 2017, pp. 763–773.

[179] K. Chatfield, K. Simonyan, A. Vedaldi, and A. Zisserman, "Return of the devil in the details: Delving deep into convolutional nets," *arXiv preprint arXiv:1405.3531v4 [cs.CV]*, 2014.

[180] J. Ba and R. Caruana, "Do deep nets really need to be deep?" in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2014, pp. 2654–2662.

[181] T. A. Eriksson, H. Bülow, and A. Leven, "Applying neural networks in optical communication systems: Possible pitfalls," *IEEE Photon. Technol. Lett.*, vol. 29, no. 23, pp. 2091–2094, Dec. 2017.

[182] M. Denil, B. Shakibi, L. Dinh, N. de Freitas, and M. Ranzato, "Predicting parameters in deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 2148–2156.

[183] Y. LeCun *et al.*, "Optimal brain damage," in *Proc. NIPS*, vol. 2, 1989, pp. 598–605.

[184] S. Han, J. Pool, J. Tran, and W. Dally, "Learning both weights and connections for efficient neural network," in *Proc. Adv. Neural Inf. Process. Syst.*, Montreal, QC, Canada, 2015, pp. 1135–1143.

[185] S. Han *et al.*, "EIE: Efficient inference engine on compressed deep neural network," *arXiv preprint arXiv:1602.01528v2 [cs.CV]*, 2016.

[186] W. Chen *et al.*, "Compressing neural networks with the hashing trick," in *Proc. 32nd Int. Conf. Mach. Learn.*, vol. 37. Lille, France: JMLR: W&CP, 2015, pp. 2285–2294.

[187] M. Courbariaux and Y. Bengio, "Binarized neural networks: Training deep neural networks with weights and activations constrained to $+1$ or -1," *arXiv preprint arXiv:1602.02830v3 [cs.LG]*, 2016.

[188] S. Venkataramani, A. Ranjan, K. Roy, and A. Raghunathan, "AxNN: Energy-efficient neuromorphic systems using approximate computing," in *Proc. Int. Symp. Low Power Electron. Design*, ACM, 2014, pp. 27–32.

[189] B. Moons, B. De Brabandere, L. Van Gool, and M. Verhelst, "Energy-efficient ConvNets through approximate computing," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Lake Placid, NY, USA: IEEE, 2016, pp. 1–8.

[190] S. G. Ramasubramanian, R. Venkatesan, M. Sharad, K. Roy, and A. Raghunathan, "SPINDLE: SPINtronic deep learning engine for large-scale neuromorphic computing," in *Proc. ACM Int. Symp. Low Power Electron. Design*, 2014, pp. 15–20.

[191] Y.-H. Chen, T. Krishna, J. S. Emer, and V. Sze, "Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks," *IEEE J. Solid-State Circuits*, vol. 52, no. 1, pp. 127–138, Jan. 2017.

[192] T. Chen *et al.*, "DianNao: A small-footprint high-throughput accelerator for ubiquitous machine-learning," *ACM Sigplan Notices*, vol. 49, no. 4, pp. 269–284, 2014.

[193] N. D. Lane *et al.*, "DeepX: A software accelerator for low-power deep learning inference on mobile devices," in *Proc. 15th ACM/IEEE Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, Vienna, Austria: IEEE, 2016, pp. 1–12.

[194] S. Bang *et al.*, "14.7 A 288μw programmable deep-learning processor with 270KB on-chip weight storage using non-uniform memory hierarchy for mobile intelligence," in *Proc. IEEE Int. Solid-State Circuits Conf. (ISSCC)*, San Francisco, CA, USA: IEEE, 2017, pp. 250–251.

[195] Y. Netzer *et al. The Street View House Numbers (SVHN) Dataset*. Accessed: Feb. 9, 2018. [Online]. Available: http://ufldl.stanford.edu/housenumbers/

[196] X. Xu, S. Das, and K. Kreutz-Delgado, "ApproxDBN: Approximate computing for discriminative deep belief networks," *arXiv preprint arXiv:1704.03993v3 [cs.NE]*, 2017.

[197] S. Venkataramani, K. Roy, and A. Raghunathan, "Efficient embedded learning for IoT devices," in *Proc. 21st Asia South Pac. Design Autom. Conf. (ASP-DAC)*, IEEE, 2016, pp. 308–311.

[198] D. D. Awschalom and M. E. Flatté, "Challenges for semiconductor spintronics," *Nat. Phys.*, vol. 3, no. 3, pp. 153–159, 2007.

[199] K. Bourzac, "Speck-size computers: Now with deep learning [news]," *IEEE Spectr.*, vol. 54, no. 4, pp. 13–15, Mar. 2017.

[200] B. Moons and M. Verhelst, "A 0.3–2.6 TOPS/W precision-scalable processor for real-time large-scale ConvNets," in *Proc. IEEE Symp. VLSI Circuits (VLSI-Circuits)*, Honolulu, HI, USA: IEEE, 2016, pp. 1–2.

[201] A. G. Howard *et al.*, "MobileNets: Efficient convolutional neural networks for mobile vision applications," *arXiv preprint arXiv:1704.04861v1 [cs.CV]*, 2017.

[202] D. Ravi, C. Wong, B. Lo, and G.-Z. Yang, "A deep learning approach to on-node sensor data analytics for mobile or wearable devices," *IEEE J. Biomed. Health Inform.*, vol. 21, no. 1, pp. 56–64, Jan. 2016.

[203] N. Nguyen-Thanh *et al.*, "Cognitive computation and communication: A complement solution to cloud for IoT," in *Proc. Int. Conf. Adv. Technol. Commun. (ATC)*, Hanoi, Vietnam: IEEE, 2016, pp. 222–230.

[204] B. Tang *et al.*, "Incorporating intelligence in fog computing for big data analysis in smart cities," *IEEE Trans. Ind. Informat.*, vol. 13, no. 5, pp. 2140–2150, Oct. 2017.

[205] A. Al-Fuqaha, A. Khreishah, M. Guizani, A. Rayes, and M. Mohammadi, "Toward better horizontal integration among IoT services," *IEEE Commun. Mag.*, vol. 53, no. 9, pp. 72–79, Sep. 2015.

[206] A. Coates *et al.*, "Deep learning with COTS HPC systems," in *Proc. 30th Int. Conf. Mach. Learn.*, vol. 28. Atlanta, GA, USA: JMLR:W&CO, 2013, pp. 1337–1345.

[207] T. M. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project ADAM: Building an efficient and scalable deep learning training system," in *Proc. 11th USENIX Symp. Oper. Syst. Design Implement.*, vol. 14, 2014, pp. 571–582.

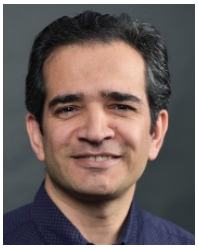
[208] N. P. Jouppi *et al.*, "In-datacenter performance analysis of a tensor processing unit," in *Proc. 44th Int. Symp. Comput. Archit. (ISCA)*, Toronto, ON, Canada, 2017, pp. 1–12.

[209] K. Daniel, "Lessons learned from deploying deep learning at scale," presented at the O'Reilly Artif. Intell. Conf., New York, NY, USA, Sep. 2016. [Online]. Available: http://conferences.oreilly.com/artificial-intelligence/ai-ny-2016/public/schedule/detail/54098

[210] N. Hemsoth, *GPU Platforms Set to Lengthen Deep Learning Reach*, Next Platform, Redwood City, CA, USA, 2015. [Online]. Available: http://www.nextplatform.com/2015/12/07/gpu-platforms-emerge-for-longer-deep-learning-reach/

[211] Y. Simmhan and S. Perera, "Big data analytics platforms for real-time applications in IoT," in *Big Data Analytics*. New Delhi, India: Springer, 2016, pp. 115–135.

[212] S. B. Qaisar and M. Usman, "Fog networking for machine health prognosis: A deep learning perspective," in *Proc. Int. Conf. Comput. Sci. Appl.*, Springer, 2017, pp. 212–219.
[213] D. Li, T. Salonidis, N. V. Desai, and M. C. Chuah, "DeepCham: Collaborative edge-mediated adaptive deep learning for mobile object recognition," in *Proc. IEEE/ACM Symp. Edge Computing (SEC)*, Washington, DC, USA: IEEE, 2016, pp. 64–76.
[214] Wikipedia. (2017). *List of Datasets for Machine Learning Research*. [Online]. Available: https://en.wikipedia.org/wiki/List_of_datasets_for_machine_learning_research
[215] P. McDaniel *et al.*, "The limitations of deep learning in adversarial settings," in *Proc. IEEE Eur. Symp. Security Privacy (EuroS P)*, Saarbrücken, Germany: IEEE, 2016, pp. 372–387.
[216] M. M. Najafabadi *et al.*, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, p. 1, 2015.
[217] S. LaValle, E. Lesser, R. Shockley, M. S. Hopkins, and N. Kruschwitz, "Big data, analytics and the path from insights to value," *MIT Sloan Manag. Rev.*, vol. 52, no. 2, p. 21, 2011.
[218] A. Nguyen, J. Yosinski, and J. Clune, "Deep neural networks are easily fooled: High confidence predictions for unrecognizable images," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 427–436.
[219] X. Qiu, L. Zhang, Y. Ren, P. N. Suganthan, and G. Amaratunga, "Ensemble deep learning for regression and time series forecasting," in *Proc. IEEE Symp. Comput. Intell. Ensemble Learn. (CIEL)*, Orlando, FL, USA: IEEE, 2014, pp. 1–6.
[220] M. A. Alsheikh, D. Niyato, S. Lin, H.-P. Tan, and Z. Han, "Mobile big data analytics using deep learning and apache spark," *IEEE Netw.*, vol. 30, no. 3, pp. 22–29, May/Jun. 2016.
[221] A. Gharaibeh *et al.*, "Online auction of cloud resources in support of the Internet of Things," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1583–1596, Oct. 2017.
[222] National Science Foundation. (2017) *Cyber-Physical Systems (CPS)—Program Solicitation (NSF 17-529)*. [Online]. Available: https://www.nsf.gov/pubs/2017/nsf17529/nsf17529.pdf
[223] P. V. Klaine, M. A. Imran, O. Onireti, and R. D. Souza, "A survey of machine learning techniques applied to self-organizing cellular networks," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2392–2431, 4th Quart., 2017.
[224] J. Lee, J. Wang, D. Crandall, S. Šabanović, and G. Fox, "Real-time, cloud-based object detection for unmanned aerial vehicles," in *Proc. IEEE Int. Conf. Robot. Comput. (IRC)*, Taichung, Taiwan: IEEE, 2017, pp. 36–43.
[225] L. Tai, S. Li, and M. Liu, "A deep-network solution towards model-less obstacle avoidance," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Daejeon, South Korea: IEEE, 2016, pp. 2759–2764.
[226] O. Akgul, H. I. Penekli, and Y. Genc, "Applying deep learning in augmented reality tracking," in *Proc. 12th Int. Conf. Signal Image Technol. Internet Based Syst. (SITIS)*, Naples, Italy: IEEE, 2016, pp. 47–54.
[227] R. E. Sutanto, L. Pribadi, and S. Lee, "3D integral imaging based augmented reality with deep learning implemented by faster R-CNN," in *Proc. Int. Conf. Mobile Wireless Technol.*, Springer, 2017, pp. 241–247.
[228] L. Tai and M. Liu, "Deep-learning in mobile robotics-from perception to control systems: A survey on why and why not," *arXiv preprint arXiv:1612.07139v3 [cs.RO]*, 2016.
[229] R. Goeddel and E. Olson, "Learning semantic place labels from occupancy grids using CNNs," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*, Daejeon, South Korea: IEEE, 2016, pp. 3999–4004.



Mehdi Mohammadi (GS'14) received the B.S. degree in computer engineering from Kharazmi University, Tehran, Iran, in 2003, the M.S. degree in computer engineering (software) from Sheikhbahaee University, Isfahan, Iran, in 2010, and the Ph.D. degree in computer science from Western Michigan University, Kalamazoo, MI, USA. His research interests include Internet of Things, IoT data analytics, machine learning, and cloud computing. He was a recipient of six travel grants from the National Science Foundation. He served as a reviewer for multiple journals, including IEEE INTERNET OF THINGS JOURNAL, the *IEEE Communications Magazine*, the IEEE COMMUNICATIONS LETTERS, the IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTATIONAL INTELLIGENCE, Wiley's *Security and Wireless Communication Networks Journal* and Wiley's *Wireless Communications and Mobile Computing Journal*.

Ala Al-Fuqaha (S'00–M'04–SM'09) received the M.S. degree from the University of Missouri-Columbia and the Ph.D. degree from the University of Missouri-Kansas City. He is currently a Professor and the Director of NEST Research Lab, Computer Science Department, Western Michigan University. His research interests include the use of machine learning in general and deep learning in particular in support of the data-driven and self-driven management of large-scale deployments of IoT and smart city infrastructure and services, wireless vehicular networks, cooperation and spectrum access etiquettes in cognitive radio networks, and management and planning of software defined networks. He served on editorial boards and technical program committees of multiple international journals and conferences. He is an ABET Program Evaluator.

Sameh Sorour (S'98–M'11–SM'16) received the B.Sc. and M.Sc. degrees in electrical engineering from Alexandria University, Egypt, in 2002 and 2006, respectively, and the Ph.D. degree in electrical and computer engineering from the University of Toronto, Canada, in 2011. He was a Post-Doctoral Fellow with the University of Toronto and King Abduallah University of Science and Technology. He joined the King Fahd University of Petroleum and Minerals in 2013. He moved to the University of Idaho in 2016, where he is an Assistant Professor with the Department of Electrical and Computer Engineering. His research interests lie in the broad area of advanced communications/networking/computing/learning technologies for smart cities applications, including cyber physical systems, Internet of Things (IoT) and IoT-enabled systems, cloud and fog networking, network coding, device-to-device networking, autonomous driving and autonomous systems, intelligent transportation systems, and mathematical modeling and optimization for smart systems.

Mohsen Guizani (S'85–M'89–SM'99–F'09) received the B.S. (with Distinction) and M.S. degrees in electrical engineering, the M.S. and Ph.D. degrees in computer engineering from Syracuse University, Syracuse, NY, USA, in 1984, 1986, 1987, and 1990, respectively. He is currently a Professor and the ECE Department Chair with the University of Idaho, USA. He served as the Associate Vice President of Graduate Studies, Qatar University, the Chair of the Computer Science Department, Western Michigan University, and the Chair of the Computer Science Department, University of West Florida. He also served in academic positions with the University of Missouri-Kansas City, University of Colorado-Boulder, Syracuse University, and Kuwait University. He has authored nine books and over 450 publications in refereed journals and conferences. His research interests include wireless communications and mobile computing, computer networks, mobile cloud computing, security, and smart grid. He was a recipient of the Teaching Award multiple times from different institutions as well as the Best Research Award from three institutions. He currently serves on the editorial boards of several international technical journals and the Founder and the Editor-in-Chief of *Wireless Communications* and *Mobile Computing* journal (Wiley). He guest edited a number of special issues in IEEE journals and magazines. He also served as a member, the chair, and the general chair of a number of international conferences. He was the Chair of the IEEE Communications Society Wireless Technical Committee and the Chair of the TAOS Technical Committee. He served as the IEEE Computer Society Distinguished Speaker from 2003 to 2005. He is a Senior Member of ACM.