Deep Learning in Mobile and Wireless Networking: A Survey

Chaoyun Zhang, Paul Patras, *Senior Member, IEEE*, and Hamed Haddadi

***Abstract*—The rapid uptake of mobile devices and the rising popularity of mobile applications and services pose unprecedented demands on mobile and wireless networking infrastructure. Upcoming 5G systems are evolving to support exploding mobile traffic volumes, real-time extraction of fine-grained analytics, and agile management of network resources, so as to maximize user experience. Fulfilling these tasks is challenging, as mobile environments are increasingly complex, heterogeneous, and evolving. One potential solution is to resort to advanced machine learning techniques, in order to help manage the rise in data volumes and algorithm-driven applications. The recent success of deep learning underpins new and powerful tools that tackle problems in this space. In this paper, we bridge the gap between deep learning and mobile and wireless networking research, by presenting a comprehensive survey of the crossovers between the two areas. We first briefly introduce essential background and state-of-the-art in deep learning techniques with potential applications to networking. We then discuss several techniques and platforms that facilitate the efficient deployment of deep learning onto mobile systems. Subsequently, we provide an encyclopedic review of mobile and wireless networking research based on deep learning, which we categorize by different domains. Drawing from our experience, we discuss how to tailor deep learning to mobile environments. We complete this survey by pinpointing current challenges and open future directions for research.**

***Index Terms*—Deep learning, machine learning, mobile networking, wireless networking, mobile big data, 5G systems, network management.**

I. Introduction

INTERNET connected mobile devices are penetrating every aspect of individuals' life, work, and entertainment. The increasing number of smartphones and the emergence of evermore diverse applications trigger a surge in mobile data traffic. Indeed, the latest industry forecasts indicate that the annual worldwide IP traffic consumption will reach 3.3 zettabytes (10^{15} MB) by 2021, with smartphone traffic exceeding PC traffic by the same year [1]. Given the shift in user preference towards wireless connectivity, current mobile infrastructure faces great capacity demands. In response to this increasing demand, early efforts propose to agilely provision resources [2] and tackle mobility management distributively [3]. In the long run, however, Internet Service Providers (ISPs) must develop *intelligent* heterogeneous architectures and tools that can spawn the 5^{th} generation of mobile systems (5G) and gradually meet more stringent end-user application requirements [4], [5].

The growing diversity and complexity of mobile network architectures has made monitoring and managing the multitude of network elements intractable. Therefore, embedding versatile machine intelligence into future mobile networks is drawing unparalleled research interest [6], [7]. This trend is reflected in machine learning (ML) based solutions to problems ranging from radio access technology (RAT) selection [8] to malware detection [9], as well as the development of networked systems that support machine learning practices (e.g., [10] and [11]). ML enables systematic mining of valuable information from traffic data and automatically uncover correlations that would otherwise have been too complex to extract by human experts [12]. As the flagship of machine learning, deep learning has achieved remarkable performance in areas such as computer vision [13] and natural language processing (NLP) [14]. Networking researchers are also beginning to recognize the power and importance of deep learning, and are exploring its potential to solve problems specific to the mobile networking domain [15], [16].

Embedding deep learning into the 5G mobile and wireless networks is well justified. In particular, data generated by mobile environments are increasingly heterogeneous, as these are usually collected from various sources, have different formats, and exhibit complex correlations [17]. As a consequence, a range of specific problems become too difficult or impractical for traditional machine learning tools (e.g., shallow neural networks). This is because *(i)* their performance does not improve if provided with more data [18] and *(ii)* they cannot handle highly dimensional state/action spaces in control problems [19]. In contrast, big data fuels the performance of deep learning, as it eliminates domain expertise and instead employs hierarchical feature extraction. In essence this means information can be distilled efficiently and increasingly abstract correlations can be obtained from the data, while reducing the pre-processing effort. Graphics Processing Unit (GPU)-based parallel computing further enables deep learning to make inferences within milliseconds. This facilitates network analysis and management with high accuracy and in a timely manner, overcoming the run-time limitations of traditional mathematical

Manuscript received March 12, 2018; revised September 16, 2018 and January 29, 2019; accepted March 8, 2019. Date of publication March 13, 2019; date of current version August 20, 2019. *(Corresponding author: Chaoyun Zhang.)*

C. Zhang and P. Patras are with the Institute for Computing Systems Architecture, School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, U.K. (e-mail: chaoyun.zhang@ed.ac.uk; paul.patras@ed.ac.uk).

H. Haddadi is with the Dyson School of Design Engineering, Imperial College London, London SW7 2AZ, U.K. (e-mail: h.haddadi@imperial.ac.uk).

Digital Object Identifier 10.1109/COMST.2019.2904897

techniques (e.g., convex optimization, game theory, meta heuristics).

Despite growing interest in deep learning in the mobile networking domain, existing contributions are scattered across different research areas and a comprehensive survey is lacking. This article fills this gap between deep learning and mobile and wireless networking, by presenting an up-to-date survey of research that lies at the intersection between these two fields. Beyond reviewing the most relevant literature, we discuss the key pros and cons of various deep learning architectures, and outline deep learning model selection strategies, in view of solving mobile networking problems. We further investigate methods that tailor deep learning to individual mobile networking tasks, to achieve the best performance in complex environments. We wrap up this paper by pinpointing future research directions and important problems that remain unsolved and are worth pursing with deep neural networks. Our ultimate goal is to provide a definite guide for networking researchers and practitioners, who intend to employ deep learning to solve problems of interest.

Survey Organization: We structure this article in a top-down manner, as shown in Figure 1. We begin by discussing work that gives a high-level overview of deep learning, future mobile networks, and networking applications built using deep learning, which help define the scope and contributions of this paper (Section II). Since deep learning techniques are relatively new in the mobile networking community, we provide a basic deep learning background in Section III, highlighting immediate advantages in addressing mobile networking problems. There exist many factors that enable implementing deep learning for mobile networking applications (including dedicated deep learning libraries, optimization algorithms, etc.). We discuss these enablers in Section IV, aiming to help mobile network researchers and engineers in choosing the right software and hardware platforms for their deep learning deployments.

In Section V, we introduce and compare state-of-the-art deep learning models and provide guidelines for model selection toward solving networking problems. In Section VI we review recent deep learning applications to mobile and wireless networking, which we group by different scenarios ranging from mobile traffic analytics to security, and emerging applications. We then discuss how to tailor deep learning models to mobile networking problems (Section VII) and conclude this article with a brief discussion of open challenges, with a view to future research directions (Section VIII).[1]

II. Related High-Level Articles and the Scope of This Survey

Mobile networking and deep learning problems have been researched mostly independently. Only recently crossovers between the two areas have emerged. Several notable works paint a comprehensives picture of the deep learning and/or mobile networking research landscape. We categorize these works into *(i)* pure overviews of deep learning techniques, *(ii)* reviews of analyses and management techniques in modern mobile networks, and *(iii)* reviews of works at the intersection between deep learning and computer networking. We summarize these earlier efforts in Table II and in this section discuss the most representative publications in each class.

[1]We list the abbreviations used throughout this paper in Table I.

TABLE I
List of Abbreviations in Alphabetical Order

Acronym	Explanation
5G	5th Generation mobile networks
A3C	Asynchronous Advantage Actor-Critic
AdaNet	Adaptive learning of neural Network
AE	Auto-Encoder
AI	Artificial Intelligence
AMP	Approximate Message Passing
ANN	Artificial Neural Network
ASR	Automatic Speech Recognition
BSC	Base Station Controller
BP	Back-Propagation
CDR	Call Detail Record
CNN or ConvNet	Convolutional Neural Network
ConvLSTM	Convolutional Long Short-Term Memory
CPU	Central Processing Unit
CSI	Channel State Information
CUDA	Compute Unified Device Architecture
cuDNN	CUDA Deep Neural Network library
D2D	Device to Device communication
DAE	Denoising Auto-Encoder
DBN	Deep Belief Network
OFDM	Orthogonal Frequency-Division Multiplexing
DPPO	Distributed Proximal Policy Optimization
DQN	Deep Q-Network
DRL	Deep Reinforcement Learning
DT	Decision Tree
ELM	Extreme Learning Machine
GAN	Generative Adversarial Network
GP	Gaussian Process
GPS	Global Positioning System
GPU	Graphics Processing Unit
GRU	Gate Recurrent Unit
HMM	Hidden Markov Model
HTTP	HyperText Transfer Protocol
IDS	Intrusion Detection System
IoT	Internet of Things
IoV	Internet of Vehicle
ISP	Internet Service Provider
LAN	Local Area Network
LTE	Long-Term Evolution
LSTM	Long Short-Term Memory
LSVRC	Large Scale Visual Recognition Challenge
MAC	Media Access Control
MDP	Markov Decision Process
MEC	Mobile Edge Computing
ML	Machine Learning
MLP	Multilayer Perceptron
MIMO	Multi-Input Multi-Output
MTSR	Mobile Traffic Super-Resolution
NFL	No Free Lunch theorem
NLP	Natural Language Processing
NMT	Neural Machine Translation
NPU	Neural Processing Unit
PCA	Principal Components Analysis
PIR	Passive Infra-Red
QoE	Quality of Experience
RBM	Restricted Boltzmann Machine
ReLU	Rectified Linear Unit
RFID	Radio Frequency Identification
RNC	Radio Network Controller
RNN	Recurrent Neural Network
SARSA	State-Action-Reward-State-Action
SELU	Scaled Exponential Linear Unit
SGD	Stochastic Gradient Descent
SON	Self-Organising Network
SNR	Signal-to-Noise Ratio
SVM	Support Vector Machine
TPU	Tensor Processing Unit
VAE	Variational Auto-Encoder
VR	Virtual Reality
WGAN	Wasserstein Generative Adversarial Network
WSN	Wireless Sensor Network

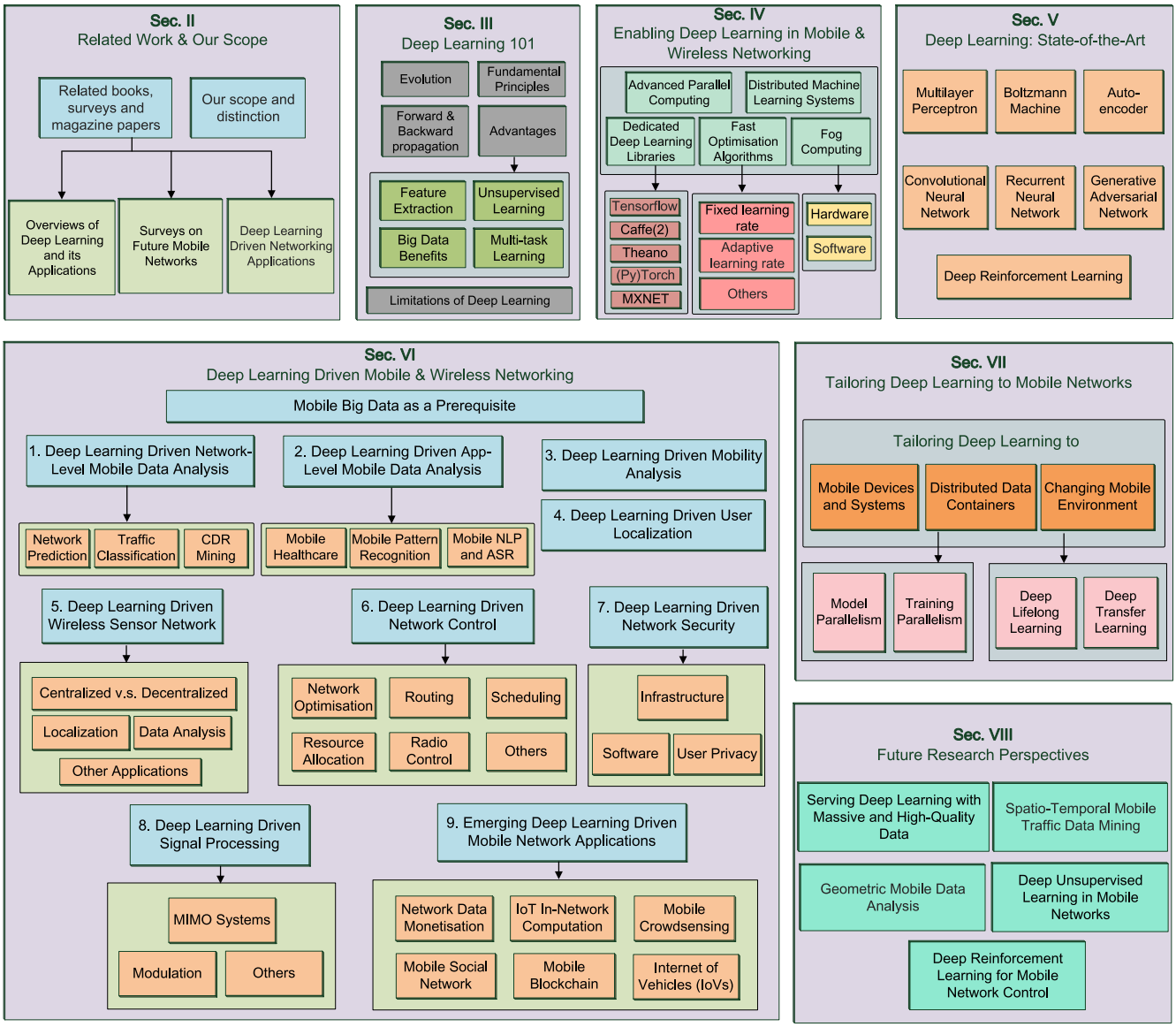


Fig. 1. Diagramatic view of the organization of this survey.

A. Overviews of Deep Learning and Its Applications

The era of big data is triggering wide interest in deep learning across different research disciplines [28]–[31] and a growing number of surveys and tutorials are emerging (e.g., [23] and [24]). LeCun *et al.* [20] give a milestone overview of deep learning, introduce several popular models, and look ahead at the potential of deep neural networks. Schmidhuber [21] undertakes an encyclopedic survey of deep learning, likely the most comprehensive thus far, covering the evolution, methods, applications, and open research issues. Liu *et al.* [22] summarize the underlying principles of several deep learning models, and review deep learning developments in selected applications, such as speech processing, pattern recognition, and computer vision.

Arulkumaran *et al.* [26] present several architectures and core algorithms for deep reinforcement learning, including deep Q-networks, trust region policy optimization, and asynchronous advantage actor-critic. Their survey highlights the remarkable performance of deep neural networks in different control problem (e.g., video gaming, Go board game play, etc.). Similarly, deep reinforcement learning has also been surveyed in [77], where Li shed more light on applications. Zhang *et al.* survey developments in deep learning for recommender systems [32], which have potential to play an important role in mobile advertising. As deep learning becomes increasingly popular, Goodfellow *et al.* [18] provide a comprehensive tutorial of deep learning in a book that covers prerequisite knowledge, underlying principles, and popular applications.

B. Surveys on Future Mobile Networks

The emerging 5G mobile networks incorporate a host of new techniques to overcome the performance limitations of current deployments and meet new application requirements. Progress to date in this space has been summarized through surveys, tutorials, and magazine papers

TABLE II
SUMMARY OF EXISTING SURVEYS, MAGAZINE PAPERS, AND BOOKS RELATED TO DEEP LEARNING AND MOBILE NETWORKING. THE SYMBOL ✓ INDICATES A PUBLICATION IS IN THE SCOPE OF A DOMAIN; ✗ MARKS PAPERS THAT DO NOT DIRECTLY COVER THAT AREA, BUT FROM WHICH READERS MAY RETRIEVE SOME RELATED INSIGHTS. PUBLICATIONS RELATED TO BOTH DEEP LEARNING AND MOBILE NETWORKS ARE SHADED

Publication	One-sentence summary	Scope			
		Machine learning		Mobile networking	
		Deep learning	Other ML methods	Mobile big data	5G technology
LeCun *et al.* [20]	A milestone overview of deep learning.	✓			
Schmidhuber [21]	A comprehensive deep learning survey.	✓			
Liu *et al.* [22]	A survey on deep learning and its applications.	✓			
Deng *et al.* [23]	An overview of deep learning methods and applications.	✓			
Deng [24]	A tutorial on deep learning.	✓			
Goodfellow *et al.* [18]	An essential deep learning textbook.	✓	✗		
Pouyanfar *et al.* [25]	A recent survey on deep learning.	✓	✓		
Arulkumaran *et al.* [26]	A survey of deep reinforcement learning.	✓	✗		
Hussein *et al.* [27]	A survey of imitation learning.	✓	✓		
Chen *et al.* [28]	An introduction to deep learning for big data.	✓	✗	✗	
Najafabadi [29]	An overview of deep learning applications for big data analytics.	✓	✗	✗	
Hordri *et al.* [30]	A brief of survey of deep learning for big data applications.	✓	✗	✗	
Gheisari *et al.* [31]	A high-level literature review on deep learning for big data analytics.	✓		✗	
Zhang *et al.* [32]	A survey and outlook of deep learning for recommender systems.	✓	✗	✗	
Yu *et al.* [33]	A survey on networking big data.			✓	
Alsheikh *et al.* [34]	A survey on machine learning in wireless sensor networks.		✓	✓	
Tsai *et al.* [35]	A survey on data mining in IoT.		✓	✓	
Cheng *et al.* [36]	An introductions mobile big data its applications.			✓	✗
Bkassiny *et al.* [37]	A survey on machine learning in cognitive radios.		✓	✗	✗
Andrews *et al.* [38]	An introduction and outlook of 5G networks.				✓
Gupta *et al.* [5]	A survey of 5G architecture and technologies.				✓
Agiwal *et al.* [4]	A survey of 5G mobile networking techniques.				✓
Panwar *et al.* [39]	A survey of 5G networks features, research progress and open issues.				✓
Elijah *et al.* [40]	A survey of 5G MIMO systems.				✓
Buzzi *et al.* [41]	A survey of 5G energy-efficient techniques.				✓
Peng *et al.* [42]	An overview of radio access networks in 5G.			✗	✓
Niu *et al.* [43]	A survey of 5G millimeter wave communications.				✓
Wang *et al.* [2]	5G backhauling techniques and radio resource management.				✓
Giust *et al.* [3]	An overview of 5G distributed mobility management.				✓
Foukas *et al.* [44]	A survey and insights on network slicing in 5G.				✓
Taleb *et al.* [45]	A survey on 5G edge architecture and orchestration.				✓
Mach and Becvar [46]	A survey on MEC.				✓
Mao *et al.* [47]	A survey on mobile edge computing.			✓	✓
Wang *et al.* [48]	An architecture for personalized QoE management in 5G.			✓	✓
Han *et al.* [49]	Insights to mobile cloud sensing, big data, and 5G.			✓	✓
Singh *et al.* [50]	A survey on social networks over 5G.		✗	✓	✓
Chen *et al.* [51]	An introduction to 5G cognitive systems for healthcare.	✗	✗	✗	✓
Chen *et al.* [52]	Machine learning for traffic offloading in cellular network		✓		✓
Wu *et al.* [53]	Big data toward green cellular networks		✓	✓	✓
Buda *et al.* [54]	Machine learning aided use cases and scenarios in 5G.		✓	✓	✓
Imran *et al.* [55]	An introductions to big data analysis for self-organizing networks (SON) in 5G.		✓	✓	✓
Keshavamurthy *et al.* [56]	Machine learning perspectives on SON in 5G.		✓	✓	✓
Klaine *et al.* [57]	A survey of machine learning applications in SON.	✗	✓	✓	✓
Jiang *et al.* [7]	Machine learning paradigms for 5G.	✗	✓	✓	✓
Li *et al.* [58]	Insights into intelligent 5G.	✗	✓	✓	✓
Bui *et al.* [59]	A survey of future mobile networks analysis and optimization.	✗	✓	✓	✓
Atat *et al.* [60]	A survey of big data application in cyber-physical systems.	✗	✓	✓	✓
Cheng *et al.* [61]	A tutorial of mobile big data analysis	✓	✓	✓	
Kasnesis *et al.* [62]	Insights into employing deep learning for mobile data analysis.	✓		✓	
Alsheikh *et al.* [17]	Applying deep learning and Apache Spark for mobile data analytics.	✓		✓	
Wang and Jones [63]	A survey of deep learning-driven network intrusion detection.	✓	✓	✓	✗
Kato *et al.* [64]	Proof-of-concept deep learning for network traffic control.	✓			✓
Zorzi *et al.* [65]	An introduction to machine learning driven network optimization.	✓	✓		✓
Fadlullah *et al.* [66]	A comprehensive survey of deep learning for network traffic control.	✓	✓	✓	✗
Zheng *et al.* [6]	An introduction to big data-driven 5G optimization.	✓	✓	✓	✓
Mohammadi *et al.* [67]	A survey of deep learning in IoT data analytics.	✓	✓	✓	

(e.g., [4], [5], [38], [39], and [47]). Andrews *et al.* [38] highlight the differences between 5G and prior mobile network architectures, conduct a comprehensive review of 5G techniques, and discuss research challenges facing future developments. Agiwal *et al.* [4] review new architectures for 5G networks, survey emerging wireless technologies,

TABLE III
CONTINUED FROM TABLE II

Publication	One-sentence summary	Scope			
		Machine learning		Mobile networking	
		Deep learning	Other ML methods	Mobile big data	5G technology
Ahad *et al.* [68]	A survey of neural networks in wireless networks.	✓	✗	✗	✓
Mao *et al.* [69]	A survey of deep learning for wireless networks.	✓	✓	✓	
Luong *et al.* [70]	A survey of deep reinforcement learning for networking.	✓	✓		✓
Zhou *et al.* [71]	A survey of ML and cognitive wireless communications.	✓	✓	✓	✓
Chen *et al.* [72]	A tutorial on neural networks for wireless networks.	✓	✓	✓	✓
Gharaibeh *et al.* [73]	A survey of smart cities.	✓	✓	✓	✓
Lane *et al.* [74]	An overview and introduction of deep learning-driven mobile sensing.	✓	✓	✓	
Ota *et al.* [75]	A survey of deep learning for mobile multimedia.	✓		✓	✓
Mishra *et al.* [76]	A survey machine learning driven intrusion detection.	✓	✓	✓	✓
Our work	**A comprehensive survey of deep learning for mobile and wireless network.**	✓	✓	✓	✓

and point out research problems that remain unsolved. Gupta and Jha [5] also review existing work on 5G cellular network architectures, subsequently proposing a framework that incorporates networking ingredients such as Device-to-Device (D2D) communication, small cells, cloud computing, and the IoT.

Intelligent mobile networking is becoming a popular research area and related work has been reviewed in [7], [34], [37], [54], and [56]–[59]. Jiang *et al.* [7] discuss the potential of applying machine learning to 5G network applications including massive MIMO and smart grids. This work further identifies several research gaps between ML and 5G that remain unexplored. Li *et al.* [58] discuss opportunities and challenges of incorporating artificial intelligence (AI) into future network architectures and highlight the significance of AI in the 5G era. Klaine *et al.* [57] present several successful ML practices in Self-Organizing Networks (SONs), discuss the pros and cons of different algorithms, and identify future research directions in this area. Potential exists to apply AI and exploit big data for energy efficiency purposes [53]. Chen *et al.* [52] survey traffic offloading approaches in wireless networks, and propose a novel reinforcement learning based solution. This opens a new research direction toward embedding machine learning towards greening cellular networks.

C. Deep Learning Driven Networking Applications

A growing number of papers survey recent works that bring deep learning into the computer networking domain. Alsheikh *et al.* [17] identify benefits and challenges of using big data for mobile analytics and propose a Spark based deep learning framework for this purpose. Wang and Jones [63] discuss evaluation criteria, data streaming and deep learning practices for network intrusion detection, pointing out research challenges inherent to such applications. Zheng *et al.* [6] put forward a big data-driven mobile network optimization framework in 5G networks, to enhance QoE performance. More recently, Fadlullah *et al.* [66] deliver a survey on the progress of deep learning in a board range of areas, highlighting its potential application to network traffic control systems. Their work also highlights several unsolved research issues worthy of future study.

Ahad *et al.* [68] introduce techniques, applications, and guidelines on applying neural networks to wireless networking problems. Despite several limitations of neural networks identified, this article focuses largely on old neural networks models, ignoring recent progress in deep learning and successful applications in current mobile networks. Lane and Georgiev [74] investigate the suitability and benefits of employing deep learning in mobile sensing, and emphasize on the potential for accurate inference on mobile devices. Ota *et al.* report novel deep learning applications in mobile multimedia. Their survey covers state-of-the-art deep learning practices in mobile health and wellbeing, mobile security, mobile ambient intelligence, language translation, and speech recognition. Mohammadi *et al.* [67] survey recent deep learning techniques for Internet of Things (IoT) data analytics. They overview comprehensively existing efforts that incorporate deep learning into the IoT domain and shed light on current research challenges and future directions. Mao *et al.* [69] focus on deep learning in wireless networking. Their work surveys state-of-the-art deep learning applications in wireless networks, and discusses research challenges to be solved in the future.

D. Our Scope

The objective of this survey is to provide a comprehensive view on state-of-the-art deep learning practices in the mobile networking area. By this we aim to answer the following key questions:

1) Why is deep learning promising for solving mobile networking problems?
2) What are the cutting-edge deep learning models relevant to mobile and wireless networking?
3) What are the most recent successful deep learning applications in the mobile networking domain?
4) How can researchers tailor deep learning to specific mobile networking problems?
5) Which are the most important and promising directions worthy of further study?

The research papers and books we mentioned previously only partially answer these questions. This article goes beyond these previous works and specifically focuses on the crossovers between deep learning and mobile networking. We cover a

range of neural network (NN) structures that are increasingly important and have not been explicitly discussed in earlier tutorials, e.g., [78]. This includes auto-encoders and Generative Adversarial Networks. Unlike such existing tutorials, we also review open-source libraries for deploying and training neural networks, a range of optimization algorithms, and the parallelization of neural networks models and training across large numbers of mobile devices. We also review applications not looked at in other related surveys, including traffic/user analytics, security and privacy, mobile health, etc.

While our main scope remains the mobile networking domain, for completeness we also discuss deep learning applications to wireless networks, and identify emerging application domains intimately connected to these areas. We differentiate between mobile networking, which refers to scenarios where devices are portable, battery powered, potentially wearable, and routinely connected to cellular infrastructure, and wireless networking, where devices are mostly fixed, and part of a distributed infrastructure (including WLANs and WSNs), and serve a single application. Overall, our paper distinguishes itself from earlier surveys from the following perspectives:

- *(i)* We particularly focus on deep learning applications for mobile network analysis and management, instead of broadly discussing deep learning methods (as, e.g., in [20] and [21]) or centering on a single application domain, e.g., mobile big data analysis with a specific platform [17].
- *(ii)* We discuss cutting-edge deep learning techniques from the perspective of mobile networks (e.g., [79] and [80]), focusing on their applicability to this area, whilst giving less attention to conventional deep learning models that may be out-of-date.
- *(iii)* We analyze similarities between existing non-networking problems and those specific to mobile networks; based on this analysis we provide insights into both best deep learning architecture selection strategies and adaptation approaches, so as to exploit the characteristics of mobile networks for analysis and management tasks.

To the best of our knowledge, this is the first time that mobile network analysis and management are jointly reviewed from a deep learning angle. We also provide for the first time insights into how to tailor deep learning to mobile networking problems.

III. Deep Learning 101

We begin with a brief introduction to deep learning, highlighting the basic principles behind computation techniques in this field, as well as key advantages that lead to their success. Deep learning is essentially a sub-branch of ML, which essentially enables an algorithm to make predictions, classifications, or decisions based on data, without being explicitly programmed. Classic examples include linear regression, the k-nearest neighbors classifier, and Q-learning. In contrast to traditional ML tools that rely heavily on features defined by domain experts, deep learning algorithms hierarchically extract knowledge from raw data through multiple layers of nonlinear processing units, in order to make predictions or take actions according to some target objective. The most well-known deep learning models are neural networks (NNs), but only NNs that have a sufficient number of hidden layers (usually more than one) can be regarded as 'deep' models. Besides deep NNs, other architectures have multiple layers, such as deep Gaussian processes [81], neural processes [82], and deep random forests [83], and can also be regarded as deep learning structures. The major benefit of deep learning over traditional ML is thus the automatic feature extraction, by which expensive hand-crafted feature engineering can be circumvented. We illustrate the relation between deep learning, machine learning, and artificial intelligence (AI) at a high level in Fig. 2.

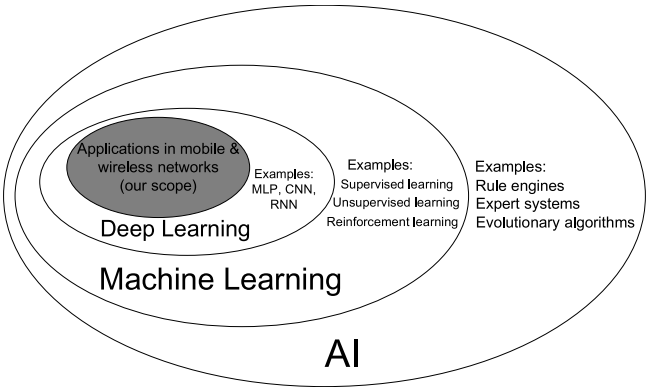


Fig. 2. Venn diagram of the relation between deep learning, machine learning, and AI. This survey particularly focuses on deep learning applications in mobile and wireless networks.

In general, AI is a computation paradigm that endows machines with intelligence, aiming to teach them how to work, react, and learn like humans. Many techniques fall under this broad umbrella, including machine learning, expert systems, and evolutionary algorithms. Among these, machine learning enables the artificial processes to absorb knowledge from data and make decisions without being explicitly programmed. Machine learning algorithms are typically categorized into supervised, unsupervised, and reinforcement learning. Deep learning is a family of machine learning techniques that mimic biological nervous systems and perform representation learning through multi-layer transformations, extending across all three learning paradigms mentioned before. As deep learning has growing number of applications in mobile an wireless networking, the crossovers between these domains make the scope of this manuscript.

A. The Evolution of Deep Learning

The discipline traces its origins 75 years back, when threshold logic was employed to produce a computational model for neural networks [84]. However, it was only in the late 1980s that neural networks (NNs) gained interest, as Rumelhart *et al.* [85] showed that multi-layer NNs could be trained effectively by back-propagating errors. LeCun and Bengio subsequently proposed the now popular Convolutional Neural Network (CNN) architecture [86], but progress stalled due to computing power limitations of systems available at that time. Following the recent success of GPUs, CNNs have

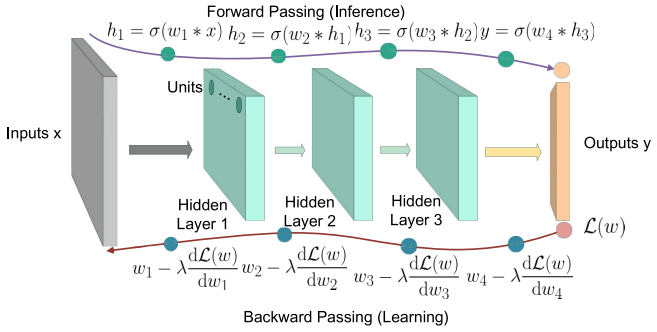


Fig. 3. Illustration of the learning and inference processes of a 4-layer CNN. $w_{(\cdot)}$ denote weights of each hidden layer, $\sigma(\cdot)$ is an activation function, λ refers to the learning rate, $*(\cdot)$ denotes the convolution operation and $\mathcal{L}(w)$ is the loss function to be optimized.

been employed to dramatically reduce the error rate in the Large Scale Visual Recognition Challenge (LSVRC) [87]. This has drawn unprecedented interest in deep learning and breakthroughs continue to appear in a wide range of computer science areas.

B. Fundamental Principles of Deep Learning

The key aim of deep neural networks is to approximate complex functions through a composition of simple and predefined operations of units (or neurons). Such an objective function can be almost of any type, such as a mapping between images and their class labels (classification), computing future stock prices based on historical values (regression), or even deciding the next optimal chess move given the current status on the board (control). The operations performed are usually defined by a weighted combination of a specific group of hidden units with a non-linear activation function, depending on the structure of the model. Such operations along with the output units are named "layers". The neural network architecture resembles the perception process in a brain, where a specific set of units are activated given the current environment, influencing the output of the neural network model.

C. Forward and Backward Propagation

In mathematical terms, the architecture of deep neural networks is usually differentiable, therefore the weights (or parameters) of the model can be learned by minimizing a loss function using gradient descent methods through back-propagation, following the fundamental chain rule [85]. We illustrate the principles of the learning and inference processes of a deep neural network in Fig. 3, where we use a two-dimensional (2D) Convolutional Neural Network (CNN) as an example.

Forward Propagation: The figure shows a CNN with 5 layers, i.e., an input layer (grey), 3 hidden layers (blue) and an output layer (orange). In forward propagation, A 2D input $\mathbf{x}$ (e.g., images) is first processed by a convolutional layer, which perform the following convolutional operation:

$$h_1 = \sigma(w_1 * \mathbf{x}). \tag{1}$$

Here h_1 is the output of the first hidden layer, w_1 is the convolutional filter and $\sigma(\cdot)$ is the activation function, aiming at improving the non-linearity and representability of the model. The output h_1 is subsequently provided as input to and processed by the following two convolutional layers, which eventually produces a final output $\mathbf{y}$. This could be for instance vector of probabilities for different possible patterns (shapes) discovered in the (image) input. To train the CNN appropriately, one uses a loss function $\mathcal{L}(w)$ to measure the distance between the output $\mathbf{y}$ and the ground truth $\mathbf{y}^*$. The purpose of training is to find the best weights $\mathbf{w}$, so as to minimize the loss function $\mathcal{L}(w)$. This can be achieved by the back propagation through gradient descent.

Backward Propagation: During backward propagation, one computes the gradient of the loss function $\mathcal{L}(w)$ over the weight of the last hidden layer, and updates the weight by computing:

$$w_4 = w_4 - \lambda\frac{d\mathcal{L}(w)}{dw_4}. \tag{2}$$

Here λ denotes the learning rate, which controls the step size of moving in the direction indicated by the gradient. The same operation is performed for each weight, following the chain rule. The process is repeated and eventually the gradient descent will lead to a set w that minimizes the $\mathcal{L}(w)$.

For other NN structures, the training and inference processes are similar. To help less expert readers we detail the principles and computational details of various deep learning techniques in Section V.

TABLE IV
SUMMARY OF THE BENEFITS OF APPLYING DEEP LEARNING TO SOLVE PROBLEMS IN MOBILE AND WIRELESS NETWORKS

Key aspect	Description	Benefits
Feature extraction	Deep neural networks can automatically extract high-level features through layers of different depths.	Reduce expensive hand-crafted feature engineering in processing heterogeneous and noisy mobile big data.
Big data exploitation	Unlike traditional ML tools, the performance of deep learning usually grow significantly with the size of training data.	Efficiently utilize huge amounts of mobile data generated at high rates.
Unsupervised learning	Deep learning is effective in processing un-/semi-labeled data, enabling unsupervised learning.	Handling large amounts of unlabeled data, which are common in mobile system.
Multi-task learning	Features learned by neural networks through hidden layers can be applied to different tasks by transfer learning.	Reduce computational and memory requirements when performing multi-task learning in mobile systems.
Geometric mobile data learning	Dedicated deep learning architectures exist to model geometric mobile data	Revolutionize geometric mobile data analysis

D. Advantages of Deep Learning in Mobile and Wireless Networking

We recognize several benefits of employing deep learning to address network engineering problems, as summarized in Table IV. Specifically:

1) It is widely acknowledged that, while vital to the performance of traditional ML algorithms, feature

engineering is costly [88]. A key advantage of deep learning is that it can automatically extract high-level features from data that has complex structure and inner correlations. The learning process does not need to be designed by a human, which tremendously simplifies prior feature handcrafting [20]. The importance of this is amplified in the context of mobile networks, as mobile data is usually generated by heterogeneous sources, is often noisy, and exhibits non-trivial spatial/temporal patterns [17], whose labeling would otherwise require outstanding human effort.

2) Secondly, deep learning is capable of handling large amounts of data. Mobile networks generate high volumes of different types of data at fast pace. Training traditional ML algorithms (e.g., Support Vector Machine (SVM) [89] and Gaussian Process (GP) [90]) sometimes requires to store all the data in memory, which is computationally infeasible under big data scenarios. Furthermore, the performance of ML does not grow significantly with large volumes of data and plateaus relatively fast [18]. In contrast, Stochastic Gradient Descent (SGD) employed to train NNs only requires sub-sets of data at each training step, which guarantees deep learning's scalability with big data. Deep neural networks further benefit as training with big data prevents model over-fitting.
3) Traditional supervised learning is only effective when sufficient labeled data is available. However, most current mobile systems generate unlabeled or semi-labeled data [17]. Deep learning provides a variety of methods that allow exploiting unlabeled data to learn useful patterns in an unsupervised manner, e.g., Restricted Boltzmann Machine (RBM) [91], Generative Adversarial Network (GAN) [92]. Applications include clustering [93], data distributions approximation [92], un/semi-supervised learning [94], [95], and one/zero shot learning [96], [97], among others.
4) Compressive representations learned by deep neural networks can be shared across different tasks, while this is limited or difficult to achieve in other ML paradigms (e.g., linear regression, random forest, etc.). Therefore, a single model can be trained to fulfill multiple objectives, without requiring complete model retraining for different tasks. We argue that this is essential for mobile network engineering, as it reduces computational and memory requirements of mobile systems when performing multi-task learning applications [98].
5) Deep learning is effective in handing geometric mobile data [99], while this is a conundrum for other ML approaches. Geometric data refers to multivariate data represented by coordinates, topology, metrics and order [100]. Mobile data, such as mobile user location and network connectivity can be naturally represented by point clouds and graphs, which have important geometric properties. These data can be effectively modelled by dedicated deep learning architectures, such as PointNet++ [101] and Graph CNN [102]. Employing these architectures has great potential to revolutionize the geometric mobile data analysis [103].

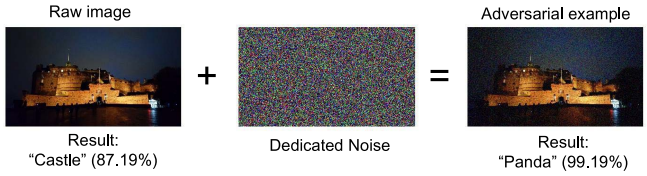


Fig. 4. An example of an adversarial attack on deep learning.

E. Limitations of Deep Learning in Mobile and Wireless Networking

Although deep learning has unique advantages when addressing mobile network problems, it also has several shortcomings, which partially restricts its applicability in this domain. Specifically,

1) In general, deep learning (including deep reinforcement learning) is vulnerable to adversarial examples [104], [105]. These refer to artifact inputs that are intentionally designed by an attacker to fool machine learning models into making mistakes [104]. While it is difficult to distinguish such samples from genuine ones, they can trigger mis-adjustments of a model with high likelihood. We illustrate an example of such an adversarial attack in Fig. 4. Deep learning, especially CNNs are vulnerable to these types of attacks. This may also affect the applicability of deep learning in mobile systems. For instance, hackers may exploit this vulnerability and construct cyber attacks that subvert deep learning based detectors [106]. Constructing deep models that are robust to adversarial examples is imperative, but remains challenging.
2) Deep learning algorithms are largely black boxes and have low interpretability. Their major breakthroughs are in terms of accuracy, as they significantly improve performance of many tasks in different areas. However, although deep learning enables creating "machines" that have high accuracy in specific tasks, we still have limited knowledge as of why NNs make certain decisions. This limits the applicability of deep learning, e.g., in network economics. Therefore, businesses would rather continue to employ statistical methods that have high interpretability, whilst sacrificing on accuracy. Researchers have recognized this problem and investing continuous efforts to address this limitation of deep learning (e.g., [107]–[109]).
3) Deep learning is heavily reliant on data, which sometimes can be more important than the model itself. Deep models can further benefit from training data augmentation [110]. This is indeed an opportunity for mobile networking, as networks generates tremendous amounts of data. However, data collection may be costly, and face privacy concern, therefore it may be difficult to obtain sufficient information for model training. In such scenarios, the benefits of employing deep learning may be outweigth by the costs.

TABLE V
SUMMARY OF TOOLS AND TECHNIQUES THAT ENABLE DEPLOYING DEEP LEARNING IN MOBILE SYSTEMS

Technique	Examples	Scope	Functionality	Performance improvement	Energy consumption	Economic cost
Advanced parallel computing	GPU, TPU [115], CUDA [116], cuDNN [117]	Mobile servers, workstations	Enable fast, parallel training/inference of deep learning models in mobile applications	High	High	Medium (hardware)
Dedicated deep learning library	TensorFlow [118], Theano [119], Caffe [120], Torch [121]	Mobile servers and devices	High-level toolboxes that enable network engineers to build purpose-specific deep learning architectures	Medium	Associated with hardware	Low (software)
Fog computing	nn-X [122], ncnn [123], Kirin 970 [124], Core ML [125]	Mobile devices	Support edge-based deep learning computing	Medium	Low	Medium (hardware)
Fast optimization algorithms	Nesterov [126], Adagrad [127], RMSprop, Adam [128]	Training deep architectures	Accelerate and stabilize the model optimization process	Medium	Associated with hardware	Low (software)
Distributed machine learning systems	MLbase [129], Gaia [10], Tux^2 [11], Adam [130], GeePS [131]	Distributed data centers, cross-server	Support deep learning frameworks in mobile systems across data centers	High	High	High (hardware)

4) Deep learning can be computationally demanding. Advanced parallel computing (e.g., GPUs, high-performance chips) fostered the development and popularity of deep learning, yet deep learning also heavily relies on these. Deep NNs usually require complex structures to obtain satisfactory accuracy performance. However, when deploying NNs on embedded and mobile devices, energy and capability constraints have to be considered. Very deep NNs may not be suitable for such scenario and this would inevitably compromise accuracy. Solutions are being developed to mitigate this problem and we will dive deeper into these in Sections IV and VII.
5) Deep neural networks usually have many hyper-parameters and finding their optimal configuration can be difficult. For a single convolutional layer, we need to configure at least hyper-parameters for the number, shape, stride, and dilation of filters, as well as for the residual connections. The number of such hyper-parameters grows exponentially with the depth of the model and can highly influence its performance. Finding a good set of hyper-parameters can be similar to looking for a needle in a haystack. The AutoML platform[2] provides a first solution to this problem, by employing progressive neural architecture search [111]. This task, however, remains costly.

To circumvent some of the aforementioned problems and allow for effective deployment in mobile networks, deep learning requires certain system and software support. We review and discuss such enablers in the next section.

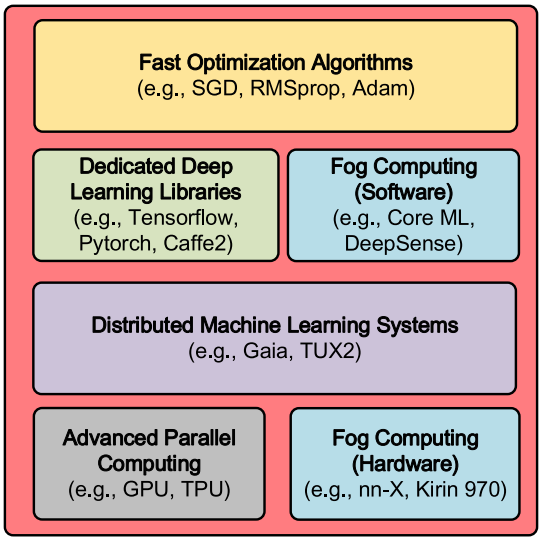


Fig. 5. Hierarchical view of deep learning enablers. Parallel computing and hardware in fog computing lay foundations for deep learning. Distributed machine learning systems can build upon them, to support large-scale deployment of deep learning. Deep learning libraries run at the software level, to enable fast deep learning implementation. Higher-level optimizers are used to train the NN, to fulfill specific objectives.

IV. ENABLING DEEP LEARNING IN MOBILE NETWORKING

5G systems seek to provide high throughput and ultra-low latency communication services, to improve users' QoE [4]. Implementing deep learning to build intelligence into 5G systems, so as to meet these objectives is expensive. This is because powerful hardware and software is required to support training and inference in complex settings. Fortunately, several tools are emerging, which make deep learning in mobile networks tangible; namely, *(i)* advanced parallel computing, *(ii)* distributed machine learning systems, *(iii)* dedicated deep learning libraries, *(iv)* fast optimization algorithms, and *(v)* fog computing. These tools can be seen as forming a hierarchical structure, as illustrated in Fig. 5; synergies between them exist that makes networking problem amenable to deep learning based solutions. By employing these tools, once the training is completed, inferences can be made within millisecond timescales, as already reported by a number of papers for a range of tasks (e.g., [112]–[114]). We summarize these advances in Table V and review them in what follows.

[2]AutoML – training high-quality custom machine learning models with minimum effort and machine learning expertise. https://cloud.google.com/automl/.

A. Advanced Parallel Computing

Compared to traditional machine learning models, deep neural networks have significantly larger parameters spaces, intermediate outputs, and number of gradient values. Each of these need to be updated during every training step, requiring powerful computation resources. The training and inference processes involve huge amounts of matrix multiplications and other operations, though they could be massively parallelized. Traditional Central Processing Units (CPUs) have a limited number of cores, thus they only support restricted computing parallelism. Employing CPUs for deep learning implementations is highly inefficient and will not satisfy the low-latency requirements of mobile systems.

Engineers address these issues by exploiting the power of GPUs. GPUs were originally designed for high performance video games and graphical rendering, but new techniques such as Compute Unified Device Architecture (CUDA) [116] and the CUDA Deep Neural Network library (cuDNN) [117] developed by NVIDIA add flexibility to this type of hardware, allowing users to customize their usage for specific purposes. GPUs usually incorporate thousand of cores and perform exceptionally in fast matrix multiplications required for training neural networks. This provides higher memory bandwidth over CPUs and dramatically speeds up the learning process. Recent advanced Tensor Processing Units (TPUs) developed by Google even demonstrate 15-30$\times$ higher processing speeds and 30-80$\times$ higher performance-per-watt, as compared to CPUs and GPUs [115].

Diffractive neural networks ($\mathrm{D}^2\mathrm{NNs}$) that completely rely on light communication were recently introduced in [132], to enable zero-consumption and zero-delay deep learning. The $\mathrm{D}^2\mathrm{NN}$ is composed of several transmissive layers, where points on these layers act as neurons in a NN. The structure is trained to optimize the transmission/reflection coefficients, which are equivalent to weights in a NN. Once trained, transmissive layers will be materialized via 3D printing and they can subsequently be used for inference.

There are also a number of toolboxes that can assist the computational optimization of deep learning on the server side. Spring and Shrivastava [133] introduce a hashing based technique that substantially reduces computation requirements of deep network implementations. Mirhoseini *et al.* employ a reinforcement learning scheme to enable machines to learn the optimal operation placement over mixture hardware for deep neural networks. Their solution achieves up to 20% faster computation speed than human experts' designs of such placements [134].

Importantly, these systems are easy to deploy, therefore mobile network engineers do not need to rebuild mobile servers from scratch to support deep learning computing. This makes implementing deep learning in mobile systems feasible and accelerates the processing of mobile data streams.

B. Distributed Machine Learning Systems

Mobile data is collected from heterogeneous sources (e.g., mobile devices, network probes, etc.), and stored in multiple distributed data centers. With the increase of data volumes, it is impractical to move all mobile data to a central data center to run deep learning applications [10]. Running network-wide deep learning algorithms would therefore require distributed machine learning systems that support different interfaces (e.g., operating systems, programming language, libraries), so as to enable training and evaluation of deep models across geographically distributed servers simultaneously, with high efficiency and low overhead.

Deploying deep learning in a distributed fashion will inevitably introduce several system-level problems, which require satisfying the following properties.

Consistency – Guaranteeing that model parameters and computational processes are consistent across all machines.

Fault tolerance – Effectively dealing with equipment breakdowns in large-scale distributed machine learning systems.

Communication – Optimizing communication between nodes in a cluster and to avoid congestion.

Storage – Designing efficient storage mechanisms tailored to different environments (e.g., distributed clusters, single machines, GPUs), given I/O and data processing diversity.

Resource management – Assigning workloads and ensuring that nodes work well-coordinated.

Programming model – Designing programming interfaces to support multiple programming languages.

There exist several distributed machine learning systems that facilitate deep learning in mobile networking applications. Kraska *et al.* [129] introduce a distributed system named MLbase, which enables to intelligently specify, select, optimize, and parallelize ML algorithms. Their system helps non-experts deploy a wide range of ML methods, allowing optimization and running ML applications across different servers. Hsieh *et al.* [10] develop a geography-distributed ML system called Gaia, which breaks the throughput bottleneck by employing an advanced communication mechanism over Wide Area Networks, while preserving the accuracy of ML algorithms. Their proposal supports versatile ML interfaces (e.g., TensorFlow, Caffe), without requiring significant changes to the ML algorithm itself. This system enables deployments of complex deep learning applications over large-scale mobile networks.

Xing *et al.* [135] develop a large-scale machine learning platform to support big data applications. Their architecture achieves efficient model and data parallelization, enabling parameter state synchronization with low communication cost. Xiao *et al.* [11] propose a distributed graph engine for ML named TUX^2, to support data layout optimization across machines and reduce cross-machine communication. They demonstrate remarkable performance in terms of runtime and convergence on a large dataset with up to 64 billion edges. Chilimbi *et al.* [130] build a distributed, efficient, and scalable system named "Adam"[3] tailored to the training of deep models. Their architecture demonstrates impressive performance in terms of throughput, delay, and fault tolerance. Another dedicated distributed deep learning system called GeePS is developed by Cui *et al.* [131]. Their framework allows data

[3]Note that this is distinct from the Adam optimizer discussed in Section IV-D.

TABLE VI
SUMMARY AND COMPARISON OF MAINSTREAM DEEP LEARNING LIBRARIES

Library	Low-Layer Language	Available Interface	Pros	Cons	Mobile Supported	Popularity	Upper-Level Library
TensorFlow	C++	Python, Java, C, C++, Go	• Large user community • Well-written document • Complete functionality • Provides visualization tool (TensorBoard) • Multiple interfaces support • Allows distributed training and model serving	• Difficult to debug • The package is heavy • Higher entry barrier for beginners	Yes	High	Keras, TensorLayer, Luminoth
Theano	Python	Python	• Flexible • Good running speed	• Difficult to learn • Long compilation time • No longer maintained	No	Low	Keras, Blocks, Lasagne
Caffe(2)	C++	Python, Matlab	• Fast runtime • Multiple platform s support	• Small user base • Modest documentation	Yes	Medium	None
(Py)Torch	Lua, C++	Lua, Python, C, C++	• Easy to build models • Flexible • Well documented • Easy to debug • Rich pretrained models available • Declarative data parallelism	• Has limited resource • Lacks model serving • Lacks visualization tools	Yes	High	None
MXNET	C++	C++, Python, Matlab, R	• Lightweight • Memory-efficient • Fast training • Simple model serving • Highly scalable	• Small user base • Difficult to learn	Yes	Low	Gluon

parallelization on distributed GPUs, and demonstrates higher training throughput and faster convergence rate. More recently, Moritz *et al.* [136] designed a dedicated distributed framework named Ray to underpin reinforcement learning applications. Their framework is supported by an dynamic task execution engine, which incorporates the actor and task-parallel abstractions. They further introduce a bottom-up distributed scheduling strategy and a dedicated state storage scheme, to improve scalability and fault tolerance.

C. Dedicated Deep Learning Libraries

Building a deep learning model from scratch can prove complicated to engineers, as this requires definitions of forwarding behaviors and gradient propagation operations at each layer, in addition to CUDA coding for GPU parallelization. With the growing popularity of deep learning, several dedicated libraries simplify this process. Most of these toolboxes work with multiple programming languages, and are built with GPU acceleration and automatic differentiation support. This eliminates the need of hand-crafted definition of gradient propagation. We summarize these libraries below, and give a comparison among them in Table VI.

TensorFlow[4] is a machine learning library developed by Google [118]. It enables deploying computation graphs on CPUs, GPUs, and even mobile devices [137], allowing ML implementation on both single and distributed architectures. This permits fast implementation of deep NNs on both cloud and fog services. Although originally designed for ML and deep neural networks applications, TensorFlow is also suitable for other data-driven research purposes. It provides TensorBoard,[5] a sophisticated visualization tool, to help users understand model structures and data flows, and perform debugging. Detailed documentation and tutorials for Python exist, while other programming languages such as C, Java, and Go are also supported. currently it is the most popular deep learning library. Building upon TensorFlow, several dedicated deep learning toolboxes were released to provide higher-level programming interfaces, including Keras,[6] Luminoth[7] and TensorLayer [138].

Theano is a Python library that allows to efficiently define, optimize, and evaluate numerical computations involving multi-dimensional data [119]. It provides both GPU and CPU modes, which enables users to tailor their programs to individual machines. Learning Theano is however difficult and building a NNs with it involves substantial compiling time. Though Theano has a large user base and a support community, and at some stage was one of the most popular deep learning tools, its popularity is decreasing rapidly, as core ideas and attributes are absorbed by TensorFlow.

Caffe(2) is a dedicated deep learning framework developed by Berkeley AI Research [120] and the latest version, Caffe2,[8] was recently released by Facebook. Inheriting all the advantages of the old version, Caffe2 has become a very flexible framework that enables users to build their models efficiently. It also allows to train neural networks on multiple GPUs within distributed systems, and supports deep learning implementations on mobile operating systems, such as iOS and Android.

[4]TensorFlow, https://www.tensorflow.org/.

[5]TensorBoard – A visualization tool for TensorFlow, https://www.tensorflow.org/guide/summaries_and_tensorboard.

[6]Keras deep learning library, https://github.com/fchollet/keras.

[7]Luminoth deep learning library for computer vision, https://github.com/tryolabs/luminoth.

[8]Caffe2, https://caffe2.ai/.

Therefore, it has the potential to play an important role in the future mobile edge computing.

(Py)Torch is a scientific computing framework with wide support for machine learning models and algorithms [121]. It was originally developed in the Lua language, but developers later released an improved Python version [139]. In essence PyTorch is a lightweight toolbox that can run on embedded systems such as smart phones, but lacks comprehensive documentations. Since building NNs in PyTorch is straightforward, the popularity of this library is growing rapidly. It also offers rich pretrained models and modules that are easy to reuse and combine. PyTorch is now officially maintained by Facebook and mainly employed for research purposes.

MXNET is a flexible and scalable deep learning library that provides interfaces for multiple languages (e.g., C++, Python, MATLAB, R, etc.) [140]. It supports different levels of machine learning models, from logistic regression to GANs. MXNET provides fast numerical computation for both single machine and distributed ecosystems. It wraps workflows commonly used in deep learning into high-level functions, such that standard neural networks can be easily constructed without substantial coding effort. However, learning how to work with this toolbox in short time frame is difficult, hence the number of users who prefer this library is relatively small. MXNET is the official deep learning framework in Amazon.

Although less popular, there are other excellent deep learning libraries, such as CNTK,[9] Deeplearning4j,[10] Blocks,[11] Gluon,[12] and Lasagne,[13] which can also be employed in mobile systems. Selecting among these varies according to specific applications. For AI beginners who intend to employ deep learning for the networking domain, PyTorch is a good candidate, as it is easy to build neural networks in this environment and the library is well optimized for GPUs. On the other hand, if for people who pursue advanced operations and large-scale implementation, Tensorflow might be a better choice, as it is well-established, under good maintainance and has standed the test of many Google industrial projects.

D. Fast Optimization Algorithms

The objective functions to be optimized in deep learning are usually complex, as they involve sums of extremely large numbers of data-wise likelihood functions. As the depth of the model increases, such functions usually exhibit high non-convexity with multiple local minima, critical points, and saddle points. In this case, conventional Stochastic Gradient Descent (SGD) algorithms [141] are slow in terms of convergence, which will restrict their applicability to latency constrained mobile systems. To overcome this problem and stabilize the optimization process, many algorithms evolve the traditional SGD, allowing NN models to be trained faster for mobile applications. We summarize the key principles behind these optimizers and make a comparison between them in Table VII. We delve into the details of their operation next.

Fixed Learning Rate SGD Algorithms: Sutskever *et al.* [126] introduce a variant of the SGD optimizer with Nesterov's momentum, which evaluates gradients after the current velocity is applied. Their method demonstrates faster convergence rate when optimizing convex functions. Another approach is Adagrad, which performs adaptive learning to model parameters according to their update frequency. This is suitable for handling sparse data and significantly outperforms SGD in terms of robustness [127]. Adadelta improves the traditional Adagrad algorithm, enabling it to converge faster, and does not rely on a global learning rate [142]. RMSprop is a popular SGD based method introduced by G. Hinton. RMSprop divides the learning rate by an exponential smoothing the average of gradients and does not require one to set the learning rate for each training step [141].

Adaptive Learning Rate SGD Algorithms: Kingma and Ba [128] propose an adaptive learning rate optimizer named Adam, which incorporates momentum by the first-order moment of the gradient. This algorithm is fast in terms of convergence, highly robust to model structures, and is considered as the first choice if one cannot decide what algorithm to use. By incorporating the momentum into Adam, Nadam applies stronger constraints to the gradients, which enables faster convergence [143].

Other Optimizers: Andrychowicz *et al.* [144] suggest that the optimization process can be even learned dynamically. They pose the gradient descent as a trainable learning problem, which demonstrates good generalization ability in neural network training. Wen *et al.* [147] propose a training algorithm tailored to distributed systems. They quantize float gradient values to $\{-1, 0 \text{ and } +1\}$ in the training processing, which theoretically require 20 times less gradient communications between nodes. Szegedy *et al.* prove that such gradient approximation mechanism allows the objective function to converge to optima with probability 1, where in their experiments only a 2% accuracy loss is observed on average on GoogleLeNet [145] training. Zhou *et al.* [146] employ a differential private mechanism to compare training and validation gradients, to reuse samples and keep them fresh. This can dramatically reduce overfitting during training.

E. Fog Computing

The fog computing paradigm presents a new opportunity to implement deep learning in mobile systems. Fog computing refers to a set of techniques that permit deploying applications or data storage at the edge of networks [148], e.g., on individual mobile devices. This reduces the communications overhead, offloads data traffic, reduces user-side latency, and lightens the sever-side computational burdens [149], [150]. A formal definition of fog computing is given in [151], where this is interpreted as *'a huge number of heterogeneous (wireless and sometimes autonomous) ubiquitous and decentralized devices [that] communicate and potentially cooperate among*

[9]MS Cognitive Toolkit, https://www.microsoft.com/en-us/cognitive-toolkit/.

[10]Deeplearning4j, http://deeplearning4j.org.

[11]Blocks, A Theano framework for building and training neural networks https://github.com/mila-udem/blocks.

[12]Gluon, A deep learning library https://gluon.mxnet.io/.

[13]Lasagne, https://github.com/Lasagne.

TABLE VII
SUMMARY AND COMPARISON OF DIFFERENT OPTIMIZATION ALGORITHMS

Optimization algorithm	Core idea	Pros	Cons
SGD [141]	Computes the gradient of mini-batches iteratively and updates the parameters	• Easy to implement	• Setting a global learning rate required • Algorithm may get stuck on saddle points or local minima • Slow in terms of convergence • Unstable
Nesterov's momentum [126]	Introduces momentum to maintain the last gradient direction for the next update	• Stable • Faster learning • Can escape local minima	• Setting a learning rate needed
Adagrad [127]	Applies different learning rates to different parameters	• Learning rate tailored to each parameter • Handle sparse gradients well	• Still requires setting a global learning rate • Gradients sensitive to the regularizer • Learning rate becomes very slow in the late stages
Adadelta [142]	Improves Adagrad, by applying a self-adaptive learning rate	• Does not rely on a global learning rate • Faster speed of convergence • Fewer hyper-parameters to adjust	• May get stuck in a local minima at late training
RMSprop [141]	Employs root mean square as a constraint of the learning rate	• Learning rate tailored to each parameter • Learning rate do not decrease dramatically at late training • Works well in RNN training	• Still requires a global learning rate • Not good at handling sparse gradients
Adam [128]	Employs a momentum mechanism to store an exponentially decaying average of past gradients	• Learning rate stailored to each parameter • Good at handling sparse gradients and non-stationary problems • Memory-efficient • Fast convergence	• It may turn unstable during training
Nadam [143]	Incorporates Nesterov accelerated gradients into Adam	• Works well in RNN training	—
Learn to optimize [144]	Casts the optimization problem as a learning problem using a RNN	• Does not require to design the learning by hand	• Require an additional RNN for learning in the optimizer
Quantized training [145]	Quantizes the gradients into {-1, 0, 1} for training	• Good for distributed training • Memory-efficient	• Loses training accuracy
Stable gradient descent [146]	Employs a differential private mechanism to compare training and validation gradients, to reuse samples and keep them fresh.	• More stable • Less overfitting • Converges faster than SGD	• Only validated on convex functions

them and with the network to perform storage and processing tasks without the intervention of third parties.' To be more concrete, it can refer to smart phones, wearables devices and vehicles which store, analyze and exchange data, to offload the burden from cloud and perform more delay-sensitive tasks [152], [153]. Since fog computing involves deployment at the edge, participating devices usually have limited computing resource and battery power. Therefore, special hardware and software are required for deep learning implementation, as we explain next.

Hardware: There exist several efforts that attempt to shift deep learning computing from the cloud side to mobile devices [154]. For example, Gokhale *et al.* [122] develop a mobile coprocessor named neural network neXt (nn-X), which accelerates the deep neural networks execution in mobile devices, while retaining low energy consumption. Bang *et al.* [155] introduce a low-power and programmable deep learning processor to deploy mobile intelligence on edge devices. Their hardware only consumes 288 μW but achieves 374 GOPS/W efficiency. A Neurosynaptic Chip called TrueNorth is proposed by IBM [156]. Their solution seeks to support computationally intensive applications on embedded battery-powered mobile devices. Qualcomm introduces a Snapdragon neural processing engine to enable deep learning computational optimization tailored to mobile devices.[14] Their hardware allows developers to execute neural network models on Snapdragon 820 boards to serve a variety of applications. In close collaboration with Google, Movidius[15] develops an embedded neural network computing framework that allows user-customized deep learning deployments at the edge of mobile networks. Their products can achieve satisfying runtime efficiency, while operating with ultra-low power requirements. It further supports difference

[14]Qualcomm Helps Make Your Mobile Devices Smarter With New Snapdragon Machine Learning Software Development Kit: https://www.qualcomm.com/news/releases/2016/05/02/qualcomm-helps-make-your-mobile-devices-smarter-new-snapdragon-machine.

[15]Movidius, an Intel company, provides cutting edge solutions for deploying deep learning and computer vision algorithms on ultra-low power devices. https://www.movidius.com/.

TABLE VIII
COMPARISON OF MOBILE DEEP LEARNING PLATFORM

Platform	Developer	Mobile hardware supported	Speed	Code size	Mobile compatibility	Open-sourced
TensorFlow	Google	CPU	Slow	Medium	Medium	Yes
Caffe	Facebook	CPU	Slow	Large	Medium	Yes
ncnn	Tencent	CPU	Medium	Small	Good	Yes
CoreML	Apple	CPU/GPU	Fast	Small	Only iOS 11+ supported	No
DeepSense	Yao *et al.*	CPU	Medium	Unknown	Medium	No

frameworks, such as TensorFlow and Caffe, providing users with flexibility in choosing among toolkits. More recently, Huawei officially announced the Kirin 970 as a mobile AI computing system on chip.[16] Their innovative framework incorporates dedicated Neural Processing Units (NPUs), which dramatically accelerates neural network computing, enabling classification of 2,000 images per second on mobile devices.

Software: Beyond these hardware advances, there are also software platforms that seek to optimize deep learning on mobile devices (e.g., [157]). We compare and summarize all these platforms in Table VIII.[17] In addition to the mobile version of TensorFlow and Caffe, Tencent released a lightweight, high-performance neural network inference framework tailored to mobile platforms, which relies on CPU computing.[18] This toolbox performs better than all known CPU-based open source frameworks in terms of inference speed. Apple has developed "Core ML", a private ML framework to facilitate mobile deep learning implementation on iOS 11.[19] This lowers the entry barrier for developers wishing to deploy ML models on Apple equipment. Yao *et al.* develop a deep learning framework called DeepSense dedicated to mobile sensing related data processing, which provides a general machine learning toolbox that accommodates a wide range of edge applications. It has moderate energy consumption and low latency, thus being amenable to deployment on smartphones.

The techniques and toolboxes mentioned above make the deployment of deep learning practices in mobile network applications feasible. In what follows, we briefly introduce several representative deep learning architectures and discuss their applicability to mobile networking problems.

V. DEEP LEARNING: STATE-OF-THE-ART

Revisiting Fig. 2, machine learning methods can be naturally categorized into three classes, namely supervised learning, unsupervised learning, and reinforcement learning. Deep learning architectures have achieved remarkable performance in all these areas. In this section, we introduce the key principles underpinning several deep learning models and discuss their largely unexplored potential to solve mobile networking problems. Technical details of classical models are provided to readers who seek to obtain a deeper understanding of neural networks. The more experienced can continue reading with Section VI. We illustrate and summarize the most salient architectures that we present in Fig. 6 and Table IX, respectively.

A. Multilayer Perceptron

The Multilayer Perceptrons (MLPs) is the initial Artificial Neural Network (ANN) design, which consists of at least three layers of operations [174]. Units in each layer are densely connected, hence require to configure a substantial number of weights. We show an MLP with two hidden layers in Fig. 6(a). Note that usually only MLPs containing more than one hidden layer are regarded as deep learning structures.

Given an input vector $\mathbf{x}$, a standard MLP layer performs the following operation:

$$\mathbf{y} = \sigma(W \cdot \mathbf{x} + b). \tag{3}$$

Here $\mathbf{y}$ denotes the output of the layer, W are the weights and b the biases. $\sigma(\cdot)$ is an activation function, which aims at improving the non-linearity of the model. Commonly used activation function are the sigmoid,

$$\mathbf{sigmoid}(\mathbf{x}) = \frac{1}{1+e^{-\mathbf{x}}},$$

the Rectified Linear Unit (ReLU) [175],

$$\mathbf{ReLU}(\mathbf{x}) = \max(\mathbf{x}, 0),$$

tanh,

$$\mathbf{tanh}(\mathbf{x}) = \frac{e^{\mathbf{x}} - e^{-\mathbf{x}}}{e^{\mathbf{x}} + e^{-\mathbf{x}}},$$

and the Scaled Exponential Linear Units (SELUs) [176],

$$\mathbf{SELU}(\mathbf{x}) = \lambda \begin{cases} \mathbf{x}, & \text{if } \mathbf{x} > 0; \\ \alpha e^{\mathbf{x}} - \alpha, & \text{if } \mathbf{x} \leq 0, \end{cases}$$

where the parameters $\lambda = 1.0507$ and $\alpha = 1.6733$ are frequently used. In addition, the softmax function is typically employed in the last layer when performing classification:

$$\mathbf{softmax}(\mathbf{x}_i) = \frac{e^{\mathbf{x}_i}}{\sum_{j=0}^{k} e^{\mathbf{x}_k}},$$

where k is the number of labels involved in classification. Until recently, **sigmoid** and **tanh** have been the activation functions most widely used. However, they suffer from a known gradient vanishing problem, which hinders gradient propagation through layers. Therefore these functions are increasingly more often replaced by ReLU or SELU. SELU enables to normalize the output of each layer, which dramatically accelerates the training convergence, and can be viewed as a replacement of Batch Normalization [177].

[16]Huawei announces the Kirin 970 – new flagship SoC with AI capabilities http://www.androidauthority.com/huawei-announces-kirin-970-797788/.

[17]Adapted from https://mp.weixin.qq.com/s/3gTp1kqkiGwdq5olrpOvKw.

[18]ncnn is a high-performance neural network inference framework optimized for the mobile platform, https://github.com/Tencent/ncnn.

[19]Core ML: Integrate machine learning models into your app, https://developer.apple.com/documentation/coreml.

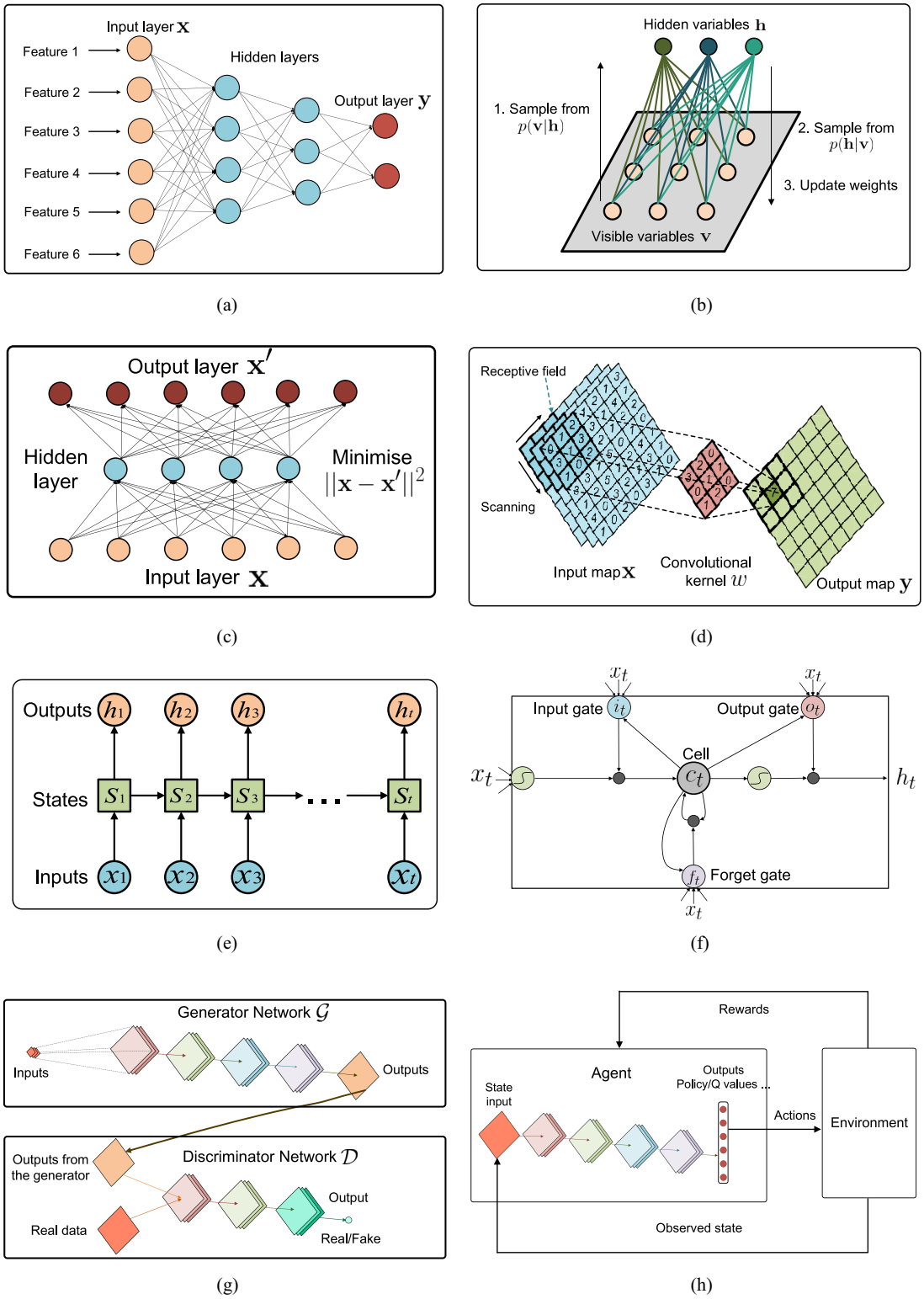


Fig. 6. Typical structure and operation principles of MLP, RBM, AE, CNN, RNN, LSTM, GAN, and DRL. (a) Structure of an MLP with 2 hidden layers (blue circles). (b) Graphical model and training process of an RBM. $\mathbf{v}$ and $\mathbf{h}$ denote visible and hidden variables, respectively. (c) Operating principle of an auto-encoder, which seeks to reconstruct the input from the hidden layer. (d) Operating principle of a convolutional layer. (e) Recurrent layer – $x_{1:t}$ is the input sequence, indexed by time t, s_t denotes the state vector and h_t the hidden outputs. (f) The inner structure of an LSTM layer. (g) Underlying principle of a generative adversarial network (GAN). (h) Typical deep reinforcement learning architecture. The agent is a neural network model that approximates the required function.

The MLP can be employed for supervised, unsupervised, and even reinforcement learning purposes. Although this structure was the most popular neural network in the past, its popularity is decreasing because it entails high complexity (fully-connected structure), modest performance, and low convergence efficiency. MLPs are mostly used as a baseline or

TABLE IX
SUMMARY OF DIFFERENT DEEP LEARNING ARCHITECTURES. GAN AND DRL ARE SHADED, SINCE THEY ARE BUILT UPON OTHER MODELS

Model	Learning scenarios	Example architectures	Suitable problems	Pros	Cons	Potential applications in mobile networks
MLP	Supervised, unsupervised, reinforcement	ANN, AdaNet [158]	Modeling data with simple correlations	Naive structure and straightforward to build	High complexity, modest performance and slow convergence	Modeling multi-attribute mobile data; auxiliary or component of other deep architectures
RBM	Unsupervised	DBN [159], Convolutional DBN [160]	Extracting robust representations	Can generate virtual samples	Difficult to train well	Learning representations from unlabeled mobile data; model weight initialization; network flow prediction
AE	Unsupervised	DAE [161], VAE [162]	Learning sparse and compact representations	Powerful and effective unsupervised learning	Expensive to pretrain with big data	model weight initialization; mobile data dimension reduction; mobile anomaly detection
CNN	Supervised, unsupervised, reinforcement	AlexNet [87], ResNet [163], 3D-ConvNet [164], GoogLeNet [145], DenseNet [165]	Spatial data modeling	Weight sharing; affine invariance	High computational cost; challenging to find optimal hyper-parameters; requires deep structures for complex tasks	Spatial mobile data analysis
RNN	Supervised, unsupervised, reinforcement	LSTM [166], Attention based RNN [167], ConvLSTM [168]	Sequential data modeling	Expertise in capturing temporal dependencies	High model complexity; gradient vanishing and exploding problems	Individual traffic flow analysis; network-wide (spatio-) temporal data modeling
GAN	Unsupervised	WGAN [80], LS-GAN [169], BigGAN [170]	Data generation	Can produce lifelike artifacts from a target distribution	Training process is unstable (convergence difficult)	Virtual mobile data generation; assisting supervised learning tasks in network data analysis
DRL	Reinforcement	DQN [19], Deep Policy Gradient [171], A3C [79], Rainbow [172], DPPO [173]	Control problems with high-dimensional inputs	Ideal for high-dimensional environment modeling	Slow in terms of convergence	Mobile network control and management.

integrated into more complex architectures (e.g., the final layer in CNNs used for classification). Building an MLP is straightforward, and it can be employed, e.g., to assist with feature extraction in models built for specific objectives in mobile network applications. The advanced Adaptive learning of neural Network (AdaNet) enables MLPs to dynamically train their structures to adapt to the input [158]. This new architecture can be potentially explored for analyzing continuously changing mobile environments.

B. Boltzmann Machine

Restricted Boltzmann Machines (RBMs) [91] were originally designed for unsupervised learning purposes. They are essentially a type of energy-based undirected graphical models, and include a visible layer and a hidden layer, and where each unit can only assume binary values (i.e., 0 and 1). The probabilities of these values are given by:

$$P\big(h_j = 1|v\big) = \frac{1}{1 + e^{-\mathbf{W}\cdot\mathbf{v}+\mathbf{b_j}}}$$
$$P\big(v_j = 1|h\big) = \frac{1}{1 + e^{-\mathbf{W^T}\cdot\mathbf{h}+\mathbf{a_j}}},$$

where h, v are the hidden and visible units respectively, and $\mathbf{W}$ are weights and $\mathbf{a}$, $\mathbf{b}$ are biases. The visible units are conditional independent to the hidden units, and *vice versa*. A typical structure of an RBM is shown in Fig. 6(b). In general, input data are assigned to visible units v. Hidden units h are invisible and they fully connect to all v through weights W, which is similar to a standard feed forward neural network. However, unlike in MLPs where only the input vector can affect the hidden units, with RBMs the state of v can affect the state of h, and *vice versa*.

RBMs can be effectively trained using the contrastive divergence algorithm [178] through multiple steps of Gibbs sampling [179]. We illustrate the structure and the training process of an RBM in Fig. 6(b). RBM-based models are usually employed to initialize the weights of a neural network in more recent applications. The pre-trained model can be subsequently fine-tuned for supervised learning purposes using a standard back-propagation algorithm. A stack of RBMs is called a Deep Belief Network (DBN) [159], which performs layer-wise training and achieves superior performance as compared to MLPs in many applications, including time series forecasting [180], ratio matching [181], and speech recognition [182]. Such

structures can be even extended to a convolutional architecture, to learn hierarchical spatial representations [160].

C. Auto-Encoders

Auto-Encoders (AEs) are also designed for unsupervised learning and attempt to copy inputs to outputs. The underlying principle of an AE is shown in Fig. 6(c). AEs are frequently used to learn compact representation of data for dimension reduction [183]. Extended versions can be further employed to initialize the weights of a deep architecture, e.g., the Denoising Auto-Encoder (DAE) [161]), and generate virtual examples from a target data distribution, e.g., Variational Auto-Encoders (VAEs) [162].

A VAE typically comprises two neural networks – an encoder and a decoder. The input of the encoder is a data point $\mathbf{x}$ (e.g., images) and its functionality is to encode this input into a latent representation space $\mathbf{z}$. Let $f_\Theta(\mathbf{z}|\mathbf{x})$ be an encoder parameterized by Θ and $\mathbf{z}$ is sampled from a Gaussian distribution, the objective of the encoder is to output the mean and variance of the Gaussian distribution. Similarly, denoting $g_\Omega(\mathbf{x}|\mathbf{z})$ the decoder parameterized by Ω, this accepts the latent representation $\mathbf{z}$ as input, and outputs the parameter of the distribution of $\mathbf{x}$. The objective of the VAE is to minimize the reconstruction error of the data and the Kullback-Leibler (KL) divergence between $p(\mathbf{z})$ and $f_\Theta(\mathbf{z}|\mathbf{x})$. Once trained, the VAE can generate new data point samples by *(i)* drawing latent variables $z_i \sim p(\mathbf{z})$ and *(ii)* drawing a new data point $x_i \sim p(\mathbf{x}|\mathbf{z})$.

AEs can be employed to address network security problems, as several research papers confirm their effectiveness in detecting anomalies under different circumstances [184]–[186], which we will further discuss in Section VI-H. The structures of RBMs and AEs are based upon MLPs, CNNs or RNNs. Their goals are similar, while their learning processes are different. Both can be exploited to extract patterns from unlabeled mobile data, which may be subsequently employed for various supervised learning tasks, e.g., routing [187], mobile activity recognition [188], [189], periocular verification [190] and base station user number prediction [191].

D. Convolutional Neural Network

Instead of employing full connections between layers, Convolutional Neural Networks (CNNs or ConvNets) employ a set of locally connected kernels (filters) to capture correlations between different data regions. Mathematically, for each location $\boldsymbol{p}_y$ of the output $\mathbf{y}$, the standard convolution performs the following operation:

$$\mathbf{y}(\boldsymbol{p}_y) = \sum_{\boldsymbol{p}_G \in \mathbb{G}} \mathbf{w}(\boldsymbol{p}_G) \cdot \mathbf{x}(\boldsymbol{p}_y + \boldsymbol{p}_G), \tag{4}$$

where $\boldsymbol{p}_G$ denotes all positions in the receptive field $\mathbb{G}$ of the convolutional filter W, effectively representing the receptive range of each neuron to inputs in a convolutional layer. Here the weights W are shared across different locations of the input map. We illustrate the operation of one 2D convolutional layer in Fig. 6(d). Specifically, the inputs of a 2D CNN layer are multiple 2D matrices with different channels (e.g., the RGB representation of images). A convolutional layer employs multiple filters shared across different locations, to "scan" the inputs and produce output maps. In general, if the inputs and outputs have M and N filters respectively, the convolutional layer will require $M \times N$ filters to perform the convolution operation.

CNNs improve traditional MLPs by leveraging three important ideas, namely, *(i)* sparse interactions, *(ii)* parameter sharing, and *(iii)* equivariant representations [18]. This reduces the number of model parameters significantly and maintains the affine invariance (i.e., recognition results are robust to the affine transformation of objects). Specifically, The sparse interactions imply that the weight kernel has smaller size than the input. It performs moving filtering to produce outputs (with roughly the same size as the inputs) for the current layer. Parameter sharing refers to employing the same kernel to scan the whole input map. This significantly reduces the number of parameters needed, which mitigates the risk of over-fitting. Equivariant representations indicate that convolution operations are invariant in terms of translation, scale, and shape. This is particularly useful for image processing, since essential features may show up at different locations in the image, with various affine patterns.

Owing to the properties mentioned above, CNNs achieve remarkable performance in imaging applications. Krizhevsky *et al.* [87] exploit a CNN to classify images on the ImageNet dataset [192]. Their method reduces the top-5 error by 39.7% and revolutionizes the imaging classification field. GoogLeNet [145] and ResNet [163] significantly increase the depth of CNN structures, and propose inception and residual learning techniques to address problems such as over-fitting and gradient vanishing introduced by "depth". Their structure is further improved by the Dense Convolutional Network (DenseNet) [165], which reuses feature maps from each layer, thereby achieving significant accuracy improvements over other CNN based models, while requiring fewer layers. CNNs have also been extended to video applications. Ji *et al.* propose 3D convolutional neural networks for video activity recognition [164], demonstrating superior accuracy as compared to 2D CNN. More recent research focuses on learning the shape of convolutional kernels [193]–[195]. These dynamic architectures allow to automatically focus on important regions in input maps. Such properties are particularly important in analyzing large-scale mobile environments exhibiting clustering behaviors (e.g., surge of mobile traffic associated with a popular event).

Given the high similarity between image and spatial mobile data (e.g., mobile traffic snapshots, users' mobility, etc.), CNN-based models have huge potential for network-wide mobile data analysis. This is a promising future direction that we further discuss in Section VIII.

E. Recurrent Neural Network

Recurrent Neural Networks (RNNs) are designed for modeling sequential data, where sequential correlations exist between samples. At each time step, they produce output via recurrent connections between hidden units [18], as shown in

Fig. 6(e). Given a sequence of inputs $\mathbf{x} = \{x_1, x_2, \ldots, x_T\}$, a standard RNN performs the following operations:

$$s_t = \sigma_s(W_x x_t + W_s s_{t-1} + b_s)$$
$$h_t = \sigma_h(W_h s_t + b_h),$$

where s_t represents the state of the network at time *t* and it constructs a memory unit for the network. Its values are computed by a function of the input x_t and previous state s_{t-1}. h_t is the output of the network at time *t*. In natural language processing applications, this usually represents a language vector and becomes the input at $t + 1$ after being processed by an embedding layer. The weights W_x, W_h and biases b_s, b_h are shared across different temporal locations. This reduces the model complexity and the degree of over-fitting.

The RNN is trained via a Backpropagation Through Time (BPTT) algorithm. However, gradient vanishing and exploding problems are frequently reported in traditional RNNs, which make them particularly hard to train [196]. The Long Short-Term Memory (LSTM) mitigates these issues by introducing a set of "gates" [166], which has been proven successful in many applications (e.g., speech recognition [197], text categorization [198], and wearable activity recognition [113]). A standard LSTM performs the following operations:

$$i_t = \sigma(W_{xi} X_t + W_{hi} H_{t-1} + b_i),$$
$$f_t = \sigma\big(W_{xf} X_t + W_{hf} H_{t-1} + b_f\big),$$
$$C_t = f_t \odot C_{t-1} + i_t \odot \tanh(W_{xc} X_t + W_{hc} H_{t-1} + b_c),$$
$$o_t = \sigma(W_{xo} X_t + W_{ho} H_{t-1} + b_o),$$
$$h_t = o_t \odot \tanh(C_t).$$

Here, '$\odot$' denotes the Hadamard product, C_t denotes the cell outputs, h_t are the hidden states, i_t, f_t, and o_t are input gates, forget gates, and output gates, respectively. These gates mitigate the gradient issues and significantly improve the RNN. We illustrated the structure of an LSTM in Fig. 6(f).

Sutskever *et al.* [167] introduce attention mechanisms to RNNs, which achieves outstanding accuracy in tokenized predictions. Xingjian *et al.* [168] substitute the dense matrix multiplication in LSTMs with convolution operations, designing a Convolutional Long Short-Term Memory (ConvLSTM). Their proposal reduces the complexity of traditional LSTM and demonstrates significantly lower prediction errors in precipitation nowcasting (i.e., forecasting the volume of precipitation).

Mobile networks produce massive sequential data from various sources, such as data traffic flows, and the evolution of mobile network subscribers' trajectories and application latencies. Exploring the RNN family is promising to enhance the analysis of time series data in mobile networks.

F. Generative Adversarial Network

The Generative Adversarial Network (GAN) is a framework that trains generative models using the following adversarial process. It simultaneously trains two models: a generative one $\mathcal{G}$ that seeks to approximate the target data distribution from training data, and a discriminative model $\mathcal{D}$ that estimates the probability that a sample comes from the real training data rather than the output of $\mathcal{G}$ [92]. Both of $\mathcal{G}$ and $\mathcal{D}$ are normally neural networks. The training procedure for $\mathcal{G}$ aims to maximize the probability of $\mathcal{D}$ making a mistake. The overall objective is solving the following minimax problem [92]:

$$\min_{\mathcal{G}} \max_{\mathcal{D}} \mathbb{E}_{x \sim P_r(x)}[\log \mathcal{D}(x)] + \mathbb{E}_{z \sim P_n(z)}[\log(1 - \mathcal{D}(\mathcal{G}(z)))].$$

Algorithm 1 Typical GAN Training Algorithm

Inputs:
Batch size m.
The number of steps for the discriminator K.
Learning rate λ and an optimizer Opt$(\cdot)$
Noise vector $z \sim p_g(z)$.
Target data set $x \sim p_{data}(x)$.
Initialise:
Generative and discriminative models, $\mathcal{G}$ and $\mathcal{D}$, parameterized by $\Theta_{\mathcal{G}}$ and $\Theta_{\mathcal{D}}$.
while $\Theta_{\mathcal{G}}$ and $\Theta_{\mathcal{D}}$ have not converged **do**
  for $k = 1$ to K **do**
    Sample m-element noise vector $\{z^{(1)}, \cdots, z^{(m)}\}$ from the noise prior $p_g(z)$
    Sample m data points $\{x^{(1)}, \cdots, x^{(m)}\}$ from the target data distribution $p_{data}(x)$
    $g_{\mathcal{D}} \leftarrow \Delta_{\Theta_{\mathcal{D}}}[\frac{1}{m}\sum_{i=1}^{m} \log \mathcal{D}(x^{(i)}) + \frac{1}{m}\sum_{i=1}^{m} \log(1 - \mathcal{D}(\mathcal{G}(z^{(i)})))]$.
    $\Theta_{\mathcal{D}} \leftarrow \Theta_{\mathcal{D}} + \lambda \cdot \mathrm{Opt}(\Theta_{\mathcal{D}}, g_{\mathcal{D}})$.
  end for
  Sample m-element noise vector $\{z^{(1)}, \cdots, z^{(m)}\}$ from the noise prior $p_g(z)$
  $g_{\mathcal{G}} \leftarrow \frac{1}{m}\sum_{i=1}^{m} \log(1 - \mathcal{D}(\mathcal{G}(z^{(i)})))$
  $\Theta_{\mathcal{G}} \leftarrow \Theta_{\mathcal{G}} - \lambda \cdot \mathrm{Opt}(\Theta_{\mathcal{G}}, g_{\mathcal{G}})$.
end while

Algorithm 1 shows the typical routine used to train a simple GAN. Both the generators and the discriminator are trained iteratively while fixing the other one. Finally $\mathcal{G}$ can produce data close to a target distribution (the same with training examples), if the model converges. We show the overall structure of a GAN in Fig. 6(g). In practice, the generator $\mathcal{G}$ takes a noise vector z as input, and generates an output $\mathcal{G}(z)$ that follows the target distribution. $\mathcal{D}$ will try to discriminate whether $\mathcal{G}(z)$ is a real sample or an artifact [199]. This effectively constructs a dynamic game, for which a Nash Equilibrium is reached if both $\mathcal{G}$ and $\mathcal{D}$ become optimal, and $\mathcal{G}$ can produce lifelike data that $\mathcal{D}$ can no longer discriminate, i.e., $\mathcal{D}(\mathcal{G}(z)) = 0.5, \forall z$.

The training process of traditional GANs is highly sensitive to model structures, learning rates, and other hyper-parameters. Researchers are usually required to employ numerous ad hoc 'tricks' to achieve convergence and improve the fidelity of data generated. There exist several solutions for mitigating this problem, e.g., Wasserstein Generative Adversarial Network (WGAN) [80], Loss-Sensitive Generative Adversarial Network (LS-GAN) [169] and BigGAN [170], but research on the theory of GANs remains shallow. Recent work confirms that GANs can promote the performance of some supervised tasks (e.g., super-resolution [200], object detection [201], and face completion [202]) by minimizing the divergence between

inferred and real data distributions. Exploiting the unsupervised learning abilities of GANs is promising in terms of generating synthetic mobile data for simulations, or assisting specific supervised tasks in mobile network applications. This becomes more important in tasks where appropriate datasets are lacking, given that operators are generally reluctant to share their network data.

G. Deep Reinforcement Learning

Deep Reinforcement Learning (DRL) refers to a set of methods that approximate value functions (deep Q learning) or policy functions (policy gradient method) through deep neural networks. An agent (neural network) continuously interacts with an environment and receives reward signals as feedback. The agent selects an action at each step, which will change the state of the environment. The training goal of the neural network is to optimize its parameters, such that it can select actions that potentially lead to the best future return. We illustrate this principle in Fig. 6(h). DRL is well-suited to problems that have a huge number of possible states (i.e., environments are high-dimensional). Representative DRL methods include Deep Q-Networks (DQNs) [19], deep policy gradient methods [171], Asynchronous Advantage Actor-Critic [79], Rainbow [172] and Distributed Proximal Policy Optimization (DPPO) [173]. These perform remarkably in AI gaming (e.g., Gym[20]), robotics, and autonomous driving [204]–[207], and have made inspiring deep learning breakthroughs recently.

In particular, the DQN [19] is first proposed by DeepMind to play Atari video games. However, traditional DQN requires several important adjustments to work well. The A3C [79] employs an actor-critic mechanism, where the actor selects the action given the state of the environment, and the critic estimates the value given the state and the action, then delivers feedback to the actor. The A3C deploys different actors and critics on different threads of a CPU to break the dependency of data. This significantly improves training convergence, enabling fast training of DRL agents on CPUs. Rainbow [172] combines different variants of DQNs, and discovers that these are complementary to some extent. This insight improved performance in many Atari games. To solve the step size problem in policy gradients methods, Schulman *et al.* [173] propose a Distributed Proximal Policy Optimization (DPPO) method to constrain the update step of new policies, and implement this on multi-threaded CPUs in a distributed manner. Based on this method, an agent developed by OpenAI defeated a human expert in Dota2 team in a 5v5 match.[21] Recent DRL method also conquers a more complex real-time multi-agent game StarCraft II.[22] In [208], DeepMind develops an game agent based on supervised learning and DRL named AlphaStar, beating one of the world's strongest professional StarCraft players by 5-0.

[20]Gym is a toolkit for developing and comparing reinforcement learning algorithms. It supports teaching agents everything from walking to playing games like Pong or Pinball. In combination with the NS3 simulator Gym becomes applicable to networking research [203] https://gym.openai.com/.

[21]Dota2 is a popular multiplayer online battle arena video game.

[22]StarCraft II is a popular multi-agent real-time strategy game.

Many mobile networking problems can be formulated as Markov Decision Processes (MDPs), where reinforcement learning can play an important role (e.g., base station on-off switching strategies [209], routing [210], and adaptive tracking control [211]). Some of these problems nevertheless involve high-dimensional inputs, which limits the applicability of traditional reinforcement learning algorithms. DRL techniques broaden the ability of traditional reinforcement learning algorithms to handle high dimensionality, in scenarios previously considered intractable. Employing DRL is thus promising to address network management and control problems under complex, changeable, and heterogeneous mobile environments. We further discuss this potential in Section VIII.

VI. Deep Learning Driven Mobile and Wireless Networks

Deep learning has a wide range of applications in mobile and wireless networks. In what follows, we present the most important research contributions across different mobile networking areas and compare their design and principles. In particular, we first discuss a key prerequisite, that of mobile big data, then organize the review of relevant works into nine subsections, focusing on specific domains where deep learning has made advances. Specifically,

1) *Deep Learning Driven Network-Level Mobile Data Analysis* focuses on deep learning applications built on mobile big data collected within the network, including network prediction, traffic classification, and Call Detail Record (CDR) mining.
2) *Deep Learning Driven App-Level Mobile Data Analysis* shifts the attention towards mobile data analytics on edge devices.
3) *Deep Learning Driven User Mobility Analysis* sheds light on the benefits of employing deep neural networks to understand the movement patterns of mobile users, either at group or individual levels.
4) *Deep Learning Driven User Localization* reviews literature that employ deep neural networks to localize users in indoor or outdoor environments, based on different signals received from mobile devices or wireless channels.
5) *Deep Learning Driven Wireless Sensor Networks* discusses important work on deep learning applications in WSNs from four different perspectives, namely centralized vs. decentralized sensing, WSN data analysis, WSN localization and other applications.
6) *Deep Learning Driven Network Control* investigate the usage of deep reinforcement learning and deep imitation learning on network optimization, routing, scheduling, resource allocation, and radio control.
7) *Deep Learning Driven Network Security* presents work that leverages deep learning to improve network security, which we cluster by focus as infrastructure, software, and privacy related.
8) *Deep Learning Driven Signal Processing* scrutinizes physical layer aspects that benefit from deep learning and reviews relevant work on signal processing.

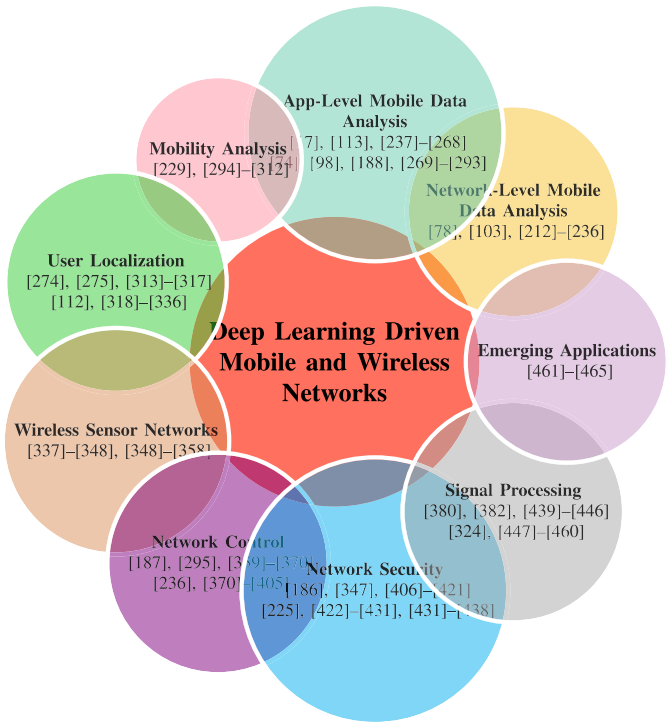


Fig. 7. Classification of the literature reviewed in Section VI.

TABLE X
THE TAXONOMY OF MOBILE BIG DATA

Mobile data	Source	Information
Network-level data	Infrastructure	Infrastructure locations, capability, equipment holders, etc
	Performance indicators	Data traffic, end-to-end delay, QoE, jitter, etc.
	Call detail records (CDR)	Session start and end times, type, sender and receiver, etc.
	Radio information	Signal power, frequency, spectrum, modulation etc.
App-level data	Device	Device type, usage, Media Access Control (MAC) address, etc.
	Profile	User settings, personal information, etc
	Sensors	Mobility, temperature, magnetic field, movement, etc
	Application	Picture, video, voice, health condition, preference, etc.
	System log	Software and hardware failure logs, etc.

9) *Emerging Deep Learning Driven Mobile Network Application* warps up this section, presenting other interesting deep learning applications in mobile networking.

For each domain, we summarize work broadly in tabular form, providing readers with a general picture of individual topics. Most important works in each domain are discussed in more details in text. Lessons learned are also discussed at the end of each subsection. We give a diagramatic view of the topics dealt with by the literature reviewed in this section in Fig. 7.

A. Mobile Big Data as a Prerequisite

The development of mobile technology (e.g., smartphones, augmented reality, etc.) are forcing mobile operators to evolve mobile network infrastructures. As a consequence, both the cloud and edge side of mobile networks are becoming increasingly sophisticated to cater for users who produce and consume huge amounts of mobile data daily. These data can be either generated by the sensors of mobile devices that record individual user behaviors, or from the mobile network infrastructure, which reflects dynamics in urban environments. Appropriately mining these data can benefit multidisciplinary research fields and the industry in areas such mobile network management, social analysis, public transportation, personal services provision, and so on [36]. Network operators, however, could become overwhelmed when managing and analyzing massive amounts of heterogeneous mobile data [466]. Deep learning is probably the most powerful methodology that can overcoming this burden. We begin therefore by introducing characteristics of mobile big data, then present a holistic review of deep learning driven mobile data analysis research.

Yazti and Krishnaswamy [467] propose to categorize mobile data into two groups, namely *network-level* data and *app-level* data. The key difference between them is that in the former data is usually collected by the edge mobile devices, while in the latter obtained throughout network infrastructure. We summarize these two types of data and their information comprised in Table X. Before delving into mobile data analytics, we illustrate the typical data collection process in Figure 9.

Network-level mobile data generated by the networking infrastructure not only deliver a global view of mobile network performance (e.g., throughput, end-to-end delay, jitter, etc.), but also log individual session times, communication types, sender and receiver information, through Call Detail Records (CDRs). Network-level data usually exhibit significant spatio-temporal variations resulting from users' behaviors [468], which can be utilized for network diagnosis and management, user mobility analysis and public transportation planning [218]. Some network-level data (e.g., mobile traffic snapshots) can be viewed as pictures taken by 'panoramic cameras', which provide a city-scale sensing system for urban sensing.

On the other hand, *App-level* data is directly recorded by sensors or mobile applications installed in various mobile devices. These data are frequently collected through crowdsourcing schemes from heterogeneous sources, such as Global Positioning Systems (GPS), mobile cameras and video recorders, and portable medical monitors. Mobile devices act as sensor hubs, which are responsible for data gathering and preprocessing, and subsequently distributing such data to specific locations, as required [36]. We show a typical app-level data processing system in Fig. 8. App-level mobile data is generated and collected by a Software Development Kit (SDK)

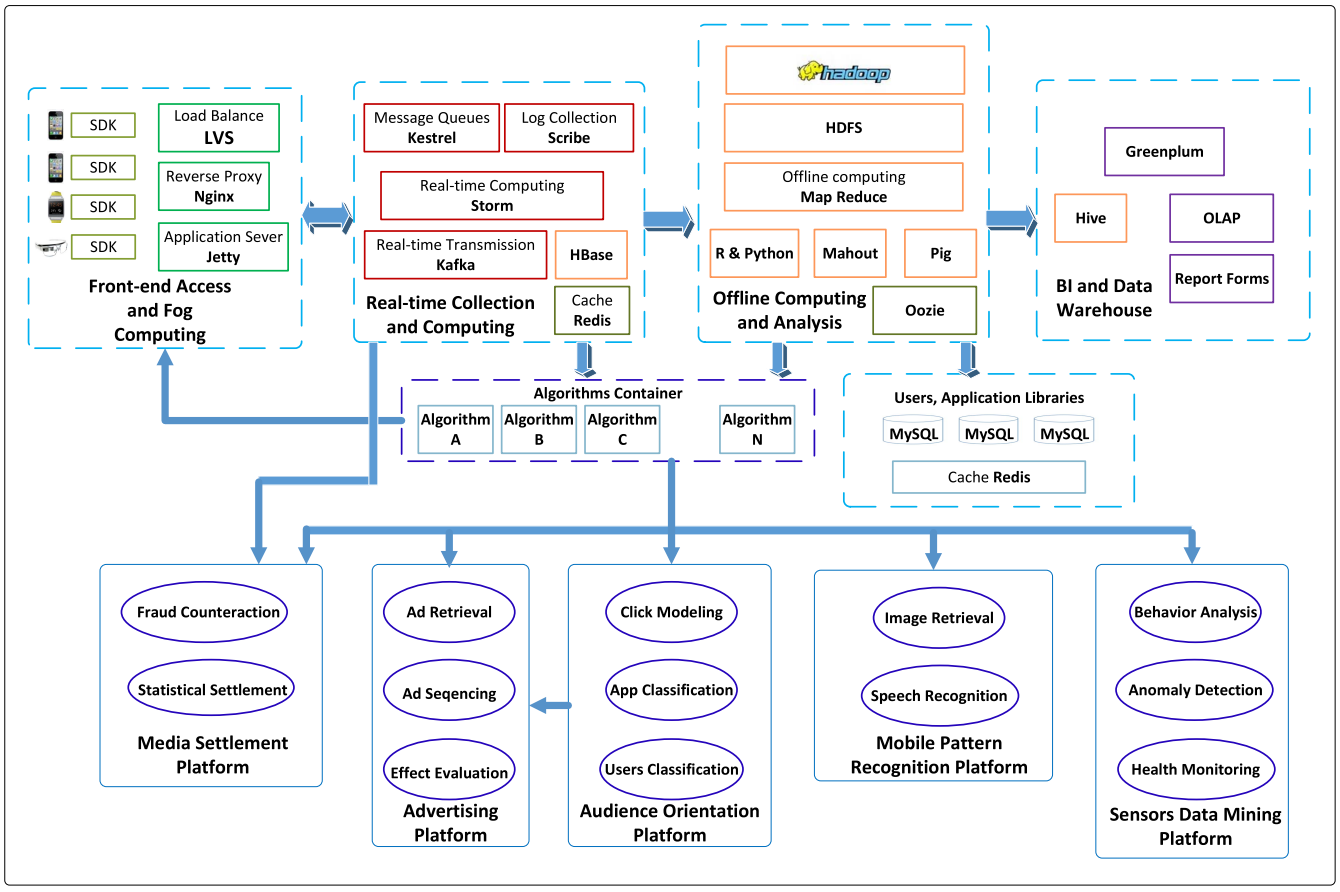


Fig. 8. Typical pipeline of an app-level mobile data processing system.

installed on mobile devices. After pre-processing and load-balancing (e.g., Nginx[23]), Such data is subsequently processed by real-time collection and computing services (e.g., Storm,[24] Kafka,[25] HBase,[26] Redis,[27] etc.) as required. Further offline storage and computing with mobile data can be performed with various tools, such as Hadoop Distribute File System (HDFS),[28] Python, Mahout,[29] Pig,[30] or Oozie.[31] The raw data and analysis results will be further transferred to databases (e.g., MySQL[32]) Business Intelligence – BI (e.g., Online Analytical Processing – OLAP[33]), and data warehousing (e.g., Hive[34]). Among these, the algorithms container is the core of the entire system as it connects to front-end access and fog computing, real-time collection and computing, and offline computing and analysis modules, while it links directly to mobile applications, such as mobile healthcare, pattern recognition, and advertising platforms. Deep learning logic can be placed within the algorithms container.

App-level data may directly or indirectly reflect users' behaviors, such as mobility, preferences, and social links [61]. Analyzing app-level data from individuals can help reconstructing one's personality and preferences, which can be used in recommender systems and users targeted advertising. Some of these data comprise explicit information about individuals' identities. Inappropriate sharing and use can raise significant privacy issues. Therefore, extracting useful patterns from multi-modal sensing devices without compromising user's privacy remains a challenging endeavor.

Compared to traditional data analysis techniques, deep learning embraces several unique features to address the aforementioned challenges [17]. Namely:

1) Deep learning achieves remarkable performance in various data analysis tasks, on both structured and

[23]Nginx is an HTTP and reverse proxy server, a mail proxy server, and a generic TCP/UDP proxy server, https://nginx.org/en/.

[24]Storm is a free and open-source distributed real-time computation system, http://storm.apache.org/.

[25]Kafka is used for building real-time data pipelines and streaming apps, https://kafka.apache.org/.

[26]Apache HBase is the Hadoop database, a distributed, scalable, big data store, https://hbase.apache.org/.

[27]Redis is an open source, in-memory data structure store, used as a database, cache and message broker, https://redis.io/.

[28]The Hadoop Distributed File System (HDFS) is a distributed file system designed to run on commodity hardware, https://hadoop.apache.org/docs/r1.2.1/hdfs_design.html.

[29]Apache Mahout is a distributed linear algebra framework, https://mahout.apache.org/.

[30]Apache Pig is a high-level platform for creating programs that run on Apache Hadoop, https://pig.apache.org/.

[31]Oozie is a workflow scheduler system to manage Apache Hadoop jobs, http://oozie.apache.org/.

[32]MySQL is the open source database, https://www.oracle.com/technetwork/database/mysql/index.html.

[33]OLAP is an approach to answer multi-dimensional analytical queries swiftly in computing, and is part of the broader category of business intelligence.

[34]The Apache Hive is a data warehouse software, https://hive.apache.org/.

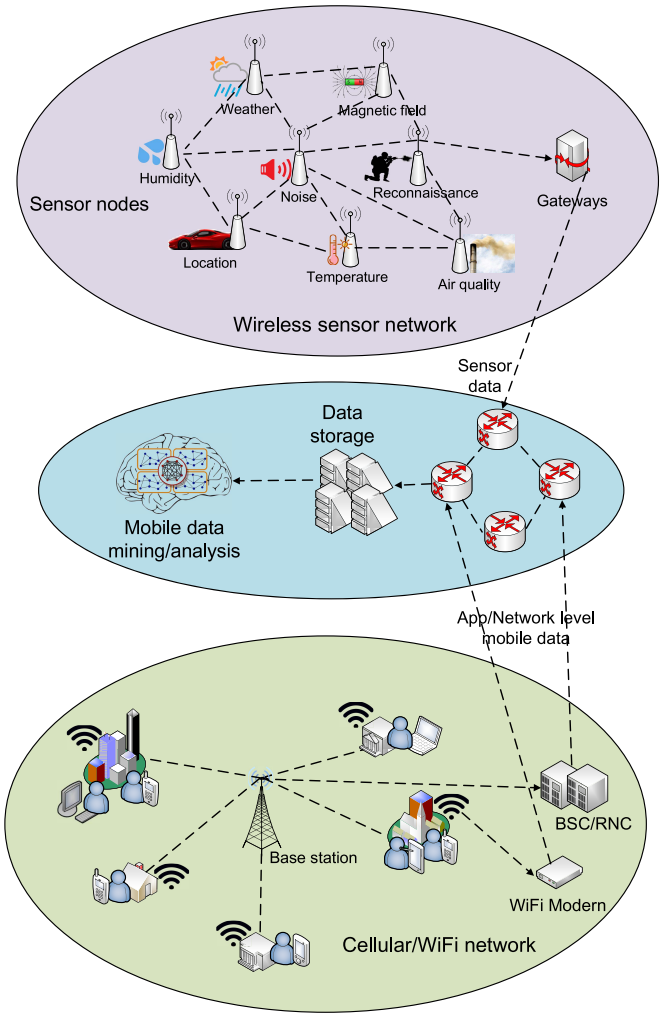


Fig. 9. Illustration of the mobile data collection process in cellular, WiFi and wireless sensor networks. BSC: Base Station Controller; RNC: Radio Network Controller.

unstructured data. Some types of mobile data can be represented as image-like (e.g., [218]) or sequential data [226].

2) Deep learning performs remarkably well in feature extraction from raw data. This saves tremendous effort of hand-crafted feature engineering, which allows spending more time on model design and less on sorting through the data itself.
3) Deep learning offers excellent tools (e.g., RBM, AE, GAN) for handing unlabeled data, which is common in mobile network logs.
4) Multi-modal deep learning allows to learn features over multiple modalities [469], which makes it powerful in modeling with data collected from heterogeneous sensors and data sources.

These advantages make deep learning as a powerful tool for mobile data analysis.

B. Deep Learning Driven Network-Level Mobile Data Analysis

Network-level mobile data refers broadly to logs recorded by Internet service providers, including infrastructure metadata, network performance indicators and call detail records (CDRs) (see Table XI). The recent remarkable success of deep learning ignites global interests in exploiting this methodology for mobile network-level data analysis, so as to optimize mobile networks configurations, thereby improving end-uses' QoE. These work can be categorized into four types: network state prediction, network traffic classification, CDR mining and radio analysis. In what follows, we review work in these directions, which we first summarize and compare in Table XI.

Network State Prediction refers to inferring mobile network traffic or performance indicators, given historical measurements or related data. Pierucci and Micheli [212] investigate the relationship between key objective metrics and QoE. They employ MLPs to predict users' QoE in mobile communications, based on average user throughput, number of active users in a cells, average data volume per user, and channel quality indicators, demonstrating high prediction accuracy. Network traffic forecasting is another field where deep learning is gaining importance. By leveraging sparse coding and max-pooling, Gwon and Kung [213] develop a semi-supervised deep learning model to classify received frame/packet patterns and infer the original properties of flows in a WiFi network. Their proposal demonstrates superior performance over traditional ML techniques. Nie *et al.* [214] investigate the traffic demand patterns in wireless mesh network. They design a DBN along with Gaussian models to precisely estimate traffic distributions.

In addition to the above, several researchers employ deep learning to forecast mobile traffic at city scale, by considering spatio-temporal correlations of geographic mobile traffic measurements. We illustrate the underlying principle in Fig. 10. Wang *et al.*[216] propose to use an AE-based architecture and LSTMs to model spatial and temporal correlations of mobile traffic distribution, respectively. In particular, the authors use a global and multiple local stacked AEs for spatial feature extraction, dimension reduction and training parallelism. Compressed representations extracted are subsequently processed by LSTMs, to perform final forecasting. Experiments with a real-world dataset demonstrate superior performance over SVM and the Autoregressive Integrated Moving Average (ARIMA) model. The work in [217] extends mobile traffic forecasting to long time frames. The authors combine ConvLSTMs and 3D CNNs to construct spatio-temporal neural networks that capture the complex spatio-temporal features at city scale. They further introduce a fine-tuning scheme and lightweight approach to blend predictions with historical means, which significantly extends the length of reliable prediction steps. Deep learning was also employed in [78], [219], [235], and [470], where the authors employ CNNs and LSTMs to perform mobile traffic forecasting. By effectively extracting spatio-temporal features, their proposals gain significantly higher accuracy than traditional approaches, such as ARIMA. Wang *et al.* [103] represent spatio-temporal dependencies in mobile traffic using graphs, and learn such dependencies using Graph Neural Networks. Beyond the accurate inference achieved in their study, this work also demonstrates potential for precise social events inference.

TABLE XI
A SUMMARY OF WORK ON NETWORK-LEVEL MOBILE DATA ANALYSIS

Domain	Reference	Applications	Model	Optimizer	Key contribution
Network prediction	Pierucci and Micheli [212]	QoE prediction	MLP	Unknown	Uses NNs to correlate Quality of Service parameters and QoE estimations.
	Gwon and Kung [213]	Inferring Wi-Fi flow patterns	Sparse coding + Max pooling	SGD	Semi-supervised learning.
	Nie *et al.* [214]	Wireless mesh network traffic prediction	DBN + Gaussian models	SGD	Considers both long-term dependency and short-term fluctuations.
	Moyo and Sibanda [215]	TCP/IP traffic prediction	MLP	Unknown	Investigates the impact of learning in traffic forecasting.
	Wang *et al.* [216]	Mobile traffic forecasting	AE + LSTM	SGD	Uses an AE to model spatial correlations and an LSTM to model temporal correlation
	Zhang and Patras [217]	Long-term mobile traffic forecasting	ConvLSTM + 3D-CNN	Adam	Combines 3D-CNNs and ConvLSTMs to perform long-term forecasting
	Zhang *et al.* [218]	Mobile traffic super-resolution	CNN + GAN	Adam	Introduces the MTSR concept and applies image processing techniques for mobile traffic analysis
	Huang *et al.* [219]	Mobile traffic forecasting	LSTM + 3D-CNN	Unknown	Combines CNNs and RNNs to extract geographical and temporal features from mobile traffic.
	Zhang *et al.* [220]	Cellular traffic prediction	Densely connected CNN	Adam	Uses separate CNNs to model closeness and periods in temporal dependency.
	Chen *et al.* [78]	Cloud RAN optimization	Multivariate LSTM	Unknown	Uses mobile traffic forecasting to aid cloud radio access network optimization
	Navabi *et al.* [221]	Wireless WiFi channel feature prediction	MLP	SGD	Infers non-observable channel information from observable features.
	Feng *et al.* [235]	Mobile cellular traffic prediction	LSTM	Adam	Extracts spatial and temporal dependencies using separate modules.
	Alawe *et al.* [470]	Mobile traffic load forecasting	MLP, LSTM	Unknown	Employs traffic forecasting to improve 5G network scalability.
	Wang *et al.* [103]	Cellular traffic prediction	Graph neural networks	Pineda algorithm	Represents spatio-temporal dependency via graphs and first work employing Graph neural networks for traffic forecasting.
	Fang *et al.* [232]	Mobile demand forecasting	Graph CNN, LSTM, Spatio-temporal graph ConvLSTM	Unknown	Modelling the spatial correlations between cells using a dependency graph.
	Luo *et al.* [233]	Channel state information prediction	CNN and LSTM	RMSprop	Employing a two-stage offline-online training scheme to improve the stability of framework.
Traffic classification	Wang [222]	Traffic classification	MLP, stacked AE	Unknown	Performs feature learning, protocol identification and anomalous protocol detection simultaneously.
	Wang *et al.* [223]	Encrypted traffic classification	CNN	SGD	Employs an end-to-end deep learning approach to perform encrypted traffic classification.
	Lotfollahi *et al.* [224]	Encrypted traffic classification	CNN	Adam	Can perform both traffic characterization and application identification.
	Wang *et al.* [225]	Malware traffic classification	CNN	SGD	First work to use representation learning for malware classification from raw traffic.
	Aceto *et al.* [471]	Mobile encrypted traffic classification	MLP, CNN, LSTM	SGD, Adam	Comprehensive evaluations of different NN architectures and excellent performance.
	Li *et al.* [234]	Network traffic classification	Bayesian auto-encoder	SGD	Applying Bayesian probability theory to obtain the a posteriori distribution of model parameters.
CDR mining	Liang *et al.* [226]	Metro density prediction	RNN	SGD	Employs geo-spatial data processing, a weight-sharing RNN and parallel stream analytic programming.
	Felbo *et al.* [227]	Demographics prediction	CNN	Adam	Exploits the temporal correlation inherent to mobile phone metadata.
	Chen *et al.* [228]	Tourists' next visit location prediction	MLP, RNN	Scaled conjugate gradient descent	LSTM that performs significantly better than other ML approaches.
	Lin *et al.* [229]	Human activity chains generation	Input-Output HMM + LSTM	Adam	First work that uses an RNN to generate human activity chains.
Others	Xu *et al.* [230]	Wi-Fi hotpot classification	CNN	Unknown	Combining deep learning with frequency analysis.
	Meng *et al.* [231]	QoE-driven big data analysis	CNN	SGD	Investigates trade-off between accuracy of high-dimensional big data analysis and model training speed.

More recently, Zhang *et al.* [218] propose an original Mobile Traffic Super-Resolution (MTSR) technique to infer network-wide fine-grained mobile traffic consumption given coarse-grained counterparts obtained by probing, thereby reducing traffic measurement overheads. We illustrate the principle of MTSR in Fig. 11. Inspired by image super-resolution

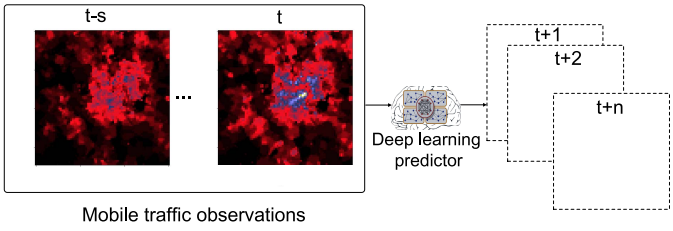


Fig. 10. The underlying principle of city-scale mobile traffic forecasting. The deep learning predictor takes as input a sequence of mobile traffic measurements in a region (snapshots $t - s$ to t), and forecasts how much mobile traffic will be consumed in the same areas in the future $t + 1$ to $t + n$ instances.

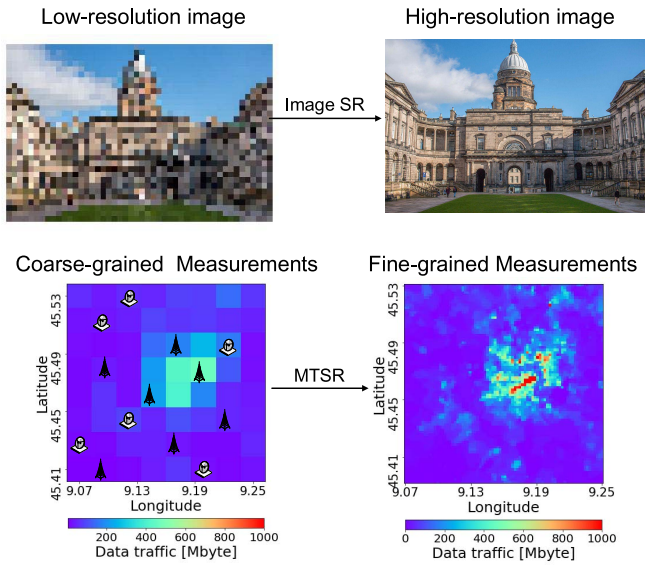


Fig. 11. Illustration of the image super-resolution (SR) principle (above) and the mobile traffic super-resolution (MTSR) technique (below). Figure adapted from [218].

techniques, they design a dedicated CNN with multiple skip connections between layers, named deep zipper network, along with a Generative Adversarial Network (GAN) to perform precise MTSR and improve the fidelity of inferred traffic snapshots. Experiments with a real-world dataset show that this architecture can improve the granularity of mobile traffic measurements over a city by up to 100×, while significantly outperforming other interpolation techniques.

Traffic Classification is aimed at identifying specific applications or protocols among the traffic in networks. Wang [222] recognizes the powerful feature learning ability of deep neural networks and uses a deep AE to identify protocols in a TCP flow dataset, achieving excellent precision and recall rates. Work in [223] proposes to use a 1D CNN for encrypted traffic classification. The authors suggest that this structure works well for modeling sequential data and has lower complexity, thus being promising in addressing the traffic classification problem. Similarly, Lotfollahi *et al.* [224] present Deep Packet, which is based on a CNN, for encrypted traffic classification. Their framework reduces the amount of hand-crafted feature engineering and achieves great accuracy. An improved stacked AE is employed in [234], where Li *et al.* incorporate Bayesian methods into AEs to enhance the inference accuracy in network traffic classification. More recently, Aceto *et al.* [471] employ MLPs, CNNs, and LSTMs to perform encrypted mobile traffic classification, arguing that deep NNs can automatically extract complex features present in mobile traffic. As reflected by their results, deep learning based solutions obtain superior accuracy over RFs in classifying Android, IOS and Facebook traffic. CNNs have also been used to identify malware traffic, where work in [225] regards traffic data as images and unusual patterns that malware traffic exhibit are classified by representation learning. Similar work on mobile malware detection will be further discussed in Section VI-H.

CDR Mining involves extracting knowledge from specific instances of telecommunication transactions such as phone number, cell ID, session start/end time, traffic consumption, etc. Using deep learning to mine useful information from CDR data can serve a variety of functions. For example, Liang *et al.* [226] propose Mercury to estimate metro density from streaming CDR data, using RNNs. They take the trajectory of a mobile phone user as a sequence of locations; RNN-based models work well in handling such sequential data. Likewise, Felbo *et al.* [227] use CDR data to study demographics. They employ a CNN to predict the age and gender of mobile users, demonstrating the superior accuracy of these structures over other ML tools. More recently, Chen *et al.* [228] compare different ML models to predict tourists' next locations of visit by analyzing CDR data. Their experiments suggest that RNN-based predictors significantly outperform traditional ML methods, including Naive Bayes, SVM, RF, and MLP.

Lessons Learned: Network-level mobile data, such as mobile traffic, usually involves essential spatio-temporal correlations. These correlations can be effectively learned by CNNs and RNNs, as they are specialized in modeling spatial and temporal data (e.g., images, traffic series). An important observation is that large-scale mobile network traffic can be processed as sequential snapshots, as suggested in [217] and [218], which resemble images and videos. Therefore, potential exists to exploit image processing techniques for network-level analysis. Techniques previously used for imaging usually, however, cannot be directly employed with mobile data. Efforts must be made to adapt them to the particularities of the mobile networking domain. We expand on this future research direction in Section VIII-B.

On the other hand, although deep learning brings precision in network-level mobile data analysis, making causal inference remains challenging, due to limited model interpretability. For example, a NN may predict there will be a traffic surge in a certain region in the near future, but it is hard to explain why this will happen and what triggers such a surge. Additional efforts are required to enable explanation and confident decision making. At this stage, the community should rather use deep learning algorithms as intelligent assistants that can make accurate inferences and reduce human effort, instead of relying exclusively on these.

C. Deep Learning Driven App-Level Mobile Data Analysis

Triggered by the increasing popularity of Internet of Things (IoT), current mobile devices bundle increasing numbers of

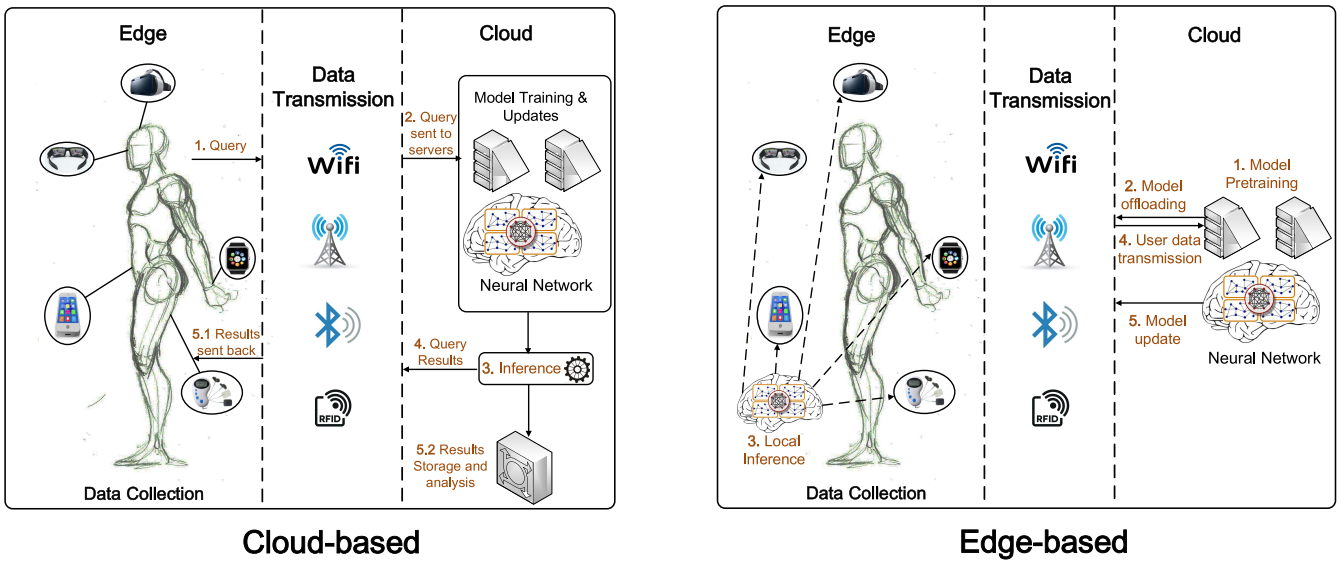


Fig. 12. Illustration of two deployment approaches for app-level mobile data analysis, namely cloud-based (left) and edge-based (right). The cloud-based approach makes inference on clouds and send results to edge devices. On the contrary, the edge-based approach deploys models on edge devices which can make local inference.

applications and sensors that can collect massive amounts of app-level mobile data [472]. Employing artificial intelligence to extract useful information from these data can extend the capability of devices [75], [473], [474], thus greatly benefiting users themselves, mobile operators, and indirectly device manufacturers. Analysis of mobile data therefore becomes an important and popular research direction in the mobile networking domain. Nonetheless, mobile devices usually operate in noisy, uncertain and unstable environments, where their users move fast and change their location and activity contexts frequently. As a result, app-level mobile data analysis becomes difficult for traditional machine learning tools, which performs relatively poorly. Advanced deep learning practices provide a powerful solution for app-level data mining, as they demonstrate better precision and higher robustness in IoT applications [475].

There exist two approaches to app-level mobile data analysis, namely *(i)* cloud-based computing and *(ii)* edge-based computing. We illustrate the difference between these scenarios in Fig. 12. As shown in the left part of the figure, the cloud-based computing treats mobile devices as data collectors and messengers that constantly send data to cloud servers, via local points of access with limited data preprocessing capabilities. This scenario typically includes the following steps: *(i)* users query on/interact with local mobile devices; *(ii)* queries are transmitted to severs in the cloud; *(iii)* servers gather the data received for model training and inference; *(iv)* query results are subsequently sent back to each device, or stored and analyzed without further dissemination, depending on specific application requirements. The drawback of this scenario is that constantly sending and receiving messages to/from servers over the Internet introduces overhead and may result in severe latency. In contrast, in the edge-based computing scenario pre-trained models are offloaded from the cloud to individual mobile devices, such that they can make inferences locally. As illustrated in the right part of Fig. 12, this scenario typically consists of the following: *(i)* servers use offline datasets to pertrain a model; *(ii)* the pre-trained model is offloaded to edge devices; *(iii)* mobile devices perform inferences locally using the model; *(iv)* cloud servers accept data from local devices; *(v)* the model is updated using these data whenever necessary. While this scenario requires less interactions with the cloud, its applicability is limited by the computing and battery capabilities of edge hardware. Therefore, it can only support tasks that require light computations.

Many researchers employ deep learning for app-level mobile data analysis. We group the works reviewed according to their application domains, namely mobile healthcare, mobile pattern recognition, and mobile Natural Language Processing (NLP) and Automatic Speech Recognition (ASR). Table XII gives a high-level summary of existing research efforts and we discuss representative work next.

Mobile Health: There is an increasing variety of wearable health monitoring devices being introduced to the market. By incorporating medical sensors, these devices can capture the physical conditions of their carriers and provide real-time feedback (e.g., heart rate, blood pressure, breath status etc.), or trigger alarms to remind users of taking medical actions [476].

Liu and Du [237] design a deep learning-driven MobiEar to aid deaf people's awareness of emergencies. Their proposal accepts acoustic signals as input, allowing users to register different acoustic events of interest. MobiEar operates efficiently on smart phones and only requires infrequent communications with servers for updates. Likewise, Sicong *et al.* [238] develop a UbiEar, which is operated on the Android platform to assist hard-to-hear sufferers in recognizing acoustic events, without requiring location information. Their design adopts

[34]Human profile source: https://lekeart.deviantart.com/art/male-body-profile-251793336.

TABLE XII
A SUMMARY OF WORKS ON APP-LEVEL MOBILE DATA ANALYSIS

Subject	Reference	Application	Deployment	Model
Mobile Healthcare	Liu and Du [237]	Mobile ear	Edge-based	CNN
	Liu *at al.* [238]	Mobile ear	Edge-based	CNN
	Jindal [239]	Heart rate prediction	Cloud-based	DBN
	Kim *et al.* [240]	Cytopathology classification	Cloud-based	CNN
	Sathyanarayana *et al.* [241]	Sleep quality prediction	Cloud-based	MLP, CNN, LSTM
	Li and Trocan [242]	Health conditions analysis	Cloud-based	Stacked AE
	Hosseini *et al.* [243]	Epileptogenicity localisation	Cloud-based	CNN
	Stamate *et al.* [244]	Parkinson's symptoms management	Cloud-based	MLP
	Quisel *et al.* [245]	Mobile health data analysis	Cloud-based	CNN, RNN
	Khan *et al.* [246]	Respiration surveillance	Cloud-based	CNN
Mobile Pattern Recognition	Li *et al.* [247]	Mobile object recognition	Edge-based	CNN
	Tobías *et al.* [248]	Mobile object recognition	Edge-based & Cloud based	CNN
	Pouladzadeh and Shirmohammadi [249]	Food recognition system	Cloud-based	CNN
	Tanno *et al.* [250]	Food recognition system	Edge-based	CNN
	Kuhad *et al.* [251]	Food recognition system	Cloud-based	MLP
	Teng and Yang [252]	Facial recognition	Cloud-based	CNN
	Wu *et al.* [293]	Mobile visual search	Edge-based	CNN
	Rao *et al.* [253]	Mobile augmented reality	Edge-based	CNN
	Ohara *et al.* [292]	WiFi-driven indoor change detection	Cloud-based	CNN,LSTM
	Zeng *et al.* [254]	Activity recognition	Cloud-based	CNN, RBM
	Almaslukh *et al.* [255]	Activity recognition	Cloud-based	AE
	Li *et al.* [256]	RFID-based activity recognition	Cloud-based	CNN
	Bhattacharya and Lane [257]	Smart watch-based activity recognition	Edge-based	RBM
	Antreas and Angelov [258]	Mobile surveillance system	Edge-based & Cloud based	CNN
	Ordóñez and Roggen [113]	Activity recognition	Cloud-based	ConvLSTM
	Wang *et al.* [259]	Gesture recognition	Edge-based	CNN, RNN
	Gao *et al.* [260]	Eating detection	Cloud-based	DBM, MLP
	Zhu *et al.* [261]	User energy expenditure estimation	Cloud-based	CNN, MLP
	Sundsøy *et al.* [262]	Individual income classification	Cloud-based	MLP
	Chen and Xue [263]	Activity recognition	Cloud-based	CNN
	Ha and Choi [264]	Activity recognition	Cloud-based	CNN
	Edel and Köppe [265]	Activity recognition	Edge-based	Binarized-LSTM
	Okita and Inoue [268]	Multiple overlapping activities recognition	Cloud-based	CNN+LSTM
	Alsheikh *et al.* [17]	Activity recognition using Apache Spark	Cloud-based	MLP
	Mittal *et al.* [269]	Garbage detection	Edge-based & Cloud based	CNN
	Seidenari *et al.* [270]	Artwork detection and retrieval	Edge-based	CNN
	Zeng *et al.* [271]	Mobile pill classification	Edge-based	CNN
	Lane and Georgiev [74]	Mobile activity recognition, emotion recognition and speaker identification	Edge-based	MLP
	Yao *et al.* [291]	Car tracking,heterogeneous human activity recognition and user identification	Edge-based	CNN, RNN
	Zou *et al.* [272]	IoT human activity recognition	Cloud-based	AE, CNN, LSTM
	Zeng [273]	Mobile object recognition	Edge-based	Unknown
	Katevas *et al.* [290]	Notification attendance prediction	Edge-based	RNN
	Radu *et al.* [188]	Activity recognition	Edge-based	RBM, CNN
	Wang *et al.* [274], [275]	Activity and gesture recognition	Cloud-based	Stacked AE
	Feng *et al.* [276]	Activity detection	Cloud-based	LSTM
	Cao *et al.* [277]	Mood detection	Cloud-based	GRU
	Ran *et al.* [278]	Object detection for AR applications.	Edge-based & cloud-based	CNN
	Zhao *et al.* [289]	Estimating 3D human skeleton from radio frequently signal	Cloud-based	CNN
Mobile NLP and ASR	Siri [279]	Speech synthesis	Edge-based	Mixture density networks
	McGraw *et al.* [280]	Personalised speech recognition	Edge-based	LSTM
	Prabhavalkar *et al.* [281]	Embedded speech recognition	Edge-based	LSTM
	Yoshioka *et al.* [282]	Mobile speech recognition	Cloud-based	CNN
	Ruan *et al.* [283]	Shifting from typing to speech	Cloud-based	Unknown
	Georgiev *et al.* [98]	Multi-task mobile audio sensing	Edge-based	MLP
Others	Ignatov *et al.* [284]	Mobile images quality enhancement	Cloud-based	CNN
	Lu *et al.* [285]	Information retrieval from videos in wireless network	Cloud-based	CNN
	Lee *et al.* [286]	Reducing distraction for smartwatch users	Cloud-based	MLP
	Vu *et al.* [287]	Transportation mode detection	Cloud-based	RNN
	Fang *et al.* [288]	Transportation mode detection	Cloud-based	MLP
	Xue *et al.* [266]	Mobile App classification	Cloud-based	AE, MLP, CNN, and LSTM
	Liu *et al.* [267]	Mobile motion sensor fingerprinting	Cloud-based	LSTM

a lightweight CNN architecture for inference acceleration and demonstrates comparable accuracy over traditional CNN models.

Hosseini *et al.* [243] design an edge computing system for health monitoring and treatment. They use CNNs to extract features from mobile sensor data, which plays an

important role in their epileptogenicity localization application. Stamate *et al.* [244] develop a mobile Android app called cloudUPDRS to manage Parkinson's symptoms. In their work, MLPs are employed to determine the acceptance of data collected by smart phones, to maintain high-quality data samples. The proposed method outperforms other ML methods such as GPs and RFs. Quisel *et al.* [245] suggest that deep learning can be effectively used for mobile health data analysis. They exploit CNNs and RNNs to classify lifestyle and environmental traits of volunteers. Their models demonstrate superior prediction accuracy over RFs and logistic regression, over six datasets.

As deep learning performs remarkably in medical data analysis [477], we expect more and more deep learning powered health care devices will emerge to improve physical monitoring and illness diagnosis.

Mobile Pattern Recognition: Recent advanced mobile devices offer people a portable intelligent assistant, which fosters a diverse set of applications that can classify surrounding objects (e.g., [247]–[249], and [252]) or users' behaviors (e.g., [113], [254], [257], [263], [264], [478], and [479]) based on patterns observed in the output of the mobile camera or other sensors. We review and compare recent works on mobile pattern recognition in this part.

Object classification in pictures taken by mobile devices is drawing increasing research interest. Li *et al.* [247] develop DeepCham as a mobile object recognition framework. Their architecture involves a crowd-sourcing labeling process, which aims to reduce the hand-labeling effort, and a collaborative training instance generation pipeline that is built for deployment on mobile devices. Evaluations of the prototype system suggest that this framework is efficient and effective in terms of training and inference. Tobías *et al.* [248] investigate the applicability of employing CNN schemes on mobile devices for objection recognition tasks. They conduct experiments on three different model deployment scenarios, i.e., on GPU, CPU, and respectively on mobile devices, with two benchmark datasets. The results obtained suggest that deep learning models can be efficiently embedded in mobile devices to perform real-time inference.

Mobile classifiers can also assist Virtual Reality (VR) applications. A CNN framework is proposed in [252] for facial expressions recognition when users are wearing head-mounted displays in the VR environment. Rao *et al.* [253] incorporate a deep learning object detector into a mobile augmented reality (AR) system. Their system achieves outstanding performance in detecting and enhancing geographic objects in outdoor environments. Further work focusing on mobile AR applications is introduced in [480], where Ran *et al.* characterize the tradeoffs between accuracy, latency, and energy efficiency of object detection.

Activity recognition is another interesting area that relies on data collected by mobile motion sensors [479], [481]. This refers to the ability to classify based on data collected via, e.g., video capture, accelerometer readings, motion – Passive Infra-Red (PIR) sensing, specific actions and activities that a human subject performs. Data collected will be delivered to servers for model training and the model will be subsequently deployed for domain-specific tasks.

Essential features of sensor data can be automatically extracted by neural networks. The first work in this space that is based on deep learning employs a CNN to capture local dependencies and preserve scale invariance in motion sensor data [254]. The authors evaluate their proposal on 3 offline datasets, demonstrating their proposal yields higher accuracy over statistical methods and Principal Components Analysis (PCA). Almaslukh *et al.* [255] employ a deep AE to perform human activity recognition by analyzing an offline smart phone dataset gathered from accelerometers and gyroscope sensors. Li *et al.* [256] consider different scenarios for activity recognition. In their implementation, Radio Frequency Identification (RFID) data is directly sent to a CNN model for recognizing human activities. While their mechanism achieves high accuracy in different applications, experiments suggest that the RFID-based method does not work well with metal objects or liquid containers.

Reference [257] exploits an RBM to predict human activities, given 7 types of sensor data collected by a smart watch. Experiments on prototype devices show that this approach can efficiently fulfill the recognition objective under tolerable power requirements. Ordóñez and Roggen [113] architect an advanced ConvLSTM to fuse data gathered from multiple sensors and perform activity recognition. By leveraging CNN and LSTM structures, ConvLSTMs can automatically compress spatio-temporal sensor data into low-dimensional representations, without heavy data post-processing effort. Wang *et al.* [259] exploit Google Soli to architect a mobile user-machine interaction platform. By analyzing radio frequency signals captured by millimeter-wave radars, their architecture is able to recognize 11 types of gestures with high accuracy. Their models are trained on the server side, and inferences are performed locally on mobile devices. More recently, Zhao *et al.* [289] design a 4D CNN framework (3D for the spatial dimension + 1D for the temporal dimension) to reconstruct human skeletons using radio frequency signals. This novel approach resembles virtual "X-ray", enabling to accurately estimate human poses, without requiring an actual camera.

Mobile NLP and ASR: Recent remarkable achievements obtained by deep learning in Natural Language Processing (NLP) and Automatic Speech Recognition (ASR) are also embraced by applications for mobile devices.

Powered by deep learning, the intelligent personal assistant Siri, developed by Apple, employs a deep mixture density networks [482] to fix typical robotic voice issues and synthesize more human-like voice [279]. An Android app released by Google supports mobile personalized speech recognition [280]; this quantizes the parameters in LSTM model compression, allowing the app to run on low-power mobile phones. Likewise, Prabhavalkar *et al.* [281] propose a mathematical RNN compression technique that reduces two thirds of an LSTM acoustic model size, while only compromising negligible accuracy. This allows building both memory- and energy-efficient ASR applications on mobile devices.

Yoshioka *et al.* [282] present a framework that incorporates a network-in-network architecture into a CNN model, which allows to perform ASR with mobile multi-microphone devices used in noisy environments. Mobile ASR can also accelerate text input on mobile devices, Ruan *et al.*'s [283] study showing that with the help of ASR, the input rates of English and Mandarin are 3.0 and 2.8 times faster over standard typing on keyboards. More recently, the applicability of deep learning to multi-task audio sensing is investigated in [98], where Georgiev *et al.* propose and evaluate a novel deep learning modelling and optimization framework tailored to embedded audio sensing tasks. To this end, they selectively share compressed representations between different tasks, which reduces training and data storage overhead, without significantly compromising accuracy of an individual task. The authors evaluate their framework on a memory-constrained smartphone performing four audio tasks (i.e., speaker identification, emotion recognition, stress detection, and ambient scene analysis). Experiments suggest this proposal can achieve high efficiency in terms of energy, runtime and memory, while maintaining excellent accuracy.

Other Applications: Deep learning also plays an important role in other applications that involve app-level data analysis. For instance, Ignatov *et al.* [284] show that deep learning can enhance the quality of pictures taken by mobile phones. By employing a CNN, they successfully improve the quality of images obtained by different mobile devices, to a digital single-lens reflex camera level. Lu *et al.* focus on video post-processing under wireless networks [285], where their framework exploits a customized AlexNet to answer queries about detected objects. This framework further involves an optimizer, which instructs mobile devices to offload videos, in order to reduce query response time.

Another interesting application is presented in [286], where Lee *et al.* show that deep learning can help smartwatch users reduce distraction by eliminating unnecessary notifications. Specifically, the authors use an 11-layer MLP to predict the importance of a notification. Fang *et al.* [288] exploit an MLP to extract features from high-dimensional and heterogeneous sensor data, including accelerometer, magnetometer, and gyroscope measurements. Their architecture achieves 95% accuracy in recognizing human transportation modes, i.e., still, walking, running, biking, and on vehicle.

Lessons Learned: App-level data is heterogeneous and generated from distributed mobile devices, and there is a trend to offload the inference process to these devices. However, due to computational and battery power limitations, models employed in the edge-based scenario are constrained to lightweight architectures, which are less suitable for complex tasks. Therefore, the trade-off between model complexity and accuracy should be carefully considered [67]. Numerous efforts were made towards tailoring deep learning to mobile devices, in order to make algorithms faster and less energy-consuming on embedded equipment. For example, model compression, pruning, and quantization are commonly used for this purpose. Mobile device manufacturers are also developing new software and hardware to support deep learning based applications. We will discuss this work in more detail in Section VII.

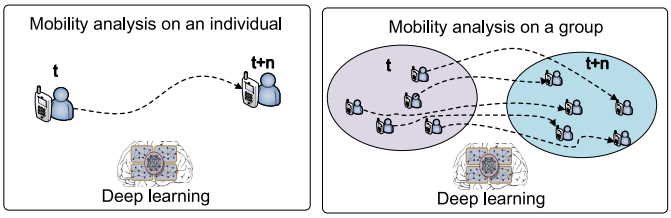


Fig. 13. Illustration of mobility analysis paradigms at individual (left) and group (right) levels.

At the same time, app-level data usually contains important users information and processing this poses significant privacy concerns. Although there have been efforts that commit to preserve user privacy, as we discuss in Section VI-H, research efforts in this direction are new, especially in terms of protecting user information in distributed training. We expect more efforts in this direction in the future.

D. Deep Learning Driven Mobility Analysis

Understanding movement patterns of groups of human beings and individuals is becoming crucial for epidemiology, urban planning, public service provisioning, and mobile network resource management [483]. Deep learning is gaining increasing attention in this area, both from a group and individual level perspective (see Fig. 13). In this subsection, we thus discuss research using deep learning in this space, which we summarize in Table XIII.

Since deep learning is able to capture spatial dependencies in sequential data, it is becoming a powerful tool for mobility analysis. The applicability of deep learning for trajectory prediction is studied in [484]. By sharing representations learned by RNN and Gate Recurrent Unit (GRU), the framework can perform multi-task learning on both social networks and mobile trajectories modeling. Specifically, the authors first use deep learning to reconstruct social network representations of users, subsequently employing RNN and GRU models to learn patterns of mobile trajectories with different time granularity. Importantly, these two components jointly share representations learned, which tightens the overall architecture and enables efficient implementation. Ouyang *et al.* argue that mobility data are normally high-dimensional, which may be problematic for traditional ML models. Therefore, they build upon deep learning advances and propose an online learning scheme to train a hierarchical CNN architecture, allowing model parallelization for data stream processing [294]. By analyzing usage records, their framework "DeepSpace" predicts individuals' trajectories with much higher accuracy as compared to naive CNNs, as shown with experiments on a real-world dataset. Tkačík and Kordík [307] design a Neural Turing Machine [485] to predict trajectories of individuals using mobile phone data. The Neural Turing Machine embraces two major components: a memory module to store the historical trajectories, and a controller to manage the "read" and "write" operations over the memory. Experiments show that their architecture achieves superior generalization over stacked RNN and LSTM, while also delivering more

TABLE XIII
A SUMMARY OF WORK ON DEEP LEARNING DRIVEN MOBILITY ANALYSIS

Reference	Application	Mobility level	Model	Key contribution
Ouyang *et al.* [294]	Mobile user trajectory prediction	Individual	CNN	Online framework for data stream processing.
Yang *et al.* [295]	Social networks and mobile trajectories modeling	Mobile Ad-hoc Network	RNN, GRU	Multi-task learning.
Tkačík and Kordík [307]	Mobility modelling and prediction	Individual	Neural Turing Machine	The Neural Turing Machine can store historical data and perform "read" and "write" operations automatically.
Song *et al.* [296]	City-wide mobility prediction and transportation modeling	City-wide	Multi-task LSTM	Multi-task learning.
Zhang *et al.* [297]	City-wide crowd flows prediction	City-wide	Deep spatio-temporal residual networks (CNN-based)	Exploitation of spatio-temporal characteristics of mobility events.
Lin *et al.* [229]	Human activity chains generation	User group	Input-Output HMM + LSTM	Generative model.
Subramanian and Sadiq [298]	Mobile movement prediction	Individual	MLP	Fewer location updates and lower paging signaling costs.
Ezema and Ani [299]	Mobile location estimation	Individual	MLP	Operates with received signal strength in GSM.
Shao *et al.* [300]	CNN driven Pedometer	Individual	CNN	Reduced false negatives caused by periodic movements and lower initial response time.
Yayeh *et al.* [301]	Mobility prediction in mobile Ad-hoc network	Individual	MLP	Achieving high prediction accuracy under random waypoint mobility model.
Chen *et al.* [302]	Mobility driven traffic accident risk prediction	City-wide	Stacked denoising AE	Automatically learning correlation between human mobility and traffic accident risk.
Song *et al.* [303]	Human emergency behavior and mobility modelling	City-wide	DBN	Achieves accurate prediction over various disaster events, including earthquake, tsunami and nuclear accidents.
Yao *et al.* [304]	Trajectory clustering	User group	sequence-to-sequence AE with RNNs	The learned representations can robustly encode the movement characteristics of the objects and generate spatio-temporally invariant clusters.
Liu *et al.* [305]	Urban traffic prediction	City-wide	CNN, RNN, LSTM, AE, and RBM	Reveals the potential of employing deep learning to urban traffic prediction with mobility data.
Wickramasuriya *et al.* [306]	Base station prediction with proactive mobility management	Individual	RNN	Employs proactive and anticipatory mobility management for dynamic base station selection.
Kim and Song [308]	User mobility and personality modelling	Individual	MLP, RBM	Foundation for the customization of location-based services.
Jiang *et al.* [309]	Short-term urban mobility prediction	City-wide	RNN	Superior prediction accuracy and verified as highly deployable prototype system.
Wang *et al.* [310]	Mobility management in dense networks	Individual	LSTM	Improves the quality of service of mobile users in the handover process, while maintaining network energy efficiency.
Jiang *et al.* [311]	Urban human mobility prediction	City-wide	RNN	First work that utilizes urban region of interest to model human mobility city-wide.
Feng *et al.* [312]	Human mobility forecasting	Individual	Attention RNN	Employs the attention mechanism and combines heterogeneous transition regularity and multi-level periodicity.

precise trajectory prediction than the n-grams and k nearest neighbor methods.

Instead of focusing on individual trajectories, Song *et al.* [296] shed light on the mobility analysis at a larger scale. In their work, LSTM networks are exploited to jointly model the city-wide movement patterns of a large group of people and vehicles. Their multi-task architecture demonstrates superior prediction accuracy over a standard LSTM. City-wide mobile patterns is also researched in [297], where Zhang *et al.* architect deep spatio-temporal residual networks to forecast the movements of crowds. In order to capture the unique characteristics of spatio-temporal correlations associated with human mobility, their framework abandons RNN-based models and constructs three ResNets to extract nearby and distant spatial dependencies within a city. This scheme learns temporal features and fuses representations extracted by all models for the final prediction. By incorporating external events information, their proposal achieves the highest accuracy among all deep learning and non-deep learning methods studied. An RNN is also employed in [309], where Jiang *et al.* perform short-term urban mobility forecasting on a huge dataset collected from a real-world deployment. Their model delivers superior accuracy over the n-gram and Markovian approaches.

Lin *et al.* [229] consider generating human movement chains from cellular data, to support transportation planning. In particular, they first employ an input-output Hidden Markov Model (HMM) to label activity profiles for CDR data preprocessing. Subsequently, an LSTM is designed for activity chain generation, given the labeled activity sequences. They further synthesize urban mobility plans using the generative model and the simulation results reveal reasonable fit accuracy.

Jiang *et al.* [311] design 24-h mobility prediction system base on RNN mdoels. They employ dynamic Region of Interests (ROIs) for each hour to discovered through divide-and-merge mining from raw trajectory database, which leads to high prediction accuracy. Feng *et al.* incorporate attention mechanisms on RNN [312], to capture the complicated sequential transitions of human mobility. By combining the heterogeneous transition regularity and multi-level periodicity, their model delivers up to 10% of accuracy improvement compared to state-of-the-art forecasting models.

Yayeh *et al.* [301] employ an MLP to predict the mobility of mobile devices in mobile ad-hoc networks, given previously observed pause time, speed, and movement direction. Simulations conducted using the random waypoint mobility model show that their proposal achieves high prediction accuracy. An MLP is also adopted in [308], where Kim and Song model the relationship between human mobility and personality, and achieve high prediction accuracy. Yao *et al.* [304] discover groups of similar trajectories to facilitate higher-level mobility driven applications using RNNs. Particularly, a sequence-to-sequence AE is adopted to learn fixed-length representations of mobile users' trajectories. Experiments show that their method can effectively capture spatio-temporal patterns in both real and synthetic datasets. Shao *et al.* [300] design a sophisticated pedometer using a CNN. By reducing false negative steps caused by periodic movements, their proposal significantly improves the robustness of the pedometer.

Chen *et al.* [302] combine GPS records and traffic accident data to understand the correlation between human mobility and traffic accidents. To this end, they design a stacked denoising AE to learn a compact representation of the human mobility, and subsequently use that to predict the traffic accident risk. Their proposal can deliver accurate, real-time prediction across large regions. GPS records are also used in other mobility-driven applications. Song *et al.* [303] employ DBNs to predict and simulate human emergency behavior and mobility in natural disaster, learning from GPS records of 1.6 million users. Their proposal yields accurate predictions in different disaster scenarios such as earthquakes, tsunamis, and nuclear accidents. GPS data is also utilized in [305], where Liu *et al.* study the potential of employing deep learning for urban traffic prediction using mobility data.

Lessons Learned: Mobility analysis is concerned with the movement trajectory of a single user or large groups of users. The data of interest are essential time series, but have an additional spatial dimension. Mobility data is usually subject to stochasticity, loss, and noise; therefore precise modelling is not straightforward. As deep learning is able to perform automatic feature extraction, it becomes a strong candidate for human mobility modelling. Among them, CNNs and RNNs are the most successful architectures in such applications (e.g., [229] and [294]–[297]), as they can effectively exploit spatial and temporal correlations.

E. Deep Learning Driven User Localization

Location-based services and applications (e.g., mobile AR, GPS) demand precise individual positioning technology [486].

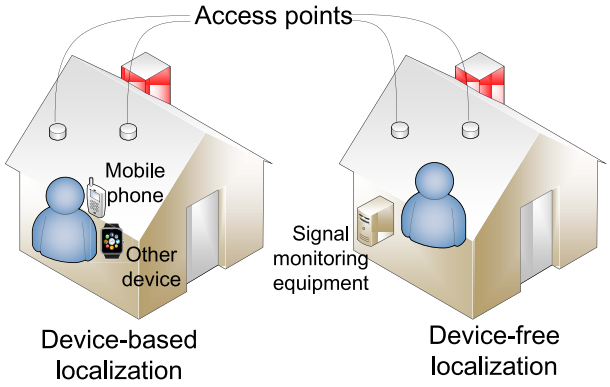


Fig. 14. An illustration of device-based (left) and device-free (right) indoor localization systems.

As a result, research on user localization is evolving rapidly and numerous techniques are emerging [487]. In general, user localization methods can be categorized as device-based and device-free [488]. We illustrate the two different paradigms in Fig. 14. Specifically, in the first category specific devices carried by users become prerequisites for fulfilling the applications' localization function. This type of approaches rely on signals from the device to identify the location. Conversely, approaches that require no device pertain to the device-free category. Instead these employ special equipment to monitor signal changes, in order to localize the entities of interest. Deep learning can enable high localization accuracy with both paradigms. We summarize the most notable contributions in Table XIV and delve into the details of these works next.

To overcome the variability and coarse-granularity limitations of signal strength based methods, Wang *et al.* [313] propose a deep learning driven fingerprinting system name "DeepFi" to perform indoor localization based on Channel State Information (CSI). Their toolbox yields much higher accuracy as compared to traditional methods, including FIFS [489], Horus [490], and Maximum Likelihood [491]. The same group of authors extend their work in [274], [275], [314], and [315], where they update the localization system, such that it can work with calibrated phase information of CSI [274], [275], [325]. They further use more sophisticated CNN [314], [333] and bi-modal structures [315] to improve the accuracy.

Nowicki and Wietrzykowski [316] propose a localization framework that reduces significantly the effort of system tuning or filtering and obtains satisfactory prediction performance. Wang *et al.* suggest that the objective of indoor localization can be achieved without the help of mobile devices. Wang *et al.* [318] employ an AE to learn useful patterns from WiFi signals. By automatic feature extraction, they produce a predictor that can fulfill multi-tasks simultaneously, including indoor localization, activity, and gesture recognition. A similar work in presented in [328], where Zhou *et al.* employ an MLP structure to perform device-free indoor localization using CSI. Kumar *et al.* [322] use deep learning to address the problem of indoor vehicles localization. They employ CNNs to analyze visual signal and localize vehicles in a car park. This can help

TABLE XIV
LEVERAGING DEEP LEARNING IN USER LOCALIZATION

Reference	Application	Input data	Device requirement	Model	Key contribution
Wang *et al.* [313]	Indoor fingerprinting	CSI	Device-based	RBM	First deep learning driven indoor localization based on CSI
Wang *et al.* [274], [275]	Indoor localization	CSI	Device-based	RBM	Works with calibrated phase information of CSI
Wang *et al.* [314]	Indoor localization	CSI	Device-based	CNN	Uses more robust angle of arrival for estimation
Wang *et al.* [315]	Indoor localization	CSI	Device-based	RBM	Bi-modal framework using both angle of arrival and average amplitudes of CSI
Nowicki and Wietrzykowski [316]	Indoor localization	WiFi scans	Device-based	Stacked AE	Requires less system tuning or filtering effort
Wang *et al.* [317], [318]	Indoor localization	Received signal strength	Device-free	Stacked AE	Device-free framework, multi-task learning
Mohammadi*et al.* [319]	Indoor localization	Received signal strength	Device-free	VAE+DQN	Handles unlabeled data; reinforcement learning aided semi-supervised learning
Anzum *et al.* [320]	Indoor localization	Received signal strength	Device-based	Counter propagation neural network	Solves the ambiguity among zones
Wang *et al.* [321]	Indoor localization	Smartphone magnetic and light sensors	Device-based	LSTM	Employ bimodal magnetic field and light intensity data
Kumar *et al.* [322]	Indoor vehicles localization	Camera images	Device-free	CNN	Focus on vehicles applications
Zheng and Weng [323]	Outdoor navigation	Camera images ad GPS	Device-free	Developmental network	Online learning scheme; edge-based
Zhang *et al.* [112]	Indoor and outdoor localization	Received signal strength	Device-based	Stacked AE	Operates under both indoor and outdoor environments
Vieira *et al.* [324]	Massive MIMO fingerprint-based positioning	Associated channel fingerprint	-	CNN	Operates with massive MIMO channels
Hsu *et al.* [335]	Activity recognition, localization and sleep monitoring	RF Signal	Device-free	CNN	Multi-user, device-free localization and sleep monitoring
Wang *et al.* [325]	Indoor localization	CSI	Device-based	RBM	Explores features of wireless channel data and obtains optimal weights as fingerprints
Wang *et al.* [333]	Indoor localization	CSI	Device-based	CNN	Exploits the angle of arrival for stable indoor localization
Xiao *et al.* [334]	3D Indoor localization	Bluetooth relative received signal strength	Device-free	Denoising AE	Low-cost and robust localization
Niitsoo *et al.* [332]	Indoor localization	Raw channel impulse response data	Device-free	CNN	Robust to multipath propagation environments
Ibrahim *et al.* [331]	Indoor localization	Received signal strength	Device-based	CNN	100% accuracy in building and floor identification
Adege *et al.* [330]	Indoor localization	Received signal strength	Device-based	MLP + linear discriminant analysis	Accurate in multi-building environments
Zhang *et al.* [329]	Indoor localization	Pervasive magnetic field and CSI	Device-based	MLP	Using the magnetic field to improve the positioning accuracy
Zhou *et al.* [328]	Indoor localization	CSI	Device-free	MLP	Device-free localization
Shokry *et al.* [327]	Outdoor localization	Crowd-sensed received signal strength information	Device-based	MLP	More accurate than cellular localization while requiring less energy
Chen *et al.* [326]	Indoor localization	CSI	Device-based	CNN	Represents CSI as feature images
Guan *et al.* [336]	Indoor localization	Line-of-sight and non line-of-sight radio signals	Device-based	MLP	Combining deep learning with genetic algorithms; visible light communication based

driver assistance systems operate in underground environments where the system has limited vision ability.

Xiao *et al.*[334] achieve low cost indoor localization with Bluetooth technology. The authors design a denosing AE to extract fingerprint features from the received signal strength of Bluetooth Low Energy beacon and subsequently project that to the exact position in 3D space. Experiments conducted in a conference room demonstrate that the proposed framework can perform precise positioning in both vertical and horizontal dimensions in real-time. Niitsoo *et al.* [332] employ a CNN to perform localization given raw channel impulse response data. Their framework is robust to multipath propagation environments and more precise than signal processing based approaches. A CNN is also adopted in [331], where the authors work with received signal strength series and achieve 100% prediction accuracy in terms of building and

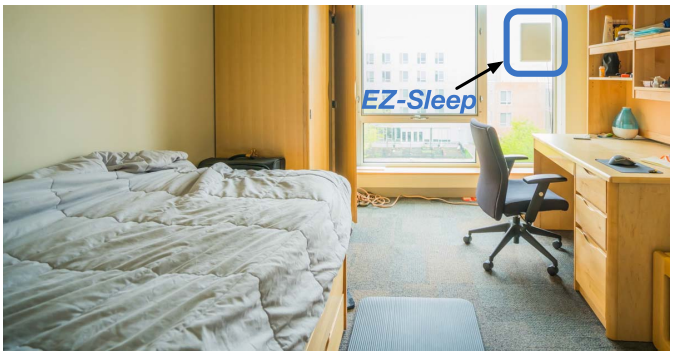


Fig. 15. EZ-Sleep setup in a subject's bedroom. Figure adopted from [335].

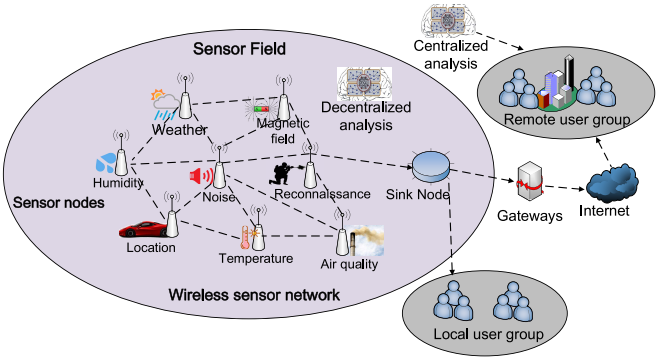


Fig. 16. An example framework for WSN data collection and (centralized and decentralized) analysis.

floor identification. The work in [330] combines deep learning with linear discriminant analysis for feature reduction, achieving low positioning errors in multi-building environments. Zhang *et al.* [329] combine pervasive magnetic field and WiFi fingerprinting for indoor localization using an MLP. Experiments show that adding magnetic field information to the input of the model can improve the prediction accuracy, compared to solutions based soley on WiFi fingerprinting.

Hsu *et al.* [335] use deep learning to provide Radio Frequency-based user localization, sleep monitoring, and insomnia analysis in multi-user home scenarios where individual sleep monitoring devices might not be available. They use a CNN classifier with a 14-layer residual network model for sleep monitoring, in addition to Hidden Markov Models, to accurately track when the user enters or leaves the bed. By deploying sleep sensors called EZ-Sleep in 8 homes (see Fig. 15), collecting data for 100 nights of sleep over a month, and cross-validating this using an electroencephalography-based sleep monitor, the authors demonstrate the performance of their solution is comparable to that of individual, electroencephalography-based devices.

Most mobile devices can only produce unlabeled position data, therefore unsupervised and semi-supervised learning become essential. Mohammadi *et al.* address this problem by leveraging DRL and VAE. In particular, their framework envisions a virtual agent in indoor environments [319], which can constantly receive state information during training, including signal strength indicators, current agent location, and the real (labeled data) and inferred (via a VAE) distance to the target. The agent can virtually move in eight directions at each time step. Each time it takes an action, the agent receives an reward signal, identifying whether it moves to a correct direction. By employing deep Q learning, the agent can finally localize accurately a user, given both labeled and unlabeled data.

Beyond indoor localization, there also exist several research works that apply deep learning in outdoor scenarios. For example, Zhengj and Weng [323] introduce a lightweight developmental network for outdoor navigation applications on mobile devices. Compared to CNNs, their architecture requires 100 times fewer weights to be updated, while maintaining decent accuracy. This enables efficient outdoor navigation on mobile devices. Work in [112] studies localization under both indoor and outdoor environments. They use an AE to pre-train a four-layer MLP, in order to avoid handcrafted feature engineering. The MLP is subsequently used to estimate the coarse position of targets. The authors further introduce an HMM to fine-tune the predictions based on temporal properties of data. This improves the accuracy estimation in both in-/out-door positioning with Wi-Fi signals. More recently, Shokry *et al.* [327] propose DeepLoc, a deep learning-based outdoor localization system using crowdsensed geo-tagged received signal strength information. By using an MLP to learn the correlation between cellular signal and users' locations, their framework can deliver median localization accuracy within 18.8m in urban areas and within 15.7m in rural areas on Android devices, while requiring modest energy budgets.

Lessons Learned: Localization relies on sensorial output, signal strength, or CSI. These data usually have complex features, therefore large amounts of data are required for learning [316]. As deep learning can extract features in an unsupervised manner, it has become a strong candidate for localization tasks. On the other hand, it can be observed that positioning accuracy and system robustness can be improved by fusing multiple types of signals when providing these as the input (e.g., [329]). Using deep learning to automatically extract features and correlate information from different sources for localization purposes is becoming a trend.

F. Deep Learning Driven Wireless Sensor Networks

Wireless Sensor Networks (WSNs) consist of a set of unique or heterogeneous sensors that are distributed over geographical regions. Theses sensors collaboratively monitor physical or environment status (e.g., temperature, pressure, motion, pollution, etc.) and transmit the data collected to centralized servers through wireless channels (see top circle in Fig. 9 for an illustration). A WSN typically involves three key core tasks, namely sensing, communication and analysis. Deep learning is becoming increasingly popular also for WSN applications [348]. In what follows, we review works adopting deep learning in this domain, covering different angles, namely: centralized vs. decentralized analysis paradigms, WSN data analysis per se, WSN localization, and other applications. Note that the contributions of these works are distinct from mobile

TABLE XV
A SUMMARY OF WORK ON DEEP LEARNING DRIVEN WSNS

Perspective	Reference	Application	Model	Optimizer	Key contribution
Centralized vs. Decentralized	Khorasani and Naji [345]	Data aggregation	MLP	Unknown	Improves the energy efficiency in the aggregation process
	Li *et al.* [346]	Distributed data mining	MLP	Unknown	Performing data analysis at distributed nodes, reducing by 58.31% the energy consumption
WSN Localization	Bernas and Płaczek [338]	Indoor localization	MLP	Resilient backpropagation	Dramatically reduces the memory consumption of received signal strength map storage
	Payal *et al.* [339]	Node localization	MLP	First-order and second-order gradient descent algorithm	Compares different training algorithms of MLP for WSN localization
	Dong *et al.* [340]	Underwater localization	MLP	RMSprop	Performs WSN localization in underwater environments
	Phoemphon *et al.* [350]	WSN node localization	Logic fuzzy system + ELM	Algorithm 4 described in [350]	Combining logic fuzzy system and ELM to achieve robust range-free WSN node localization
	Banihashemian *et al.* [351]	Range-free WSN node localization	MLP, ELM	Conjugate gradient based method	Employs a particle swarm optimization algorithm to simultaneously optimize the neural network based on storage cost and localization accuracy
	Kang *et al.* [353]	Water leakage detection and localization	CNN + SVM	SGD	Propose an enhanced graph-based local search algorithm using a virtual node scheme to select the nearest leakage location
	El *et al.* [356]	WSN localization in the presence of anisotropic signal attenuation	MLP	Unknown	Robust against anisotropic signal attenuation
WSN Data Analysis	Yan *et al.* [341]	Smoldering and flaming combustion identification	MLP	SGD	Achieves high accuracy in detecting fire in forests using smoke, CO_2 and temperature sensors
	Wang *et al.* [342]	Temperature correction	MLP	SGD	Employs deep learning to learn correlation between polar radiation and air temperature error
	Lee *et al.* [343]	Online query processing	CNN	Unknown	Employs adaptive query refinement to enable real-time analysis
	Li and Serpen [344]	Self-adaptive WSN	Hopfield network	Unknown	Embedding Hopfield NNs as a static optimizer for the weakly-connected dominating problem
	Khorasani and Naji [345]	Data aggregation	MLP	Unknown	Improves the energy efficiency in the aggregation process
	Li *et al.* [346]	Distributed data mining	MLP	Unknown	Distributed data mining
	Luo and Nagarajany [347]	Distributed WSN anomaly detection	AE	SGD	Employs distributed anomaly detection techniques to offload computations from the cloud
Other	Heydari *et al.* [349]	Energy consumption optimization and secure communication in wireless multimedia sensor networks	Stacked AE	Unknown	Use deep learning to enable fast data transfer and reduce energy consumption
	Mehmood *et al.* [354]	Robust routing for pollution monitoring in WSNs	MLP	SGD	Highly energy-efficient
	Alsheikh *et al.* [355]	Rate-distortion balanced data compression for WSNs	AE	Limited memory Broyden Fletcher Goldfarb Shann algorithm	Energy-efficient and bounded reconstruction errors
	Wang *et al.* [357]	Blind drift calibration for WSNs	Projection-recovery network (CNN based)	Adam	Exploits spatial and temporal correlations of data from all sensors; first work that adopts deep learning in WSN data calibration
	Jia *et al.* [358]	Ammonia monitoring	LSTM	Adam	Low-power and accurate ammonia monitoring

data analysis discussed in Sections VI-B and VI-C, as in this subsection we only focus on WSN applications. We begin by summarizing the most important works in Table XV.

Centralized vs Decentralized Analysis Approaches: There exist two data processing scenarios in WSNs, namely centralized and decentralized. The former simply takes sensors as

data collectors, which are only responsible for gathering data and sending these to a central location for processing. The latter assumes sensors have some computational ability and the main server offloads part of the jobs to the edge, each sensor performing data processing individually. We show an example framework for WSN data collection and analysis in Fig. 16, where sensor data is collected via various nodes in a field of interest. Such data is delivered to a sink node, which aggregates and optionally further processes this. Work in [345] focuses on the centralized approach and the authors apply a 3-layer MLP to reduce data redundancy while maintaining essential points for data aggregation. These data are sent to a central server for analysis. In contrast, Li *et al.* [346] propose to distribute data mining to individual sensors. They partition a deep neural network into different layers and offload layer operations to sensor nodes. Simulations conducted suggest that, by pre-processing with NNs, their framework obtains high fault detection accuracy, while reducing power consumption at the central server.

WSN Localization: Localization is also an important and challeging task in WSNs. Chuang and Jiang [337] exploit neural networks to localize sensor nodes in WSNs. To adapt deep learning models to specific network topology, they employ an online training scheme and correlated topology-trained data, enabling efficient model implementations and accurate location estimation. Based on this, Bernas and Płaczek [338] architect an ensemble system that involves multiple MLPs for location estimation in different regions of interest. In this scenario, node locations inferred by multiple MLPs are fused by a fusion algorithm, which improves the localization accuracy, particularly benefiting sensor nodes that are around the boundaries of regions. A comprehensive comparison of different training algorithms that apply MLP-based node localization is presented in [339]. Experiments suggest that the Bayesian regularization algorithm in general yields the best performance. Dong *et al.* [340] consider an underwater node localization scenario. Since acoustic signals are subject to loss caused by absorption, scattering, noise, and interference, underwater localization is not straightforward. By adopting a deep neural network, their framework successfully addresses the aforementioned challenges and achieves higher inference accuracy as compared to SVM and generalized least square methods.

Phoemphon *et al.* [350] combine a fuzzy logic system and an ELM via a particle swarm optimization technique to achieve robust range-free location estimation for sensor nodes. In particular, the fuzzy logic system is employed for adjusting the weight of traditional centroids, while the ELM is used for optimization for the localization precision. Their method achieves superior accuracy over other soft computing-based approaches. Similarly, Banihashemian *et al.* [351] employ the particle swarm optimization technique combining with MLPs to perform range-free WSN localization, which achieves low localization error. Kang *et al.* [353] shed light water leakage and localization in water distribution systems. They represent the water pipeline network as a graph and assume leakage events occur at vertices. They combine CNN with SVM to perform detection and localization on wireless sensor network testbed, achieving 99.3% leakage detection accuracy and localization error for less than 3 meters.

WSN Data Analysis: Deep learning has also been exploited for identification of smoldering and flaming combustion phases in forests. Yan *et al.* [341] embed a set of sensors into a forest to monitor CO_2, smoke, and temperature. They suggest that various burning scenarios will emit different gases, which can be taken into account when classifying smoldering and flaming combustion. Wang *et al.* [342] consider deep learning to correct inaccurate measurements of air temperature. They discover a close relationship between solar radiation and actual air temperature, which can be effectively learned by neural networks. Sun *et al.* [352] employ a Wavelet neural network based solution to evaluate radio link quality in WSNs on smart grids. Their proposal is more precise than traditional approaches and can provide end-to-end reliability guarantees to smart grid applications.

Missing data or de-synchronization are common in WSN data collection. These may lead to serious problems in analysis due to inconsistency. Lee *et al.* [343] address this problem by plugging a query refinement component in deep learning based WSN analysis systems. They employ exponential smoothing to infer missing data, thereby maintaining the integrity of data for deep learning analysis without significantly compromising accuracy. To enhance the intelligence of WSNs, Li and Serpen [344] embed an artificial neural network into a WSN, allowing it to agilely react to potential changes and following deployment in the field. To this end, they employ a minimum weakly-connected dominating set to represent the WSN topology, and subsequently use a Hopfield recurrent neural network as a static optimizer, to adapt network infrastructure to potential changes as necessary. This work represents an important step towards embedding machine intelligence in WSNs.

Other Applications: The benefits of deep learning have also been demonstrated in other WSN applications. The work in [349] focuses on reducing energy consumption while maintaining security in wireless multimedia sensor networks. A stacked AE is employed to categorize images in the form of continuous pieces, and subsequently send the data over the network. This enables faster data transfer rates and lower energy consumption. Mehmood *et al.* [354] employ MLPs to achieve robust routing in WSNs, so as to facilitate pollution monitoring. Their proposal use the NN to provide an efficiency threshold value and switch nodes that consume less energy than this threshold, thereby improving energy efficiency. Alsheikh *et al.* [355] introduce an algorithm for WSNs that uses AEs to minimize the energy expenditure. Their architecture exploits spatio-temporal correlations to reduce the dimensions of raw data and provides reconstruction error bound guarantees.

Wang *et al.* [357] design a dedicated projection-recovery neural network to blindly calibrate sensor measurements in an online manner. Their proposal can automatically extract features from sensor data and exploit spatial and temporal correlations among information from all sensors, to achieve high accuracy. This is the first effort that adopts deep learning in WSN data calibration. Jia *et al.* [358] shed light on ammonia monitoring using deep learning. In their design, an

LSTM is employed to predict the sensors' electrical resistance during a very short heating pulse, without waiting for settling in an equilibrium state. This dramatically reduces the energy consumption of sensors in the waiting process. Experiments with 38 prototype sensors and a home-built gas flow system show that the proposed LSTM can deliver precise prediction of equilibrium state resistance under different ammonia concentrations, cutting down the overall energy consumption by approximately 99.6%.

Lessons learned: The centralized and decentralized WSN data analysis paradigms resemble the cloud and fog computing philosophies in other areas. Decentralized methods exploit the computing ability of sensor nodes and perform light processing and analysis locally. This offloads the burden on the cloud and significantly reduces the data transmission overheads and storage requirements. However, at the moment, the centralized approach dominates the WSN data analysis landscape. As deep learning implementation on embedded devices becomes more accessible, in the future we expect to witness a grow in the popularity of the decentralized schemes.

On the other hand, looking at Table XV, it is interesting to see that the majority of deep learning practices in WSNs employ MLP models. Since MLP is straightforward to architect and performs reasonably well, it remains a good candidate for WSN applications. However, since most sensor data collected is sequential, we expect RNN-based models will play a more important role in this area.

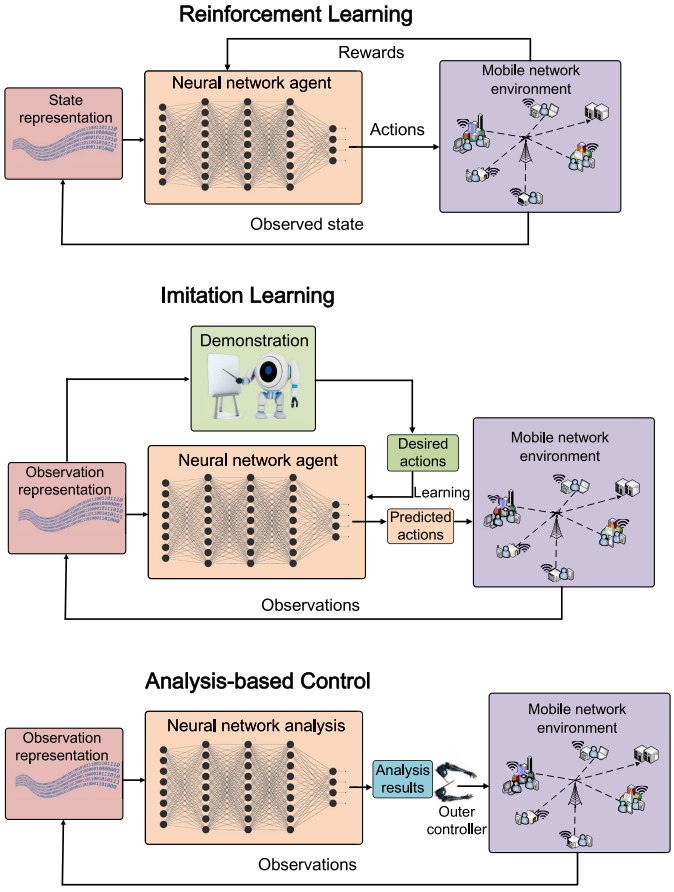


Fig. 17. Principles of three control approaches applied in mobile and wireless networks control, namely reinforcement learning (above), imitation learning (middle), and analysis-based control (below).

G. Deep Learning Driven Network Control

In this part, we turn our attention to mobile network control problems. Due to powerful function approximation mechanism, deep learning has made remarkable breakthroughs in improving traditional reinforcement learning [26] and imitation learning [492]. These advances have potential to solve mobile network control problems which are complex and previously considered intractable [493], [494]. Recall that in reinforcement learning, an agent continuously interacts with the environment to learn the best action. With constant exploration and exploitation, the agent learns to maximize its expected return. Imitation learning follows a different learning paradigm called "learning by demonstration". This learning paradigm relies on a 'teacher' who tells the agent what action should be executed under certain observations during the training. After sufficient demonstrations, the agent learns a policy that imitates the behavior of the teacher and can operate standalone without supervision. For instance, an agent is trained to mimic human behaviour (e.g., in applications such as game play, self-driving vehicles, or robotics), instead of learning by interacting with the environment, as in the case of pure reinforcement learning. This is because in such applications, making mistakes can have fatal consequences [27].

Beyond these two approaches, analysis-based control is gaining traction in mobile networking. Specifically, this scheme uses ML models for network data analysis, and subsequently exploits the results to aid network control. Unlike reinforcement/imitation learning, analysis-based control does not directly output actions. Instead, it extract useful information and delivers this to an agent, to execute the actions. We illustrate the principles between the three control paradigms in Fig. 17. We review works proposed so far in this space next, and summarize these efforts in Table XVI.

Network Optimization refers to the management of network resources and functions in a given environment, with the goal of improving the network performance. Deep learning has recently achieved several successful results in this area. For example, Liu *et al.* [359] exploit a DBN to discover the correlations between multi-commodity flow demand information and link usage in wireless networks. Based on the predictions made, they remove the links that are unlikely to be scheduled, so as to reduce the size of data for the demand constrained energy minimization. Their method reduces runtime by up to 50%, without compromising optimality. Subramanian and Banerjee [360] propose to use deep learning to predict the health condition of heterogeneous devices in machine to machine communications. The results obtained are subsequently exploited for optimizing health aware policy change decisions.

He *et al.* [361], [362] employ deep reinforcement learning to address caching and interference alignment problems in wireless networks. In particular, they treat time-varying channels as finite-state Markov channels and apply deep Q networks

TABLE XVI
A SUMMARY OF WORK ON DEEP LEARNING DRIVEN NETWORK CONTROL

Domain	Reference	Application	Control approach	Model
Network optimization	Liu *et al.* [359]	Demand constrained energy minimization	Analysis-based	DBN
	Subramanian and Banerjee [360]	Machine to machine system optimization	Analysis-based	Deep multi-modal network
	He *et al.* [361], [362]	Caching and interference alignment	Reinforcement learning	Deep Q learning
	Masmar and Evans [363]	mmWave Communication performance optimization	Reinforcement learning	Deep Q learning
	Wang *et al.* [364]	Handover optimization in wireless systems	Reinforcement learning	Deep Q learning
	Chen and Smith [365]	Cellular network random access optimization	Reinforcement learning	Deep Q learning
	Chen *et al.* [366]	Automatic traffic optimization	Reinforcement learning	Deep policy gradient
Routing	Lee *et al.* [367]	Virtual route assignment	Analysis-based	MLP
	Yang *et al.* [295]	Routing optimization	Analysis-based	Hopfield neural networks
	Mao *et al.* [187]	Software defined routing	Imitation learning	DBN
	Tang *et al.* [368]	Wireless network routing	Imitation learning	CNN
	Mao *et al.* [396]	Intelligent packet routing	Imitation learning	Tensor-based DBN
	Geyer *et al.* [397]	Distributed routing	Imitation learning	Graph-query neural network
	Pham *et al.* [404]	Routing for knowledge-defined networking	Reinforcement learning	Deep Deterministic Policy Gradient
Scheduling	Zhang *et al.* [369]	Hybrid dynamic voltage and frequency scaling scheduling	Reinforcement learning	Deep Q learning
	Atallah *et al.* [370]	Roadside communication network scheduling	Reinforcement learning	Deep Q learning
	Chinchali *et al.* [371]	Cellular network traffic scheduling	Reinforcement learning	Policy gradient
	Atallah *et al.* [370]	Roadside communications networks scheduling	Reinforcement learning	Deep Q learning
	Wei *et al.* [372]	User scheduling and content caching for mobile edge networks	Reinforcement learning	Deep policy gradient
	Mennes *et al.* [394]	Predicting free slots in a Multiple Frequencies Time Division Multiple Access (MF-TDMA) network.	Imitation learning	MLP
Resource allocation	Sun *et al.* [373]	Resource management over wireless networks	Imitation learning	MLP
	Xu *et al.* [374]	Resource allocation in cloud radio access networks	Reinforcement learning	Deep Q learning
	Ferreira *et al.* [375]	Resource management in cognitive communications	Reinforcement learning	Deep SARSA
	Challita *et al.* [377]	Proactive resource management for LTE	Reinforcement learning	Deep policy gradient
	Ye and Li [376]	Resource allocation in vehicle-to-vehicle communication	Reinforcement learning	Deep Q learning
	Li *et al.* [393]	Computation offloading and resource allocation for mobile edge computing	Reinforcement learning	Deep Q learning
	Zhou *et al.* [395]	Radio resource assignment	Analysis-based	LSTM
Radio control	Naparstek and Cohen [378]	Dynamic spectrum access	Reinforcement learning	Deep Q learning
	O'Shea and Clancy [379]	Radio control and signal detection	Reinforcement learning	Deep Q learning
	Wijaya *et al.* [380], [382]	Intercell-interference cancellation and transmit power optimization	Imitation learning	RBM
	Rutagemwa *et al.* [381]	Dynamic spectrum alignment	Analysis-based	RNN
	Yu *et al.* [389]	Multiple access for wireless network	Reinforcement learning	Deep Q learning
	Li *et al.* [399]	Power control for spectrum sharing in cognitive radios	Reinforcement learning	DQN
	Liu *et al.* [403]	Anti-jamming communications in dynamic and unknown environment	Reinforcement learning	DQN
	Luong *et al.* [398]	Transaction transmission and channel selection in cognitive radio blockchain	Reinforcement learning	Double DQN
	Ferreira *et al.* [495]	Radio transmitter settings selection in satellite communications	Reinforcement learning	Deep multi-objective reinforcement learning
Other	Mao *et al.* [383]	Adaptive video bitrate	Reinforcement learning	A3C
	Oda *et al.* [384], [385]	Mobile actor node control	Reinforcement learning	Deep Q learning
	Kim [386]	IoT load balancing	Analysis-based	DBN
	Challita *et al.* [387]	Path planning for aerial vehicle networking	Reinforcement learning	Multi-agent echo state networks
	Luo *et al.* [388]	Wireless online power control	Reinforcement learning	Deep Q learning
	Xu *et al.* [390]	Traffic engineering	Reinforcement learning	Deep policy gradient
	Liu *et al.* [391]	Base station sleep control	Reinforcement learning	Deep Q learning
	Zhao *et al.* [392]	Network slicing	Reinforcement learning	Deep Q learning
	Zhu *et al.* [236]	Mobile edge caching	Reinforcement learning	A3C
	Liu *et al.* [401]	Unmanned aerial vehicles control	Reinforcement learning	DQN
	Lee *et al.* [400]	Transmit power Control in device-to-device communications	Imitation learning	MLP
	He *et al.* [402]	Dynamic orchestration of networking, caching, and computing	Reinforcement learning	DQN

to learn the best user selection policy. This novel framework demonstrates significantly higher sum rate and energy efficiency over existing approaches. Chen *et al.* [366] shed light on automatic traffic optimization using a deep reinforcement learning approach. Specifically, they architect a two-level DRL framework, which imitates the Peripheral and Central

Nervous Systems in animals, to address scalability problems at datacenter scale. In their design, multiple peripheral systems are deployed on all end-hosts, so as to make decisions locally for short traffic flows. A central system is further employed to decide on the optimization with long traffic flows, which are more tolerant to longer delay. Experiments in a testbed with 32 severs suggest that the proposed design reduces the traffic optimization turn-around time and flow completion time significantly, compared to existing approaches.

Routing: Deep learning can also improve the efficiency of routing rules. Lee [367] exploit a 3-layer deep neural network to classify node degree, given detailed information of the routing nodes. The classification results along with temporary routes are exploited for subsequent virtual route generation using the Viterbi algorithm. Mao *et al.* [187] employ a DBN to decide the next routing node and construct a software defined router. By considering Open Shortest Path First as the optimal routing strategy, their method achieves up to 95% accuracy, while reducing significantly the overhead and delay, and achieving higher throughput with a signaling interval of 240 milliseconds. In follow up work, the authors use tensors to represent hidden layers, weights and biases in DBNs, which further improves the routing performance [396].

A similar outcome is obtained in [295], where Yang *et al.* employ Hopfield neural networks for routing, achieving better usability and survivability in mobile ad hoc network application scenarios. Geyer and Carle [397] represent the network using graphs, and design a dedicated Graph-Query NN to address the distributed routing problem. This novel architecture takes graphs as input and uses message passing between nodes in the graph, allowing it to operate with various network topologies. Pham *et al.* [404] shed light on routing protocols in knowledge-defined networking, using a Deep Deterministic Policy Gradient algorithm based on reinforcement learning. Their agent takes traffic conditions as input and incorporates QoS into the reward function. Simulations show that their framework can effectively learn the correlations between traffic flows, which leads to better routing configurations.

Scheduling: There are several studies that investigate scheduling with deep learning. Zhang *et al.* [369] introduce a deep Q learning-powered hybrid dynamic voltage and frequency scaling scheduling mechanism, to reduce the energy consumption in real-time systems (e.g., Wi-Fi, IoT, video applications). In their proposal, an AE is employed to approximate the Q function and the framework performs experience replay [496] to stabilize the training process and accelerate convergence. Simulations demonstrate that this method reduces by 4.2% the energy consumption of a traditional Q learning based method. Similarly, the work in [370] uses deep Q learning for scheduling in roadside communications networks. In particular, interactions between vehicular environments, including the sequence of actions, observations, and reward signals are formulated as an MDP. By approximating the Q value function, the agent learns a scheduling policy that achieves lower latency and busy time, and longer battery life, compared to traditional scheduling methods.

More recently, Chinchali *et al.* [371] present a policy gradient based scheduler to optimize the cellular network traffic flow. Specifically, they cast the scheduling problem as a MDP and employ RF to predict network throughput, which is subsequently used as a component of a reward function. Evaluations with a realistic network simulator demonstrate that this proposal can dynamically adapt to traffic variations, which enables mobile networks to carry 14.7% more data traffic, while outperforming heuristic schedulers by more than $2\times$. Wei *et al.* [372] address user scheduling and content caching simultaneously. In particular, they train a DRL agent, consisting of an actor for deciding which base station should serve certain content, and whether to save the content. A critic is further employed to estimate the value function and deliver feedback to the actor. Simulations over a cluster of base stations show that the agent can yield low transmission delay. Li *et al.* [393] shed light on resource allocation in a multi-user mobile computing scenario. They employ a deep Q learning framework to jointly optimize the offloading decision and computational resource allocation, so as to minimize the sum cost of delay and energy consumption of all user equipment. Simulations show that their proposal can reduce the total cost of the system, as compared to fully-local, fully-offloading, and naive Q-learning approaches.

Resource Allocation: Sun *et al.* use a deep neural network to approximate the mapping between the input and output of the Weighted Minimum Mean Square Error resource allocation algorithm [497], in interference-limited wireless network environments [373]. By effective imitation learning, the neural network approximation achieves close performance to that of its teacher. Deep learning has also been applied to cloud radio access networks, Xu *et al.* [374] employing deep Q learning to determine the on/off modes of remote radio heads given, the current mode and user demand. Comparisons with single base station association and fully coordinated association methods suggest that the proposed DRL controller allows the system to satisfy user demand while requiring significantly less energy.

Ferreira *et al.* [375] employ deep State-Action-Reward-State-Action (SARSA) to address resource allocation management in cognitive communications. By forecasting the effects of radio parameters, this framework avoids wasted trials of poor parameters, which reduces the computational resources required. Mennes *et al.* [394] employ MLPs to precisely forecast free slots prediction in a Multiple Frequencies Time Division Multiple Access (MF-TDMA) network, thereby achieving efficient scheduling. The authors conduct simulations with a network deployed in a 100×100 room, showing that their solution can effectively reduces collisions by half. Zhou *et al.* [395] adopt LSTMs to predict traffic load at base stations in ultra dense networks. Based on the predictions, their method changes the resource allocation policy to avoid congestion, which leads to lower packet loss rates, and higher throughput and mean opinion scores.

Radio Control: Naparstek and Cohen [378] address the dynamic spectrum access problem in multichannel wireless network environments using deep reinforcement learning. In this setting, they incorporate an LSTM into a deep Q network, to maintain and memorize historical observations, allowing

the architecture to perform precise state estimation, given partial observations. The training process is distributed to each user, which enables effective training parallelization and the learning of good policies for individual users. Experiments demonstrate that this framework achieves double the channel throughput, when compared to a benchmark method. Yu *et al.* apply deep reinforcement learning to address challenges in wireless multiple access control [389], recognizing that in such tasks DRL agents are fast in terms of convergence and robust against non-optimal parameter settings. Li *et al.* investigate power control for spectrum sharing in cognitive radios using DRL. In their design, a DQN agent is built to adjust the transmit power of a cognitive radio system, such that the overall signal-to-interference-plus-noise ratio is maximized.

The work in [379] sheds light on the radio control and signal detection problems. In particular, the authors introduce a radio signal search environment based on the Gym Reinforcement Learning platform. Their agent exhibits a steady learning process and is able to learn a radio signal search policy. Rutagemwa *et al.* [381] employ an RNN to perform traffic prediction, which can subsequently aid the dynamic spectrum assignment in mobile networks. With accurate traffic forecasting, their proposal improves the performance of spectrum sharing in dynamic wireless environments, as it attains near-optimal spectrum assignments. Liu *et al.* [403] approach the anti-jamming communications problem in dynamic and unknown environments with a DRL agent. Their system is based on a DQN with CNN, where the agent takes raw spectrum information as input and requires limited prior knowledge about the environment, in order to improve the overall throughput of the network in such adversarial circumstances.

Luong *et al.* incorporate the blockchain technique into cognitive radio networking [398], employing a double DQN agent to maximize the number of successful transaction transmissions for secondary users, while minimizing the channel cost and transaction fees. Simulations show that the DQN method significantly outperforms na ive Q learning in terms of successful transactions, channel cost, and learning speed. DRL can further attack problems in the satellite communications domain. Ferreira *et al.* [495] fuse multi-objective reinforcement learning [405] with deep neural networks to select among multiple radio transmitter settings while attempting to achieve multiple conflicting goals, in a dynamically changing satellite communications channel. Specifically, two set of NNs are employed to execute exploration and exploitation separately. This builds an ensembling system, with makes the framework more robust to the changing environment. Simulations demonstrate that their system can nearly optimize six different objectives (i.e., bit error rate, throughput, bandwidth, spectral efficiency, additional power consumption, and power efficiency), only with small performance errors compared to ideal solutions.

Other Applications: Deep learning is playing an important role in other network control problems as well. Mao *et al.* [383] develop the Pensieve system that generates adaptive video bit rate algorithms using deep reinforcement learning. Specifically, Pensieve employs a state-of-the-art deep reinforcement learning algorithm A3C, which takes the bandwidth, bit rate and buffer size as input, and selects the bit rate that leads to the best expected return. The model is trained offline and deployed on an adaptive bit rate server, demonstrating that the system outperforms the best existing scheme by 12%-25% in terms of QoE. Liu *et al.* [391] apply deep Q learning to reduce the energy consumption in cellular networks. They train an agent to dynamically switch on/off base stations based on traffic consumption in areas of interest. An action-wise experience replay mechanism is further designed for balancing different traffic behaviours. Experiments show that their proposal can significantly reduce the energy consumed by base stations, outperforming naive table-based Q learning approaches. A control mechanism for unmanned aerial vehicles using DQN is proposed in [401], where multiple objectives are targeted: maximizing energy efficiency, communications coverage, fairness and connectivity. The authors conduct extensive simulations in an virtual playground, showing that their agent is able to learn the dynamics of the environment, achieving superior performance over random and greedy control baselines.

Kim and Kim [386] link deep learning with the load balancing problem in IoT. The authors suggest that DBNs can effectively analyze network load and process structural configuration, thereby achieving efficient load balancing in IoT. Challita *et al.* [387] employ a deep reinforcement learning algorithm based on echo state networks to perform path planning for a cluster of unmanned aerial vehicles. Their proposal yields lower delay than a heuristic baseline. Xu *et al.* [390] employ a DRL agent to learn from network dynamics how to control traffic flow. They advocate that DRL is suitable for this problem, as it performs remarkably well in handling dynamic environments and sophisticated state spaces. Simulations conducted over three network topologies confirm this viewpoint, as the DRL agent significantly reduces the delay, while providing throughput comparable to that of traditional approaches. Zhu *et al.* employ the A3C algorithm to address the caching problem in mobile edge computing. Their method obtains superior cache hit ratios and traffic offloading performance over three baselines caching methods. Several open challenges are also pointed out, which are worthy of future pursuit. The edge caching problem is also addressed in [402], where He *et al.* architect a DQN agent to perform dynamic orchestration of networking, caching, and computing. Their method facilitates high revenue to mobile virtual network operators.

Lessons learned: There exist three approaches to network control using deep learning, i.e., reinforcement learning, imitation learning, and analysis-based control. Reinforcement learning requires to interact with the environment, trying different actions and obtaining feedback in order to improve. The agent will make mistakes during training, and usually needs a large number of steps of steps to become smart. Therefore, most works do not train the agent on the real infrastructure, as making mistakes usually can have serious consequences for the network. Instead, a simulator that mimics the real network environments is built and the agent is trained offline using that. This imposes high fidelity requirements on the simulator,

as the agent can not work appropriately in an environment that is different from the one used for training. On the other hand, although DRL performs remarkable well in many applications, considerable amount of time and computing resources are required to train an usable agent. This should be considered in real-life implementation.

In contrast, the imitation learning mechanism "learns by demonstration". It requires a teacher that provides labels telling the agent what it should do under certain circumstances. In the networking context, this mechanism is usually employed to reduce the computational time [187]. Specifically, in some network application (e.g., routing), computing the optimal solution is time-consuming, which cannot satisfy the delay constraints of mobile network. To mitigate this, one can generate a large dataset offline, and use an NN agent to learn the optimal actions.

Analysis-based control on the other hand, is suitable for problems were decisions cannot be based solely on the state of the network environment. One can use a NN to extract additional information (e.g., traffic forecasts), which subsequently aids decisions. For example, the dynamic spectrum assignment can benefit from the analysis-based control.

H. Deep Learning Driven Network Security

With the increasing popularity of wireless connectivity, protecting users, network equipment and data from malicious attacks, unauthorized access and information leakage becomes crucial. Cyber security systems guard mobile devices and users through firewalls, anti-virus software, and Intrusion Detection Systems (IDS) [498]. The firewall is an access security gateway that allows or blocks the uplink and downlink network traffic, based on pre-defined rules. Anti-virus software detects and removes computer viruses, worms and Trojans and malware. IDSs identify unauthorized and malicious activities, or rule violations in information systems. Each performs its own functions to protect network communication, central servers and edge devices.

Modern cyber security systems benefit increasingly from deep learning [500], since it can enable the system to *(i)* automatically learn signatures and patterns from experience and generalize to future intrusions (supervised learning); or *(ii)* identify patterns that are clearly differed from regular behavior (unsupervised learning). This dramatically reduces the effort of pre-defined rules for discriminating intrusions. Beyond protecting networks from attacks, deep learning can also be used for attack purposes, bringing huge potential to steal or crack user passwords or information. In this subsection, we review deep learning driven network security from three perspectives, namely infrastructure, software, and user privacy. Specifically, infrastructure level security work focuses on detecting anomalies that occur in the physical network and software level work is centred on identifying malware and botnets in mobile networks. From the user privacy perspective, we discuss methods to protect from how to protect against private information leakage, using deep learning. To our knowledge, no other reviews summarize these efforts. We summarize these works in Table XVII.

Infrastructure level security: We mostly focus on anomaly detection at the infrastructure level, i.e., identifying network events (e.g., attacks, unexpected access and use of data) that do not conform to expected behaviors. Many researchers exploit the outstanding unsupervised learning ability of AEs [406]. For example, Thing investigates features of attacks and threats that exist in IEEE 802.11 networks [186]. The author employs a stacked AE to categorize network traffic into 5 types (i.e., legitimate, flooding, injection and impersonation traffic), achieving 98.67% overall accuracy. The AE is also exploited in [407], where Aminanto and Kim use an MLP and stacked AE for feature selection and extraction, demonstrating remarkable performance. Similarly, Feng *et al.* [408] use AEs to detect abnormal spectrum usage in wireless communications. Their experiments suggest that the detection accuracy can significantly benefit from the depth of AEs.

Distributed attack detection is also an important issue in mobile network security. Khan *et al.* [409] focus on detecting flooding attacks in wireless mesh networks. They simulate a wireless environment with 100 nodes, and artificially inject moderate and severe distributed flooding attacks, to generate a synthetic dataset. Their deep learning based methods achieve excellent false positive and false negative rates. Distributed attacks are also studied in [410], where Diro and Chilamkurti focus on an IoT scenario. Another work in [411] employs MLPs to detect distributed denial of service attacks. By characterizing typical patterns of attack incidents, the proposed model works well in detecting both known and unknown distributed denial of service attacks. More recently, Nguyen *et al.* [421] employ RBMs to classify cyberattacks in the mobile cloud in an online manner. Through unsupervised layer-wise pre-training and fine-tuning, their methods obtain over 90% classification accuracy on three different datasets, significantly outperforming other machine learning approaches.

Lopez-Martin *et al.* [412] propose a conditional VAE to identify intrusion incidents in IoT. In order to improve detection performance, their VAE infers missing features associated with incomplete measurements, which are common in IoT environments. The true data labels are embedded into the decoder layers to assist final classification. Evaluations on the well-known NSL-KDD dataset [501] demonstrate that their model achieves remarkable accuracy in identifying denial of service, probing, remote to user and user to root attacks, outperforming traditional ML methods by 0.18 in terms of F1 score. Hamedani *et al.* [413] employ MLPs to detect malicious attacks in delayed feedback networks. The proposal achieves more than 99% accuracy over 10,000 simulations.

Software level security: Nowadays, mobile devices are carrying considerable amount of private information. This information can be stolen and exploited by malicious apps installed on smartphones for ill-conceived purposes [502]. Deep learning is being exploited for analyzing and detecting such threats.

Yuan *et al.* [416] use both labeled and unlabeled mobile apps to train an RBM. By learning from 300 samples, their model can classify Android malware with remarkable accuracy, outperforming traditional ML tools by up to

TABLE XVII
A SUMMARY OF WORK ON DEEP LEARNING DRIVEN NETWORK SECURITY

Level	Reference	Application	Problem considered	Learning paradigm	Model
Infrastructure	Azar *et al.* [406]	Cyber security applications	Malware classification & Denial of service, probing, remote to user & user to root	Unsupervised & supervised	Stacked AE
	Thing [186]	IEEE 802.11 network anomaly detection and attack classification	Flooding, injection and impersonation attacks	Unsupervised & supervised	Stacked AE
	Aminanto and Kim [407]	Wi-Fi impersonation attacks detection	Flooding, injection and impersonation attacks	Unsupervised & supervised	MLP, AE
	Feng *et al.* [408]	Spectrum anomaly detection	Sudden signal-to-noise ratio changes in the communication channels	Unsupervised & supervised	AE
	Khan *et al.* [409]	Flooding attacks detection in wireless mesh networks	Moderate and severe distributed flood attack	Supervised	MLP
	Diro and Chilamkurti [410]	IoT distributed attacks detection	Denial of service, probing, remote to user & user to root	Supervised	MLP
	Saied *et al.* [411]	Distributed denial of service attack detection	Known and unknown distributed denial of service attack	Supervised	MLP
	Martin *et al.* [412]	IoT intrusion detection	Denial of service, probing, remote to user & user to root	Unsupervised & supervised	Conditional VAE
	Hamedani *et al.* [413]	Attacks detection in delayed feedback networks	Attack detection in smart grids using reservoir computing	Supervised	MLP
	Luo and Nagarajany [347]	Anomalies in WSNs	Spikes and burst recorded by temperature and relative humidity sensors	Unsupervised	AE
	Das *et al.* [414]	IoT authentication	Long duration of signal imperfections	Supervised	LSTM
	Jiang *et al.* [415]	MAC spoofing detection	Packets from different hardware use same MAC address	Supervised	CNN
	Jiang *et al.* [421]	Cyberattack detection in mobile cloud computing	Various cyberattacks from 3 different datasets	Unsupervised + supervised	RBM
Software	Yuan *et al.* [416]	Android malware detection	Apps in Contagio Mobile and Google Play Store	Unsupervised & supervised	RBM
	Yuan *et al.* [417]	Android malware detection	Apps in Contagio Mobile, Google Play Store and Genome Project	Unsupervised & supervised	DBN
	Su *et al.* [418]	Android malware detection	Apps in Drebin, Android Malware Genome Project, the Contagio Community, and Google Play Store	Unsupervised & supervised	DBN + SVM
	Hou *et al.* [419]	Android malware detection	App samples from Comodo Cloud Security Center	Unsupervised & supervised	Stacked AE
	Martinelli [420]	Android malware detection	Apps in Drebin, Android Malware Genome Project and Google Play Store	Supersived	CNN
	McLaughlin *et al.* [422]	Android malware detection	Apps in Android Malware Genome project and Google Play Store	Supersived	CNN
	Chen *et al.* [423]	Malicious application detection at the network edge	Publicly-available malicious applications	Unsupervised & supervised	RBM
	Wang *et al.* [225]	Malware traffic classification	Traffic extracted from 9 types of malware	Superivised	CNN
	Oulehla *et al.* [424]	Mobile botnet detection	Client-server and hybrid botnets	Unknown	Unknown
	Torres *et al.* [425]	Botnet detection	Spam, HTTP and unknown traffic	Superivised	LSTM
	Eslahi *et al.* [426]	Mobile botnet detection	HTTP botnet traffic	Superivised	MLP
	Alauthaman *et al.* [427]	Peer-to-peer botnet detection	Waledac and Strom Bots	Superivised	MLP
User privacy	Shokri and Shmatikov [428]	Privacy preserving deep learning	Avoiding sharing data in collaborative model training	Superivised	MLP, CNN
	Phong *et al.* [429]	Privacy preserving deep learning	Addressing information leakage introduced in [428]	Supervised	MLP
	Ossia *et al.* [430]	Privacy-preserving mobile analytics	Offloading feature extraction from cloud	Supervised	CNN
	Abadi *et al.* [431]	Deep learning with differential privacy	Preventing exposure of private information in training data	Supervised	MLP
	Osia *et al.* [432]	Privacy-preserving personal model training	Offloading personal data from clouds	Unsupervised & supervised	MLP
	Servia *et al.* [433]	Privacy-preserving model inference	Breaking down large models for privacy-preserving analytics	Supervised	CNN
	Hitaj *et al.* [434]	Stealing information from collaborative deep learning	Breaking the ordinary and differentially private collaborative deep learning	Unsupervised	GAN
	Hitaj *et al.* [431]	Password guessing	Generating passwords from leaked password set	Unsupervised	GAN
	Greydanus [435]	Enigma learning	Reconstructing functions of polyalphabetic cipher	Supervised	LSTM
	Maghrebi [436]	Breaking cryptographic	Side channel attacks	Supervised	MLP, AE, CNN, LSTM
	Liu *et al.* [437]	Password guessing	Employing adversarial generation to guess passwords	Unsupervised	LSTM
	Ning *et al.* [438]	Mobile apps sniffing	Defending against mobile apps sniffing through noise injection	Supervised	CNN
	Wang *et al.* [499]	Private inference in mobile cloud	Computation offloading and privacy preserving for mobile inference	Supervised	MLP, CNN

19%. Their follow-up research in [417] named Droiddetector further improves the detection accuracy by 2%. Similarly, Su *et al.* [418] analyze essential features of Android apps, namely requested permission, used permission, sensitive application programming interface calls, action and app components. They employ DBNs to extract features of malware and

an SVM for classification, achieving high accuracy and only requiring 6 seconds per inference instance.

Hou *et al.* [419] attack the malware detection problem from a different perspective. Their research points out that signature-based detection is insufficient to deal with sophisticated Android malware. To address this problem, they propose the Component Traversal, which can automatically execute code routines to construct weighted directed graphs. By employing a Stacked AE for graph analysis, their framework Deep4MalDroid can accurately detect Android malware that intentionally repackages and obfuscates to bypass signatures and hinder analysis attempts to their inner operations. This work is followed by that of Martinelli *et al.* [420], who exploit CNNs to discover the relationship between app types and extracted syscall traces from real mobile devices. The CNN has also been used in [422], where McLaughlin *et al.* draw inspiration from NLP and take the disassembled byte-code of an app as a text for analysis. Their experiments demonstrate that CNNs can effectively learn to detect sequences of opcodes that are indicative of malware. Chen *et al.* [423] incorporate location information into the detection framework and exploit an RBM for feature extraction and classification. Their proposal improves the performance of other ML methods.

Botnets are another important threat to mobile networks. A botnet is effectively a network that consists of machines compromised by bots. These machine are usually under the control of a botmaster who takes advantages of the bots to harm public services and systems [503]. Detecting botnets is challenging and now becoming a pressing task in cyber security. Deep learning is playing an important role in this area. For example, Oulehla *et al.* [424] propose to employ neural networks to extract features from mobile botnet behaviors. They design a parallel detection framework for identifying both client-server and hybrid botnets, and demonstrate encouraging performance. Torres *et al.* [425] investigate the common behavior patterns that botnets exhibit across their life cycle, using LSTMs. They employ both under-sampling and over-sampling to address the class imbalance between botnet and normal traffic in the dataset, which is common in anomaly detection problems. Similar issues are also studies in [426] and [427], where the authors use standard MLPs to perform mobile and peer-to-peer botnet detection respectively, achieving high overall accuracy.

User privacy level: Preserving user privacy during training and evaluating a deep neural network is another important research issue [504]. Initial research is conducted in [428], where Shokri and Shmatikov enable user participation in the training and evaluation of a neural network, without sharing their input data. This allows to preserve individual's privacy while benefiting all users, as they collaboratively improve the model performance. Their framework is revisited and improved in [429], where another group of researchers employ additively homomorphic encryption, to address the information leakage problem ignored in [428], without compromising model accuracy. This significantly boosts the security of the system. More recently, Wang *et al.* [499] propose a framework called ARDEN to preserve users' privacy while reducing communication overhead in mobile-cloud deep learning applications. ARDEN partitions a NN across cloud and mobile devices, with heavy computation being conducted on the cloud and mobile devices performing only simple data transformation and perturbation, using a differentially private mechanism. This simultaneously guarantees user privacy, improves inference accuracy, and reduces resource consumption.

Ossia *et al.* [430] focus on privacy-preserving mobile analytics using deep learning. They design a client-server framework based on the Siamese architecture [505], which accommodates a feature extractor in mobile devices and correspondingly a classifier in the cloud. By offloading feature extraction from the cloud, their system offers strong privacy guarantees. An innovative work in [431] implies that deep neural networks can be trained with differential privacy. The authors introduce a differentially private SGD to avoid disclosure of private information of training data. Experiments on two publicly-available image recognition datasets demonstrate that their algorithm is able to maintain users privacy, with a manageable cost in terms of complexity, efficiency, and performance. This approach is also useful for edge-based privacy filtering techniques such as Distributed One-class Learning [506].

Servia-Rodriguez *et al.* [433] consider training deep neural networks on distributed devices without violating privacy constraints. Specifically, the authors retrain an initial model locally, tailored to individual users. This avoids transferring personal data to untrusted entities, hence user privacy is guaranteed. Osia *et al.* focus on protecting user's personal data from the inferences' perspective. In particular, they break the entire deep neural network into a feature extractor (on the client side) and an analyzer (on the cloud side) to minimize the exposure of sensitive information. Through local processing of raw input data, sensitive personal information is transferred into abstract features, which avoids direct disclosure to the cloud. Experiments on gender classification and emotion detection suggest that this framework can effectively preserve user privacy, while maintaining remarkable inference accuracy.

Deep learning has also been exploited for cyber attacks, including attempts to compromise private user information and guess passwords. Hitaj *et al.* [434] suggest that learning a deep model collaboratively is not reliable. By training a GAN, their attacker is able to affect such learning process and lure the victims to disclose private information, by injecting fake training samples. Their GAN even successfully breaks the differentially private collaborative learning in [431]. The authors further investigate the use of GANs for password guessing. In [507], they design PassGAN, which learns the distribution of a set of leaked passwords. Once trained on a dataset, PassGAN is able to match over 46% of passwords in a different testing set, without user intervention or cryptography knowledge. This novel technique has potential to revolutionize current password guessing algorithms.

Greydanus [435] breaks a decryption rule using an LSTM network. They treat decryption as a sequence-to-sequence translation task, and train a framework with large enigma pairs. The proposed LSTM demonstrates remarkable performance in learning polyalphabetic ciphers. Maghrebi *et al.* [436] exploit various deep learning models (i.e., MLP, AE, CNN, LSTM) to construct a precise profiling system and perform side channel

TABLE XVIII
A SUMMARY OF DEEP LEARNING DRIVEN SIGNAL PROCESSING

Domain	Reference	Application	Model
MIMO systems	Samuel *et al.* [448]	MIMO detection	MLP
	Yan *et al.* [449]	Signal detection in a MIMO-OFDM system	AE+ELM
	Vieira *et al.* [324]	Massive MIMO fingerprint-based positioning	CNN
	Neumann *et al.* [447]	MIMO channel estimation	CNN
	Wijaya *et al.* [380], [382]	Inter-cell interference cancellation and transmit power optimization	RBM
	O'Shea *et al.* [439]	Optimization of representations and encoding/decoding processes	AE
	Borgerding *et al.* [440]	Sparse linear inverse problem in MIMO	CNN
	Fujihashi *et al.* [441]	MIMO nonlinear equalization	MLP
	Huang *et al.* [459]	Super-resolution channel and direction-of-arrival estimation	MLP
Modulation	Rajendran *et al.* [442]	Automatic modulation classification	LSTM
	West and O'Shea [443]	Modulation recognition	CNN, ResNet, Inception CNN, LSTM
	O'Shea *et al.* [444]	Modulation recognition	Radio transformer network
	O'Shea and Hoydis [450]	Modulation classification	CNN
	Jagannath *et al.* [451]	Modulation classification in a software defined radio testbed	MLP
Others	O'Shea *et al.* [452]	Radio traffic sequence recognition	LSTM
	O'Shea *et el.* [453]	Learning to communicate over an impaired channel	AE + radio transformer network
	Ye *et al.* [454]	Channel estimation and signal detection in OFDM systsms.	MLP
	Liang *et al.* [455]	Channel decoding	CNN
	Lyu *et al.* [456]	NNs for channel decoding	MLP, CNN and RNN
	Dörner *et al.* [457]	Over-the-air communications system	AE
	Liao *et al.* [458]	Rayleigh fading channel prediction	MLP
	Huang *et al.* [460]	Light-emitting diode (LED) visible light downlink error correction	AE
	Alkhateeb *et al.* [446]	Coordinated beamforming for highly-mobile millimeter wave systems	MLP
	Gante *et al.* [445]	Millimeter wave positioning	CNN
	Ye *et al.* [508]	Channel agnostic end-to-end learning based communication system	Conditional GAN

key recovery attacks. Surprisingly, deep learning based methods demonstrate overwhelming performance over other template machine learning attacks in terms of efficiency in breaking both unprotected and protected Advanced Encryption Standard implementations. Ning *et al.* [438] demonstrate that an attacker can use a CNN to infer with over 84% accuracy what apps run on a smartphone and their usage, based on magnetometer or orientation data. The accuracy can increase to 98% if motion sensors information is also taken into account, which jeopardizes user privacy. To mitigate this issue, the authors propose to inject Gaussian noise into the magnetometer and orientation data, which leads to a reduction in inference accuracy down to 15%, thereby effectively mitigating the risk of privacy leakage.

Lessons learned: Most deep learning based solutions focus on existing network attacks, yet new attacks emerge every day. As these new attacks may have different features and appear to behave 'normally', old NN models may not easily detect them. Therefore, an effective deep learning technique should be able to *(i)* rapidly transfer the knowledge of old attacks to detect newer ones; and *(ii)* constantly absorb the features of newcomers and update the underlying model. Transfer learning and lifelong learning are strong candidates to address this problems, as we will discuss in Section VII-C. Research in this directions remains shallow, hence we expect more efforts in the future.

Another issue to which attention should be paid is the fact that NNs are vulnerable to adversarial attacks. This has been briefly discussed in Section III-E. Although formal reports on this matter are lacking, hackers may exploit weaknesses in NN models and training procedures to perform attacks that subvert deep learning based cyber-defense systems. This is an important potential pitfall that should be considered in real implementations.

I. Deep Learning Driven Signal Processing

Deep learning is also gaining increasing attention in signal processing, in applications including Multi-Input Multi-Output (MIMO) and modulation. MIMO has become a fundamental technique in current wireless communications, both in cellular and WiFi networks. By incorporating deep learning, MIMO performance is intelligently optimized based on environment conditions. Modulation recognition is also evolving to be more accurate, by taking advantage of deep learning. We give an overview of relevant work in this area in Table XVIII.

MIMO Systems: Samuel *et al.* suggest that deep neural networks can be a good estimator of transmitted vectors in a MIMO channel. By unfolding a projected gradient descent method, they design an MLP-based detection network to perform binary MIMO detection [448]. The Detection Network can be implemented on multiple channels after a single training. Simulations demonstrate that the proposed architecture achieves near-optimal accuracy, while requiring light computation without prior knowledge of Signal-to-Noise Ratio (SNR). Yan *et al.* [449] employ deep learning to solve a similar problem from a different perspective. By considering the characteristic invariance of signals, they exploit an AE as a feature extractor, and subsequently use an Extreme Learning Machine (ELM) to classify signal sources in a MIMO orthogonal frequency division multiplexing (OFDM) system. Their proposal achieves higher detection accuracy than several traditional methods, while maintaining similar complexity.

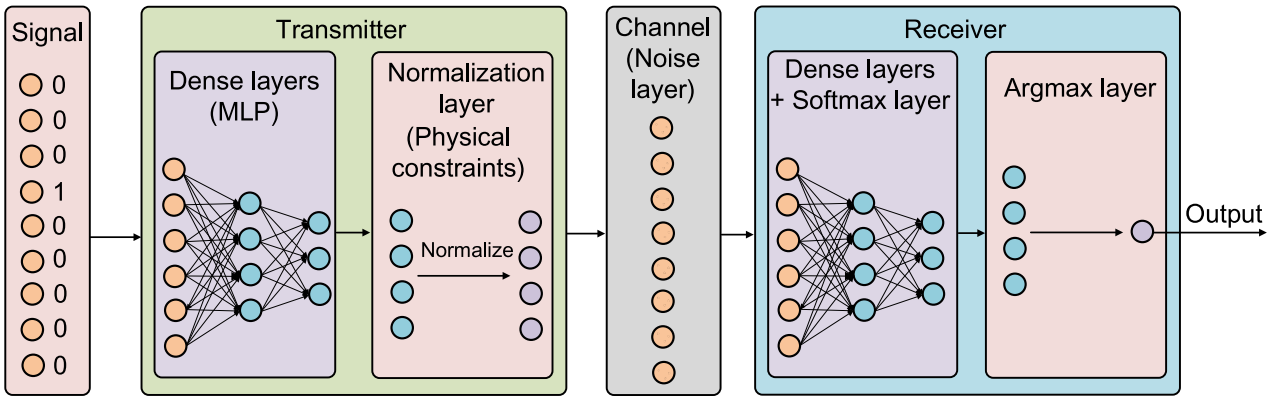


Fig. 18. A communications system over an additive white Gaussian noise channel represented as an autoencoder.

Vieira *et al.* [324] show that massive MIMO channel measurements in cellular networks can be utilized for fingerprint-based inference of user positions. Specifically, they design CNNs with weight regularization to exploit the sparse and information-invariance of channel fingerprints, thereby achieving precise positions inference. CNNs have also been employed for MIMO channel estimation. Neumann *et al.* [447] exploit the structure of the MIMO channel model to design a lightweight, approximated maximum likelihood estimator for a specific channel model. Their methods outperform traditional estimators in terms of computation cost and reduce the number of hyper-parameters to be tuned. A similar idea is implemented in [454], where Ye *et al.* employ an MLP to perform channel estimation and signal detection in OFDM systems.

Wijaya *et al.* [380], [382] consider applying deep learning to a different scenario. The authors propose to use non-iterative neural networks to perform transmit power control at base stations, thereby preventing degradation of network performance due to inter-cell interference. The neural network is trained to estimate the optimal transmit power at every packet transmission, selecting that with the highest activation probability. Simulations demonstrate that the proposed framework significantly outperform the belief propagation algorithm that is routinely used for transmit power control in MIMO systems, while attaining a lower computational cost.

More recently, O'Shea *et al.* [439] bring deep learning to physical layer design. They incorporate an unsupervised deep AE into a single-user end-to-end MIMO system, to optimize representations and the encoding/decoding processes, for transmissions over a Rayleigh fading channel. We illustrate the adopted AE-based framework in Fig. 18. This design incorporates a transmitter consisting of an MLP followed by a normalization layer, which ensures that physical constraints on the signal are guaranteed. After transfer through an additive white Gaussian noise channel, a receiver employs another MLP to decode messages and select the one with the highest probability of occurrence. The system can be trained with an SGD algorithm in an end-to-end manner. Experimental results show that the AE system outperforms the Space Time Block Code approach in terms of SNR by approximately 15 dB. Borgerding *et al.* [440] propose to use deep learning to recover a sparse signal from noisy linear measurements in MIMO environments. The proposed scheme is evaluated on compressive random access and massive-MIMO channel estimation, where it achieves better accuracy over traditional algorithms and CNNs.

Modulation: West and O'Shea [443] compare the modulation recognition accuracy of different deep learning architectures, including traditional CNN, ResNet, Inception CNN, and LSTM. Their experiments suggest that the LSTM is the best candidate for modulation recognition, since it achieves the highest accuracy. Due to its superior performance, an LSTM is also employed for a similar task in [442]. O'Shea *et al.* then focus on tailoring deep learning architectures to radio properties. Their prior work is improved in [444], where they architect a novel deep radio transformer network for precise modulation recognition. Specifically, they introduce radio-domain specific parametric transformations into a spatial transformer network, which assists in the normalization of the received signal, thereby achieving superior performance. This framework also demonstrates automatic synchronization abilities, which reduces the dependency on traditional expert systems and expensive signal analytic processes. O'Shea and Hoydis [450] introduce several novel deep learning applications for the network physical layer. They demonstrate a proof-of-concept where they employ a CNN for modulation classification and obtain satisfying accuracy.

Other Signal Processing Applciations: Deep learning is also adopted for radio signal analysis. O'Shea *et al.* [452] employ an LSTM to replace sequence translation routines between radio transmitter and receiver. Although their framework works well in ideal environments, its performance drops significantly when introducing realistic channel effects. Later, the authors consider a different scenario in [453], where they exploit a regularized AE to enable reliable communications over an impaired channel. They further incorporate a radio transformer network for signal reconstruction at the decoder side, thereby achieving receiver synchronization. Simulations demonstrate that this approach is reliable and can be efficiently implemented.

TABLE XIX
A SUMMARY OF EMERGING DEEP LEARNING DRIVEN MOBILE NETWORK APPLICATIONS

Reference	Application	Model	Key contribution
Gonzalez *et al.* [461]	Network data monetifzation	-	A platform named Net2Vec to facilitate deep learning deployment in communication networks.
Kaminski *et al.* [462]	In-network computation for IoT	MLP	Enables to perform collaborative data processing and reduces latency.
Xiao *et al.* [463]	Mobile crowdsensing	Deep Q learning	Mitigates vulnerabilities of mobile crowdsensing systems.
Luong *et al.* [464]	Resource allocation for mobile blockchains	MLP	Employs deep learning to perform monotone transformations of miners'bids and outputs the allocation and conditional payment rules in optimal auctions.
Gulati *et al.* [465]	Data dissemination in Internet of Vehicles (IoV)	CNN	Investigates the relationship between data dissemination performance and social score, energy level, number of vehicles and their speed.

Liang *et al.* [455] exploit noise correlations to decode channels using a deep learning approach. Specifically, they use a CNN to reduce channel noise estimation errors by learning the noise correlation. Experiments suggest that their framework can significantly improve the decoding performance. The decoding performance of MLPs, CNNs and RNNs is compared in [456]. By conducting experiments in different setting, the obtained results suggest the RNN achieves the best decoding performance, nonetheless yielding the highest computational overhead. Liao *et al.* [458] employ MLPs to perform accurate Rayleigh fading channel prediction. The authors further equip their proposal with a sparse channel sample construction method to save system resources without compromising precision. Deep learning can further aid visible light communication. Huang and Lin [460] employ a deep learning based system for error correction in optical communications. Specifically, an AE is used in their work to perform dimension reduction on light-emitting diode (LED) visible light downlink, thereby maximizing the channel bandwidth . The proposal follows the theory in [450], where O'Shea and Hoydis demonstrate that deep learning driven signal processing systems can perform as good as traditional encoding and/or modulation systems.

Deep learning has been further adopted in solving millimeter wave beamforming. Alkhateeb *et al.* [446] propose a millimeter wave communication system that utilizes MLPs to predict beamforming vectors from signals received from distributed base stations. By substituting a genie-aided solution with deep learning, their framework reduces the coordination overhead, enabling wide-coverage and low-latency beamforming. Similarly, Gante *et al.* employ CNNs to infer the position of a device, given the received millimeter wave radiation. Their preliminary simulations show that the CNN-based system can achieve small estimation errors in a realistic outdoors scenario, significantly outperforming existing prediction approaches.

Lessons learned: Deep learning is beginning to play an important role in signal processing applications and the performance demonstrated by early prototypes is remarkable. This is because deep learning can prove advantageous with regards to performance, complexity, and generalization capabilities. At this stage, research in this area is however incipient. We can only expect that deep learning will become increasingly popular in this area.

J. Emerging Deep Learning Applications in Mobile Networks

In this part, we review work that builds upon deep learning in other mobile networking areas, which are beyond the scopes of the subjects discussed thus far. These emerging applications open several new research directions, as we discuss next. A summary of these works is given in Table XIX.

Network Data Monetization: Gonzalez *et al.* employ unsupervised deep learning to generate real-time accurate user profiles [461] using an on-network machine learning platform called Net2Vec [509]. Specifically, they analyze user browsing data in real time and generate user profiles using product categories. The profiles can be subsequently associated with the products that are of interest to the users and employed for online advertising.

IoT In-Network Computation: Instead of regarding IoT nodes as producers of data or the end consumers of processed information, Kaminski *et al.* [462] embed neural networks into an IoT deployment and allow the nodes to collaboratively process the data generated. This enables low-latency communication, while offloading data storage and processing from the cloud. In particular, the authors map each hidden unit of a pre-trained neural network to a node in the IoT network, and investigate the optimal projection that leads to the minimum communication overhead. Their framework achieves functionality similar to in-network computation in WSNs and opens a new research directions in fog computing.

Mobile Crowdsensing: Xiao *et al.* investigate vulnerabilities facing crowdsensing in the mobile network context. They argue that there exist malicious mobile users who intentionally provide false sensing data to servers, to save costs and preserve their privacy, which in turn can make mobile crowdsensings systems vulnerable [463]. The authors model the server-users system as a Stackelberg game, where the server plays the role of a leader that is responsible for evaluating the sensing effort of individuals, by analyzing the accuracy of each sensing report. Users are paid based on the evaluation of their efforts, hence cheating users will be punished with zero reward. To design an optimal payment policy, the server employs a deep Q network, which derives knowledge from experience sensing reports, without requiring specific sensing models. Simulations demonstrate superior performance in terms of sensing quality, resilience to attacks, and server utility, as compared to traditional Q learning based and random payment strategies.

Mobile Blockchain: Substantial computing resource requirements and energy consumption limit the applicability of blockchain in mobile network environments. To mitigate this problem, Luong *et al.* shed light on resource management in mobile blockchain networks based on optimal auction in [464]. They design an MLP to first conduct monotone transformations of the miners' bids and subsequently output the allocation

scheme and conditional payment rules for each miner. By running experiments with different settings, the results suggest the propsoed deep learning based framework can deliver much higher profit to edge computing service provider than the second-price auction baseline.

Internet of Vehicles (IoV): Gulati *et al.* [465] extend the success of deep learning to IoV. The authors design a deep learning-based content centric data dissemination approach that comprises three steps, namely *(i)* performing energy estimation on selected vehicles that are capable of data dissemination; *(ii)* employing a Weiner process model to identify stable and reliable connections between vehicles; and *(iii)* using a CNN to predict the social relationship among vehicles. Experiments unveil that the volume of data disseminated is positively related to social score, energy levels, and number of vehicles, while the speed of vehicles has negative impact on the connection probability.

Lessons learned: The adoption of deep learning in the mobile and wireless networking domain is exciting and undoubtedly many advances are yet to appear. However, as discussed in Section III-E, deep learning solutions are not universal and may not be suitable for every problem. One should rather regard deep learning as a powerful tool that can assist with fast and accurate inference, and facilitate the automation of some processes previously requiring human intervention. Nevertheless, deep learning algorithms will make mistakes, and their decisions might not be easy to interpret. In tasks that require high interpretability and low fault-tolerance, deep learning still has a long way to go, which also holds for the majority of ML algorithms.

VII. Tailoring Deep Learning to Mobile Networks

Although deep learning performs remarkably in many mobile networking areas, the No Free Lunch (NFL) theorem indicates that there is no single model that can work universally well in all problems [510]. This implies that for any specific mobile and wireless networking problem, we may need to adapt different deep learning architectures to achieve the best performance. In this section, we look at how to tailor deep learning to mobile networking applications from three perspectives, namely, mobile devices and systems, distributed data centers, and changing mobile network environments.

A. Tailoring Deep Learning to Mobile Devices and Systems

The ultra-low latency requirements of future 5G networks demand runtime efficiency from all operations performed by mobile systems. This also applies to deep learning driven applications. However, current mobile devices have limited hardware capabilities, which means that implementing complex deep learning architectures on such equipment may be computationally unfeasible, unless appropriate model tuning is performed. To address this issue, ongoing research improves existing deep learning architectures [511], such that the inference process does not violate latency or energy constraints [512], [513], nor raise any privacy concern [514]. We outline these works in Table XX and discuss their key contributions next.

Iandola *et al.* [515] design a compact architecture for embedded systems named SqueezeNet, which has similar accuracy to that of AlexNet [87], a classical CNN, yet 50 times fewer parameters. SqueezeNet is also based on CNNs, but its significantly smaller model size *(i)* allows more efficiently training on distributed systems; *(ii)* reduces the transmission overhead when updating the model at the client side; and *(iii)* facilitates deployment on resource-limited embedded devices. Howard *et al.* [516] extend this work and introduce an efficient family of streamlined CNNs called MobileNet, which uses depth-wise separable convolution operations to drastically reduce the number of computations required and the model size. This new design can run with low latency and can satisfy the requirements of mobile and embedded vision applications. The authors further introduce two hyper-parameters to control the width and resolution of multipliers, which can help strike an appropriate trade-off between accuracy and efficiency. The ShuffleNet proposed by Zhang *et al.* [517] improves the accuracy of MobileNet by employing point-wise group convolution and channel shuffle, while retaining similar model complexity. In particular, the authors discover that more groups of convolution operations can reduce the computation requirements.

Zhang *et al.* [518] focus on reducing the number of parameters of structures with fully-connected layers for mobile multimedia features learning. This is achieved by applying Trucker decomposition to weight sub-tensors in the model, while maintaining decent reconstruction capability. The Trucker decomposition has also been employed in [523], where Huynh *et al.* seek to approximate a model with fewer parameters, in order to save memory. Mobile optimizations are further studied for RNN models. Cao *et al.* [519] use a mobile toolbox called RenderScript[35] to parallelize specific data structures and enable mobile GPUs to perform computational accelerations. Their proposal reduces the latency when running RNN models on Android smartphones. Chen *et al.* [520] shed light on implementing CNNs on iOS mobile devices. In particular, they reduce the model executions latency, through space exploration for data re-usability and kernel redundancy removal. The former alleviates the high bandwidth requirements of convolutional layers, while the latter reduces the memory and computational requirements, with negligible performance degradation.

Rallapalli *et al.* [521] investigate offloading *very deep* CNNs from clouds to edge devices, by employing memory optimization on both mobile CPUs and GPUs. Their framework enables running at high speed deep CNNs with large memory requirements in mobile object detection applications. Lane *et al.* develop a software accelerator, DeepX, to assist deep learning implementations on mobile devices. The proposed approach exploits two inference-time resource control algorithms, i.e., runtime layer compression and deep architecture decomposition [522]. The runtime layer compression technique controls the memory and computation runtime during the inference phase, by extending model compression

[35] Android Renderscript https://developer.android.com/guide/topics/renderscript/compute.html.

TABLE XX
SUMMARY OF WORKS ON IMPROVING DEEP LEARNING FOR MOBILE DEVICES AND SYSTEMS

Reference	Methods	Target model
Iandola *et al.* [515]	Filter size shrinking, reducing input channels and late downsampling	CNN
Howard *et al.* [516]	Depth-wise separable convolution	CNN
Zhang *et al.* [517]	Point-wise group convolution and channel shuffle	CNN
Zhang *et al.* [518]	Tucker decomposition	AE
Cao *et al.* [519]	Data parallelization by RenderScript	RNN
Chen *et al.* [520]	Space exploration for data reusability and kernel redundancy removal	CNN
Rallapalli *et al.* [521]	Memory optimizations	CNN
Lane *et al.* [522]	Runtime layer compression and deep architecture decomposition	MLP, CNN
Huynh *et al.* [523]	Caching, Tucker decomposition and computation offloading	CNN
Wu *et al.* [524]	Parameters quantization	CNN
Bhattacharya and Lane [525]	Sparsification of fully-connected layers and separation of convolutional kernels	MLP, CNN
Georgiev *et al.* [98]	Representation sharing	MLP
Cho and Brand [526]	Convolution operation optimization	CNN
Guo and Potkonjak [527]	Filters and classes pruning	CNN
Li *et al.* [528]	Cloud assistance and incremental learning	CNN
Zen *et al.* [529]	Weight quantization	LSTM
Falcao *et al.* [530]	Parallelization and memory sharing	Stacked AE
Fang *et al.* [531]	Model pruning and recovery scheme	CNN
Xu *et al.* [532]	Reusable region lookup and reusable region propagation scheme	CNN
Liu *et al.* [533]	Using deep Q learning based optimizer to achieve appropriate balance between accuracy, latency, storage and energy consumption for deep NNs on mobile platforms	CNN
Chen *et al.* [534]	Machine learning based optimization system to automatically explore and search for optimized tensor operators	All NN architectures
Yao *et al.* [535]	Learning model execution time and performing the model compression	MLP, CNN, GRU, and LSTM
Li *et al.* [536]	streamline slimming and branch slimming	CNN

principles. This is important in mobile devices, since offloading the inference process to edge devices is more practical with current hardware platforms. Further, the deep architecture designs "decomposition plans" that seek to optimally allocate data and model operations to local and remote processors. By combining these two, DeepX enables maximizing energy and runtime efficiency, under given computation and memory constraints. Yao *et al.* [535] design a framework called FastDeepIoT, which first learns the execution time of NN models on target devices, and subsequently conducts model compression to reduce the runtime without compromising the inference accuracy. Through this process, up to 78% of execution time and 69% of energy consumption is reduced, compared to state-of-the-art compression algorithms.

More recently, Fang *et al.* [531] design a framework called NestDNN, to provide flexible resource-accuracy trade-offs on mobile devices. To this end, the NestDNN first adopts a model pruning and recovery scheme, which translates deep NNs to single compact multi-capacity models. With this approach up to 4.22% inference accuracy can be achieved with six mobile vision applications, at a 2.0× faster video frame processing rate and reducing energy consumption by 1.7×. Xu *et al.* [532] accelerate deep learning inference for mobile vision from the caching perspective. In particular, the proposed framework called DeepCache stores recent input frames as cache keys and recent feature maps for individual CNN layers as cache values. The authors further employ reusable region lookup and reusable region propagation, to enable a region matcher to only run once per input video frame and load cached feature maps at all layers inside the CNN. This reduces the inference time by 18% and energy consumption by 20% on average. Liu *et al.* [533] develop a usage-driven framework named AdaDeep, to select a combination of compression techniques for a specific deep NN on mobile platforms. By using a deep Q learning optimizer, their proposal can achieve appropriate trade-offs between accuracy, latency, storage and energy consumption.

Beyond these works, researchers also successfully adapt deep learning architectures through other designs and sophisticated optimizations, such as parameters quantization [524], [529], model slimming [536], sparsification and separation [525], representation and memory sharing [98], [530], convolution operation optimization [526], pruning [527], cloud assistance [528] and compiler optimization [534]. These techniques will be of great significance when embedding deep neural networks into mobile systems.

B. Tailoring Deep Learning to Distributed Data Containers

Mobile systems generate and consume massive volumes of mobile data every day. This may involve similar content, but which is distributed around the world. Moving such data to centralized servers to perform model training and evaluation inevitably introduces communication and storage overheads, which does not scale. However, neglecting characteristics embedded in mobile data, which are associated with local culture, human mobility, geographical topology, etc., during model training can compromise the robustness of the model and implicitly the performance of the mobile network applications that build on such models. The solution is to offload model execution to distributed data centers or edge devices, to guarantee good performance, whilst alleviating the burden on the cloud.

As such, one of the challenges facing parallelism, in the context of mobile networking, is that of training neural networks on a large number of mobile devices that are battery powered,

TABLE XXI
SUMMARY OF WORK ON MODEL AND TRAINING PARALLELISM FOR MOBILE SYSTEMS AND DEVICES

Parallelism Paradigm	Reference	Target	Core Idea	Improvement
Model parallelism	Dean *et al.* [127]	Very large deep neural networks in distributed systems.	Employs downpour SGD to support a large number of model replicas and a Sandblaster framework to support a variety of batch optimizations.	Up to 12× model training speed up, using 81 machines.
	Teerapittayanon *et al.* [537]	Neural network on cloud and end devices.	Maps a deep neural network to a distributed setting and jointly trains each individual section.	Up to 20× reduction of communication cost.
	De Coninck *et al.* [114]	Neural networks on IoT devices.	Distills from a pre-trained NN to obtain a smaller NN that performs classification on a subset of the entire space.	10ms inference latency on a mobile device.
	Omidshafiei *et al.* [538]	Multi-task multi-agent reinforcement learning under partial observability.	Deep recurrent Q-networks & cautiously-optimistic learners to approximate action-value function; decentralized concurrent experience replays trajectories to stabilize training.	Near-optimal execution time.
Training parallelism	Recht *et al.* [539]	Parallelized SGD.	Eliminates overhead associated with locking in distributed SGD.	Up to 10× speed up in distributed training.
	Goyal *et al.* [540]	Distributed synchronous SGD.	Employs a hyper-parameter-free learning rule to adjust the learning rate and a warmup mechanism to address the early optimization problem.	Trains billions of images per day.
	Zhang *et al.* [541]	Asynchronous distributed SGD.	Combines the stochastic variance reduced gradient algorithm and a delayed proximal gradient algorithm.	Up to 6× speed up
	Hardy *et al.* [542]	Distributed deep learning on edge-devices.	Compression technique (*AdaComp*) to reduce ingress traffic at parameter severs.	Up to 191× reduction in ingress traffic.
	McMahan *et al.* [543]	Distributed training on mobile devices.	Users collectively enjoy benefits of shared models trained with big data without centralized storage.	Up to 64.3× training speedup.
	Keith *et al.* [544]	Data computation over mobile devices.	Secure multi-party computation to obtain model parameters on distributed mobile devices.	Up to 1.98× communication expansion.

have limited computational capabilities and in particular lack GPUs. The key goal of this paradigm is that of training with a large number of mobile CPUs at least as effective as with GPUs. The speed of training remains important, but becomes a secondary goal.

Generally, there are two routes to addressing this problem, namely, *(i)* decomposing the model itself, to train (or make inference with) its components individually; or *(ii)* scaling the training process to perform model update at different locations associated with data containers. Both schemes allow one to train a single model without requiring to centralize all data. We illustrate the principles of these two approaches in Fig. 19 and summarize the existing work in Table XXI.

Model Parallelism: Large-scale distributed deep learning is first studied in [127], where Dean *et al.* develop a framework named *DistBelief*, which enables training complex neural networks on thousands of machines. In their framework, the full model is partitioned into smaller components and distributed over various machines. Only nodes with edges (e.g., connections between layers) that cross boundaries between machines are required to communicate for parameters update and inference. This system further involves a parameter server, which enables each model replica to obtain the latest parameters during training. Experiments demonstrate that the proposed framework can be training significantly faster on a CPU cluster, compared to training on a single GPU, while achieving state-of-the-art classification performance on ImageNet [192].

Teerapittayanon *et al.* [537] propose deep neural networks tailored to distributed systems, which include cloud servers, fog layers and geographically distributed devices. The authors scale the overall neural network architecture and distribute its components hierarchically from cloud to end devices. The model exploits local aggregators and binary weights, to reduce computational storage, and communication overheads, while maintaining decent accuracy. Experiments on a multi-view multi-camera dataset demonstrate that this proposal can perform efficient cloud-based training and local inference. Importantly, without violating latency constraints, the deep neural network obtains essential benefits associated with distributed systems, such as fault tolerance and privacy.

Coninck *et al.* [114] consider distributing deep learning over IoT for classification applications. Specifically, they deploy a small neural network to local devices, to perform coarse classification, which enables fast response filtered data to be sent to central servers. If the local model fails to classify, the larger neural network in the cloud is activated to perform fine-grained classification. The overall architecture maintains good accuracy, while significantly reducing the latency typically introduced by large model inference.

Decentralized methods can also be applied to deep reinforcement learning. Omidshafiei *et al.* [538] consider a multi-agent system with partial observability and limited communication, which is common in mobile systems. They combine a set of sophisticated methods and algorithms, including hysteresis learners, a deep recurrent Q network, concurrent experience replay trajectories and distillation, to enable multi-agent coordination using a single joint policy under a set of decentralized partially observable MDPs. Their framework can potentially play an important role in addressing control problems in distributed mobile systems.

Training Parallelism is also essential for mobile system, as mobile data usually come asynchronously from different sources. Training models effectively while maintaining

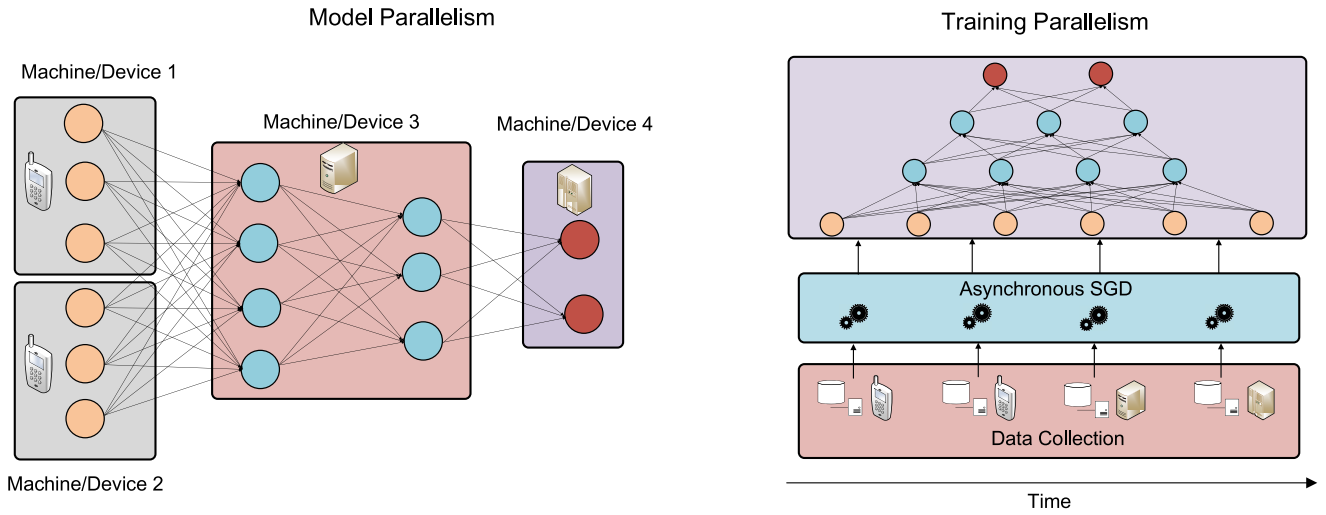


Fig. 19. The underlying principles of model parallelism (left) and training parallelism (right).

consistency, fast convergence, and accuracy remains however challenging [545].

A practical method to address this problem is to perform asynchronous SGD. The basic idea is to enable the server that maintains a model to accept delayed information (e.g., data, gradient updates) from workers. At each update iteration, the server only requires to wait for a smaller number of workers. This is essential for training a deep neural network over distributed machines in mobile systems. The asynchronous SGD is first studied in [539], where Recht *et al.* propose a lock-free parallel SGD named HOGWILD, which demonstrates significant faster convergence over locking counterparts. The Downpour SGD in [127] improves the robustness of the training process when work nodes breakdown, as each model replica requests the latest version of the parameters. Hence a small number of machine failures will not have a significant impact on the training process. A similar idea has been employed in [540], where Goyal *et al.* investigate the usage of a set of techniques (i.e., learning rate adjustment, warm-up, batch normalization), which offer important insights into training large-scale deep neural networks on distributed systems. Eventually, their framework can train an network on ImageNet within 1 hour, which is impressive in comparison with traditional algorithms.

Zhang *et al.* [541] argue that most of asynchronous SGD algorithms suffer from slow convergence, due to the inherent variance of stochastic gradients. They propose an improved SGD with variance reduction to speed up the convergence. Their algorithm outperforms other asynchronous SGD approaches in terms of convergence, when training deep neural networks on the Google Cloud Computing Platform. The asynchronous method has also been applied to deep reinforcement learning. Mnih *et al.* [79] create multiple environments, which allows agents to perform asynchronous updates to the main structure. The new A3C algorithm breaks the sequential dependency and speeds up the training of the traditional Actor-Critic algorithm significantly. Hardy *et al.* [542] further study distributed deep learning over cloud and edge devices. In particular, they propose a training algorithm, *AdaComp*, which allows to compress worker updates of the target model. This significantly reduce the communication overhead between cloud and edge, while retaining good fault tolerance.

Federated learning is an emerging parallelism approach that enables mobile devices to collaboratively learn a shared model, while retaining all training data on individual devices [543], [546]. Beyond offloading the training data from central servers, this approach performs model updates with a Secure Aggregation protocol [544], which decrypts the average updates only if enough users have participated, without inspecting individual updates. Based on this idea, Google recently build a prototype system using federated Learning in the domain of mobile devices [547]. This fulfills the objective that "bringing the code to the data, instead of the data to the code", which protects individuals' privacy.

C. Tailoring Deep Learning to Changing Mobile Network Environments

Mobile network environments often exhibit changing patterns over time. For instance, the spatial distributions of mobile data traffic over a region may vary significantly between different times of the day [548]. Applying a deep learning model in changing mobile environments requires lifelong learning ability to continuously absorb new features, without forgetting old but essential patterns. Moreover, new smartphone-targeted viruses are spreading fast via mobile networks and may severely jeopardize users' privacy and business profits. These pose unprecedented challenges to current anomaly detection systems and anti-virus software, as such tools must react to new threats in a timely manner, using limited information. To this end, the model should have transfer learning ability, which can enable the fast transfer of knowledge from pre-trained models to different jobs or datasets. This will allow models to work well with limited threat samples (one-shot learning) or limited metadata descriptions of new threats (zero-shot learning). Therefore, both lifelong learning and transfer learning are essential for applications in ever changing mobile network

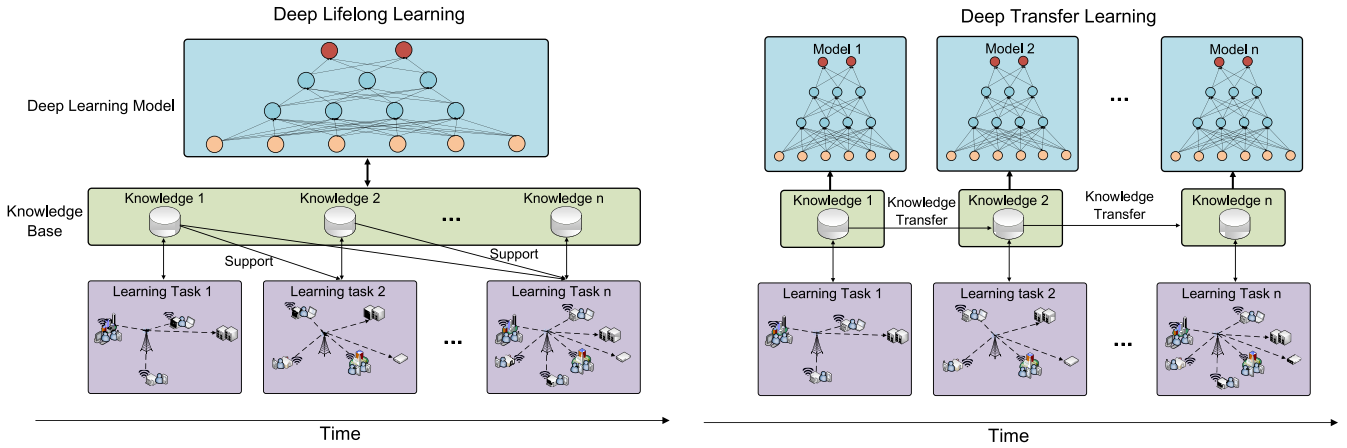


Fig. 20. The underlying principles of deep lifelong learning (left) and deep transfer learning (right). Lifelong learning retains the knowledge learned while transfer learning exploits labeled data of one domain to learn in a new target domain.

environments. We illustrated these two learning paradigms in Fig. 20 and review essential research in this subsection.

Deep Lifelong Learning mimics human behaviors and seeks to build a machine that can continuously adapt to new environments, retain as much knowledge as possible from previous learning experience [549]. There exist several research efforts that adapt traditional deep learning to lifelong learning. For example, Lee *et al.* [550] propose a dual-memory deep learning architecture for lifelong learning of everyday human behaviors over non-stationary data streams. To enable the pre-trained model to retain old knowledge while training with new data, their architecture includes two memory buffers, namely a deep memory and a fast memory. The deep memory is composed of several deep networks, which are built when the amount of data from an unseen distribution is accumulated and reaches a threshold. The fast memory component is a small neural network, which is updated immediately when coming across a new data sample. These two memory modules allow to perform continuous learning without forgetting old knowledge. Experiments on a non-stationary image data stream prove the effectiveness of this model, as it significantly outperforms other online deep learning algorithms. The memory mechanism has also been applied in [551]. In particular, the authors introduce a differentiable neural computer, which allows neural networks to dynamically read from and write to an external memory module. This enables lifelong lookup and forgetting of knowledge from external sources, as humans do.

Parisi *et al.* consider a different lifelong learning scenario in [552]. They abandon the memory modules in [550] and design a self-organizing architecture with recurrent neurons for processing time-varying patterns. A variant of the Growing When Required network is employed in each layer, to predict neural activation sequences from the previous network layer. This allows learning time-vary correlations between inputs and labels, without requiring a predefined number of classes. Importantly, the framework is robust, as it has tolerance to missing and corrupted sample labels, which is common in mobile data.

Another interesting deep lifelong learning architecture is presented in [553], where Tessler *et al.* build a DQN agent that can retain learned skills in playing the famous computer game Minecraft. The overall framework includes a pre-trained model, Deep Skill Network, which is trained a-priori on various sub-tasks of the game. When learning a new task, the old knowledge is maintained by incorporating reusable skills through a Deep Skill module, which consists of a Deep Skill Network array and a multi-skill distillation network. These allow the agent to selectively transfer knowledge to solve a new task. Experiments demonstrate that their proposal significantly outperforms traditional double DQNs in terms of accuracy and convergence. This technique has potential to be employed in solving mobile networking problems, as it can continuously acquire new knowledge.

Deep Transfer Learning: Unlike lifelong learning, transfer learning only seeks to use knowledge from a specific domain to aid learning in a target domain. Applying transfer learning can accelerate the new learning process, as the new task does not require to learn from scratch. This is essential to mobile network environments, as they require to agilely respond to new network patterns and threats. A number of important applications emerge in the computer network domain [57], such as Web mining [554], caching [555] and base station sleep strategies [209].

There exist two extreme transfer learning paradigms, namely one-shot learning and zero-shot learning. One-shot learning refers to a learning method that gains as much information as possible about a category from only one or a handful of samples, given a pre-trained model [556]. On the other hand, zero-shot learning does not require any sample from a category [557]. It aims at learning a new distribution given meta description of the new category and correlations with existing training data. Though research towards deep one-shot learning [96], [558] and deep zero-shot learning [559], [560] is in its infancy, both paradigms are very promising in detecting new threats or traffic patterns in mobile networks.

VIII. Future Research Perspectives

As deep learning is achieving increasingly promising results in the mobile networking domain, several important research issues remain to be addressed in the future. We conclude our survey by discussing these challenges and pinpointing key mobile networking research problems that could be tackled with novel deep learning tools.

A. Serving Deep Learning With Massive High-Quality Data

Deep neural networks rely on massive and high-quality data to achieve good performance. When training a large and complex architecture, data volume and quality are very important, as deeper models usually have a huge set of parameters to be learned and configured. This issue remains true in mobile network applications. Unfortunately, unlike in other research areas such as computer vision and NLP, high-quality and large-scale labeled datasets still lack for mobile network applications, because service provides and operators keep the data collected confidential and are reluctant to release datasets. While this makes sense from a user privacy standpoint, to some extent it restricts the development of deep learning mechanisms for problems in the mobile networking domain. Moreover, mobile data collected by sensors and network equipment are frequently subject to loss, redundancy, mislabeling and class imbalance, and thus cannot be directly employed for training purpose.

To build intelligent 5G mobile network architecture, efficient and mature streamlining platforms for mobile data processing are in demand. This requires considerable amount of research efforts for data collection, transmission, cleaning, clustering, transformation, and annonymization. Deep learning applications in the mobile network area can only advance if researchers and industry stakeholder release more datasets, with a view to benefiting a wide range of communities.

B. Deep Learning for Spatio-Temporal Mobile Data Mining

Accurate analysis of mobile traffic data over a geographical region is becoming increasingly essential for event localization, network resource allocation, context-based advertising and urban planning [548]. However, due to the mobility of smartphone users, the spatio-temporal distribution of mobile traffic [561] and application popularity [562] are difficult to understand (see the example city-scale traffic snapshot in Fig. 21). Recent research suggests that data collected by mobile sensors (e.g., mobile traffic) over a city can be regarded as pictures taken by panoramic cameras, which provide a city-scale sensing system for urban surveillance [564]. These traffic sensing images enclose information associated with the movements of individuals [468].

From both spatial and temporal dimensions perspective, we recognize that mobile traffic data have important similarity with videos or speech, which is an analogy made recently also in [218] and exemplified in Fig. 22. Specifically, both videos and the large-scale evolution of mobile traffic are composed of sequences of "frames". Moreover, if we zoom into a small coverage area to measure long-term traffic consumption, we can observe that a single traffic consumption series

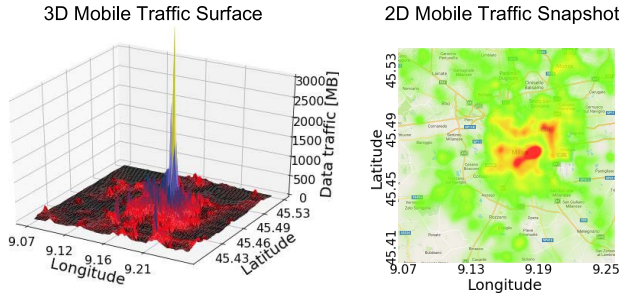


Fig. 21. Example of a 3D mobile traffic surface (left) and 2D projection (right) in Milan, Italy. Figures adapted from [218] using data from [563].

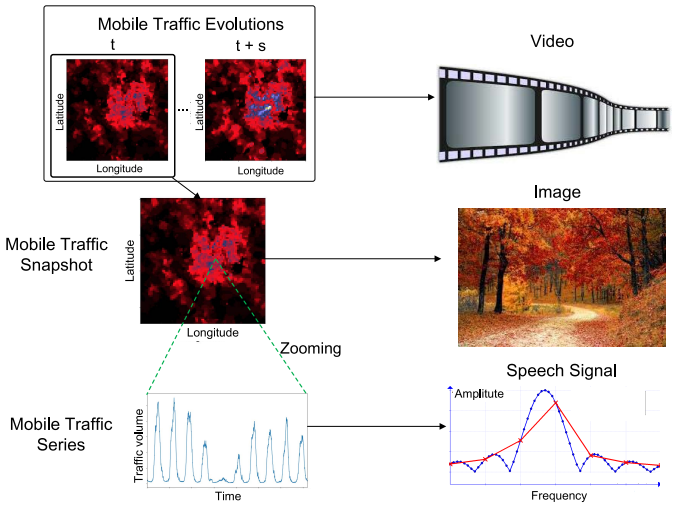


Fig. 22. Analogies between mobile traffic data consumption in a city (left) and other types of data (right).

looks similar to a natural language sequence. These observations suggest that, to some extent, well-established tools for computer vision (e.g., CNN) or NLP (e.g., RNN, LSTM) are promising candidate for mobile traffic analysis.

Beyond these similarity, we observe several properties of mobile traffic that makes it unique in comparison with images or language sequences. Namely,

1) The values of neighboring 'pixels' in fine-grained traffic snapshots are not significantly different in general, while this happens quite often at the edges of natural images.
2) Single mobile traffic series usually exhibit some periodicity (both daily and weekly), yet this is not a feature seen among video pixels.
3) Due to user mobility, traffic consumption is more likely to stay or shift to neighboring cells in the near future, which is less likely to be seen in videos.

Such spatio-temporal correlations in mobile traffic can be exploited as prior knowledge for model design. We recognize several unique advantages of employing deep learning for mobile traffic data mining:

1) CNN structures work well in imaging applications, thus can also serve mobile traffic analysis tasks, given the analogies mentioned before.

2) LSTMs capture well temporal correlations in time series data such as natural language; hence this structure can also be adapted to traffic forecasting problems.
3) GPU computing enables fast training of NNs and together with parallelization techniques can support low-latency mobile traffic analysis via deep learning tools.

In essence, we expect deep learning tools tailored to mobile networking, will overcome the limitation of traditional regression and interpolation tools such as Exponential Smoothing [565], Autoregressive Integrated Moving Average model [566], or unifrom interpolation, which are commonly used in operational networks.

C. Deep Learning for Geometric Mobile Data Mining

As discussed in Section III-D, certain mobile data has important geometric properties. For instance, the location of mobile users or base stations along with the data carried can be viewed as point clouds in a 2D plane. If the temporal dimension is also added, this leads to a 3D point cloud representation, with either fixed or changing locations. In addition, the connectivity of mobile devices, routers, base stations, gateways, and so on can naturally construct a directed graph, where entities are represented as vertices, the links between them can be seen as edges, and data flows may give direction to these edges. We show examples of geometric mobile data and their potential representations in Fig. 23. At the top of the figure a group of mobile users is represented as a point cloud. Likewise, mobile network entities (e.g., base station, gateway, users) are regarded as graphs below, following the rationale explained below. Due to the inherent complexity of such representations, traditional ML tools usually struggle to interpret geometric data and make reliable inferencess.

In contrast, a variety of deep learning toolboxes for modelling geometric data exist, albeit not having been widely employed in mobile networking yet. For instance, PointNet [567] and the follow on PointNet++ [101] are the first solutions that employ deep learning for 3D point cloud applications, including classification and segmentation [568]. We recognize that similar ideas can be applied to geometric mobile data analysis, such as clustering of mobile users or base stations, or user trajectory predictions. Further, deep learning for graphical data analysis is also evolving rapidly [569]. This is triggered by research on Graph CNNs [102], which brings convolution concepts to graph-structured data. The applicability of Graph CNNs can be further extend to the temporal domain [570]. One possible application is the prediction of future traffic demand at individual base station level. We expect that such novel architectures will play an increasingly important role in network graph analysis and applications such as anomaly detection over a mobile network graph.

D. Deep Unsupervised Learning in Mobile Networks

We observe that current deep learning practices in mobile networks largely employ supervised learning and reinforcement learning. However, as mobile networks generate considerable amounts of unlabeled data every day, data labeling is costly and requires domain-specific knowledge. To facilitate the analysis of raw mobile network data, unsupervised learning becomes essential in extracting insights from unlabeled data [571], so as to optimize the mobile network functionality to improve QoE.

The potential of a range of unsupervised deep learning tools including AE, RBM and GAN remains to be further explored. In general, these models require light feature engineering and are thus promising for learning from heterogeneous and unstructured mobile data. For instance, deep AEs work well for unsupervised anomaly detection [572]. Though less popular, RBMs can perform layer-wise unsupervised pre-training, which can accelerate the overall model training process. GANs are good at imitating data distributions, thus could be employed to mimic real mobile network environments. Recent research reveals that GANs can even protect communications by crafting custom cryptography to avoid eavesdropping [573]. All these tools require further research to fulfill their full potentials in the mobile networking domain.

E. Deep Reinforcement Learning for Mobile Network Control

Many mobile network control problems have been solved by constrained optimization, dynamic programming and game theory approaches. Unfortunately, these methods either make strong assumptions about the objective functions (e.g., function convexity) or data distribution (e.g., Gaussian or Poisson distributed), or suffer from high time and space complexity. As mobile networks become increasingly complex, such assumptions sometimes turn unrealistic. The objective functions are further affected by their increasingly large sets of variables, that pose severe computational and memory challenges to existing mathematical approaches.

In contrast, deep reinforcement learning does not make strong assumptions about the target system. It employs function approximation, which explicitly addresses the problem of large state-action spaces, enabling reinforcement learning to scale to network control problems that were previously considered hard. Inspired by remarkable achievements in Atari [19] and Go [574] games, a number of researchers begin to explore DRL to solve complex network control problems, as we discussed in Section VI-G. However, these works only scratch the surface and the potential of DRL to tackle mobile network control problems remains largely unexplored. For instance, as DeepMind trains a DRL agent to reduce Google's data centers cooling bill,[36] DRL could be exploited to extract rich features from cellular networks and enable intelligent on/off base stations switching, to reduce the infrastructure's energy footprint. Such exciting applications make us believe that advances in DRL that are yet to appear can revolutionize the autonomous control of future mobile networks.

[36]DeepMind AI Reduces Google Data Center Cooling Bill by 40% https://deepmind.com/blog/deepmind-ai-reduces-google-data-centre-cooling-bill-40/.

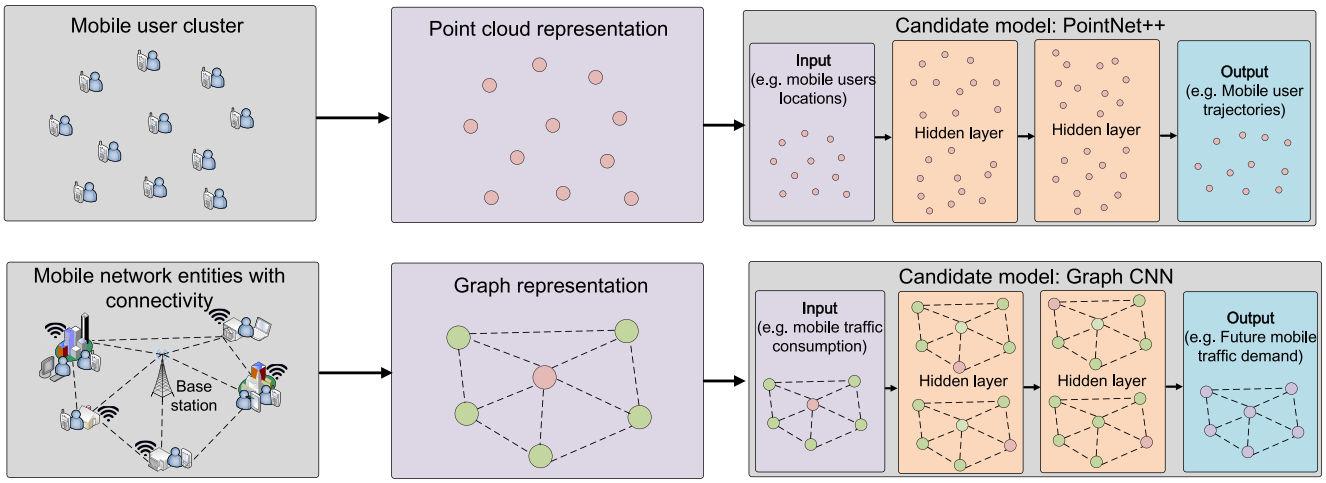


Fig. 23. Examples of mobile data with geometric properties (left), their geometric representations (middle) and their candidate models for analysis (right). PointNet++ could be used to infer user trajectories when fed with point cloud representations of user locations (above); A GraphCNN may be employed to forecast future mobile traffic demand at base station level (below).

F. Summary

Deep learning is playing an increasingly important role in the mobile and wireless networking domain. In this paper, we provided a comprehensive survey of recent work that lies at the intersection between deep learning and mobile networking. We summarized both basic concepts and advanced principles of various deep learning models, then reviewed work specific to mobile networks across different application scenarios. We discussed how to tailor deep learning models to general mobile networking applications, an aspect overlooked by previous surveys. We concluded by pinpointing several open research issues and promising directions, which may lead to valuable future research results. Our hope is that this article will become a definite guide to researchers and practitioners interested in applying machine intelligence to complex problems in mobile network environments.

ACKNOWLEDGMENT

The authors would like to thank Zongzuo Wang for sharing valuable insights on deep learning, which helped improving the quality of this paper. They also thank the anonymous reviewers and editors, whose detailed and thoughtful feedback helped them give this survey more depth and a broader scope.

REFERENCES

[1] *Cisco Visual Networking Index: Forecast and Methodology, 2016–2021*, Cisco, San Jose, CA, USA, Jun. 2017.
[2] N. Wang, E. Hossain, and V. K. Bhargava, "Backhauling 5G small cells: A radio resource management perspective," *IEEE Wireless Commun.*, vol. 22, no. 5, pp. 41–49, Oct. 2015.
[3] F. Giust, L. Cominardi, and C. J. Bernardos, "Distributed mobility management for future 5G networks: Overview and analysis of existing approaches," *IEEE Commun. Mag.*, vol. 53, no. 1, pp. 142–149, Jan. 2015.
[4] M. Agiwal, A. Roy, and N. Saxena, "Next generation 5G wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 3, pp. 1617–1655, 3rd Quart., 2016.
[5] A. Gupta and R. K. Jha, "A survey of 5G network: Architecture and emerging technologies," *IEEE Access*, vol. 3, pp. 1206–1232, 2015.
[6] K. Zheng *et al.*, "Big data-driven optimization for mobile networks toward 5G," *IEEE Netw.*, vol. 30, no. 1, pp. 44–51, Jan./Feb. 2016.
[7] C. Jiang *et al.*, "Machine learning paradigms for next-generation wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 2, pp. 98–105, Apr. 2017.
[8] D. D. Nguyen, H. X. Nguyen, and L. B. White, "Reinforcement learning with network-assisted feedback for heterogeneous RAT selection," *IEEE Trans. Wireless Commun.*, vol. 16, no. 9, pp. 6062–6076, Sep. 2017.
[9] F. A. Narudin, A. Feizollah, N. B. Anuar, and A. Gani, "Evaluation of machine learning classifiers for mobile malware detection," *Soft Comput.*, vol. 20, no. 1, pp. 343–357, 2016.
[10] K. Hsieh *et al.*, "Gaia: Geo-distributed machine learning approaching LAN speeds," in *Proc. USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2017, pp. 629–647.
[11] W. Xiao *et al.*, "Tux^2: Distributed graph computation for machine learning," in *Proc. USENIX Symp. Netw. Syst. Design Implement. (NSDI)*, 2017, pp. 669–682.
[12] M. Paolini and S. Fili, *Mastering Analytics: How to Benefit From Big Data and Network Complexity: An Analyst Report*, RCR Wireless News, London, U.K., 2017.
[13] C. Zhang, P. Zhou, C. Li, and L. Liu, "A convolutional neural network for leaves recognition using data augmentation," in *Proc. IEEE Int. Conf. Pervasive Intell. Comput. (PICOM)*, Liverpool, U.K., 2015, pp. 2143–2150.
[14] R. Socher, Y. Bengio, and C. D. Manning, "Deep learning for NLP (without magic)," in *Proc. Tuts. Abstracts ACL*, 2012, p. 5.
[15] (2017). *IEEE Network Special Issue: Exploring Deep Learning for Efficient and Reliable Mobile Sensing*. Accessed: Jul. 14, 2017. [Online]. Available: http://www.comsoc.org/netmag/cfp/exploring-deep-learning-efficient-and-reliable-mobile-sensing
[16] M. Wang, Y. Cui, X. Wang, S. Xiao, and J. Jiang, "Machine learning for networking: WorkFlow, advances and opportunities," *IEEE Netw.*, vol. 32, no. 2, pp. 92–99, Mar./Apr. 2018.
[17] 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.
[18] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016.
[19] V. Mnih *et al.*, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, no. 7540, pp. 529–533, 2015.
[20] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.
[21] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.
[22] W. Liu *et al.*, "A survey of deep neural network architectures and their applications," *Neurocomputing*, vol. 234, pp. 11–26, Apr. 2017.
[23] L. Deng and D. Yu, "Deep learning: Methods and applications," *Found.® Trends Signal Process.*, vol. 7, nos. 3–4, pp. 197–387, 2014.

[24] 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.

[25] S. Pouyanfar *et al.*, "A survey on deep learning: Algorithms, techniques, and applications," *ACM Comput. Surveys* vol. 52, no. 5, pp. 1–92, 2018.

[26] K. Arulkumaran, M. P. Deisenroth, M. Brundage, and A. A. Bharath, "Deep reinforcement learning: A brief survey," *IEEE Signal Process. Mag.*, vol. 34, no. 6, pp. 26–38, Nov. 2017.

[27] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Comput. Surveys*, vol. 50, no. 2, pp. 1–21, 2017.

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

[29] M. M. Najafabadi *et al.*, "Deep learning applications and challenges in big data analytics," *J. Big Data*, vol. 2, no. 1, p. 1, 2015.

[30] N. F. Hordri, A. Samar, S. S. Yuhaniz, and S. M. Shamsuddin, "A systematic literature review on features of deep learning in big data analytics," *Int. J. Adv. Soft Comput. Appl.*, vol. 9, no. 1, pp. 32–49, 2017.

[31] M. Gheisari, G. Wang, and M. Z. A. Bhuiyan, "A survey on deep learning in big data," in *Proc. IEEE Int. Conf. Comput. Sci. Eng. (CSE) Embedded Ubiquitous Comput. (EUC)*, vol. 2. Guangzhou, China, 2017, pp. 173–180.

[32] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Comput. Surveys*, vol. 52, no. 1, pp. 1–5, 2019.

[33] S. Yu, M. Liu, W. Dou, X. Liu, and S. Zhou, "Networking for big data: A survey," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 1, pp. 531–549, 1st Quart., 2017.

[34] 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.

[35] 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.

[36] X. Cheng, L. Fang, X. Hong, and L. Yang, "Exploiting mobile big data: Sources, features, and applications," *IEEE Netw.*, vol. 31, no. 1, pp. 72–79, Jan./Feb. 2017.

[37] M. Bkassiny, Y. Li, and S. K. Jayaweera, "A survey on machine-learning techniques in cognitive radios," *IEEE Commun. Surveys Tuts.*, vol. 15, no. 3, pp. 1136–1159, 3rd Quart., 2013.

[38] J. G. Andrews *et al.*, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[39] N. Panwar, S. Sharma, and A. K. Singh, "A survey on 5G: The next generation of mobile communication," *Phys. Commun.*, vol. 18, pp. 64–84, Mar. 2016.

[40] O. Elijah, C. Y. Leow, T. A. Rahman, S. Nunoo, and S. Z. Iliya, "A comprehensive survey of pilot contamination in massive MIMO—5G system," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 905–923, 2nd Quart., 2016.

[41] S. Buzzi *et al.*, "A survey of energy-efficient techniques for 5G networks and challenges ahead," *IEEE J. Sel. Areas Commun.*, vol. 34, no. 4, pp. 697–709, Apr. 2016.

[42] M. Peng, Y. Li, Z. Zhao, and C. Wang, "System architecture and key technologies for 5G heterogeneous cloud radio access networks," *IEEE Netw.*, vol. 29, no. 2, pp. 6–14, Mar./Apr. 2015.

[43] Y. Niu, Y. Li, D. Jin, L. Su, and A. V. Vasilakos, "A survey of millimeter wave communications (mmWave) for 5G: Opportunities and challenges," *Wireless Netw.*, vol. 21, no. 8, pp. 2657–2676, 2015.

[44] X. Foukas, G. Patounas, A. Elmokashfi, and M. K. Marina, "Network slicing in 5G: Survey and challenges," *IEEE Commun. Mag.*, vol. 55, no. 5, pp. 94–100, May 2017.

[45] T. Taleb *et al.*, "On multi-access edge computing: A survey of the emerging 5G network edge cloud architecture and orchestration," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1657–1681, 3rd Quart., 2017.

[46] P. Mach and Z. Becvar, "Mobile edge computing: A survey on architecture and computation offloading," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1628–1656, 3rd Quart., 2017.

[47] Y. Mao, C. You, J. Zhang, K. Huang, and K. B. Letaief, "A survey on mobile edge computing: The communication perspective," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2322–2358, 4th Quart., 2017.

[48] Y. Wang *et al.*, "A data-driven architecture for personalized QoE management in 5G wireless networks," *IEEE Wireless Commun.*, vol. 24, no. 1, pp. 102–110, Feb. 2017.

[49] Q. Han, S. Liang, and H. Zhang, "Mobile cloud sensing, big data, and 5G networks make an intelligent and smart world," *IEEE Netw.*, vol. 29, no. 2, pp. 40–45, Mar./Apr. 2015.

[50] S. Singh, N. Saxena, A. Roy, and H. S. Kim, "A survey on 5G network technologies from social perspective," *IETE Tech. Rev.*, vol. 34, no. 1, pp. 30–39, 2017.

[51] M. Chen, J. Yang, Y. Hao, S. Mao, and K. Hwang, "A 5G cognitive system for healthcare," *Big Data Cogn. Comput.*, vol. 1, no. 1, pp. 1–15, 2017.

[52] X. Chen, J. Wu, Y. Cai, H. Zhang, and T. Chen, "Energy-efficiency oriented traffic offloading in wireless networks: A brief survey and a learning approach for heterogeneous cellular networks," *IEEE J. Sel. Areas Commun.*, vol. 33, no. 4, pp. 627–640, Apr. 2015.

[53] J. Wu, S. Guo, J. Li, and D. Zeng, "Big data meet green challenges: Big data toward green applications," *IEEE Syst. J.*, vol. 10, no. 3, pp. 888–900, Sep. 2016.

[54] T. S. Buda *et al.*, "Can machine learning aid in delivering new use cases and scenarios in 5G?" in *Proc. IEEE/IFIP Netw. Oper. Manag. Symp. (NOMS)*, Istanbul, Turkey, 2016, pp. 1279–1284.

[55] A. Imran, A. Zoha, and A. Abu-Dayya, "Challenges in 5G: How to empower SON with big data for enabling 5G," *IEEE Netw.*, vol. 28, no. 6, pp. 27–33, Nov./Dec. 2014.

[56] B. Keshavamurthy and M. Ashraf, "Conceptual design of proactive SONs based on the big data framework for 5G cellular networks: A novel machine learning perspective facilitating a shift in the SON paradigm," in *Proc. IEEE Int. Conf. Syst. Model. Adv. Res. Trends (SMART)*, Moradabad, India, 2016, pp. 298–304.

[57] 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.

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

[59] N. Bui *et al.*, "A survey of anticipatory mobile networking: Context-based classification, prediction methodologies, and optimization techniques," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 3, pp. 1790–1821, 3rd Quart., 2017.

[60] R. Atat *et al.*, "Big data meet cyber-physical systems: A panoramic survey," *IEEE Access*, vol. 6, pp. 73603–73636, 2018.

[61] X. Cheng, L. Fang, L. Yang, and S. Cui, "Mobile big data: The fuel for data-driven wireless," *IEEE Internet Things J.*, vol. 4, no. 5, pp. 1489–1516, Oct. 2017.

[62] P. Kasnesis, C. Patrikakis, and I. Venieris, "Changing the game of mobile data analysis with deep learning," *IT Prof.*, to be published.

[63] L. Wang and R. Jones, "Big data analytics for network intrusion detection: A survey," *Int. J. Netw. Commun.*, vol. 7, no. 1, pp. 24–31, 2017.

[64] N. Kato *et al.*, "The deep learning vision for heterogeneous network traffic control: Proposal, challenges, and future perspective," *IEEE Wireless Commun.*, vol. 24, no. 3, pp. 146–153, Jun. 2017.

[65] M. Zorzi, A. Zanella, A. Testolin, M. D. F. D. Grazia, and M. Zorzi, "Cognition-based networks: A new perspective on network optimization using learning and distributed intelligence," *IEEE Access*, vol. 3, pp. 1512–1530, 2015.

[66] Z. M. 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.

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

[68] 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.

[69] Q. Mao, F. Hu, and Q. Hao, "Deep learning for intelligent wireless networks: A comprehensive survey," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 4, pp. 2595–2621, 4th Quart., 2018.

[70] N. C. Luong *et al.*, "Applications of deep reinforcement learning in communications and networking: A survey," *arXiv preprint arXiv:1810.07862*, 2018.

[71] X. Zhou, M. Sun, Y. G. Li, and B.-H. F. Juang, "Intelligent wireless communications enabled by cognitive radio and machine learning," *China Commun.*, vol. 15, no. 12, pp. 16–48, Dec. 2018.

[72] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Machine learning for wireless networks with artificial intelligence: A tutorial on neural networks," *arXiv preprint arXiv:1710.02913*, 2017.

[73] A. Gharaibeh *et al.*, "Smart cities: A survey on data management, security, and enabling technologies," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2456–2501, 4th Quart., 2017.

[74] N. D. Lane and P. Georgiev, "Can deep learning revolutionize mobile sensing?" in *Proc. 16th ACM Int. Workshop Mobile Comput. Syst. Appl.*, Santa Fe, NM, USA, 2015, pp. 117–122.

[75] K. Ota, M. S. Dao, V. Mezaris, and F. G. B. De Natale, "Deep learning for mobile multimedia: A survey," *ACM Trans. Multimedia Comput. Commun. Appl. (TOMM)*, vol. 13, no. 3S, p. 34, 2017.

[76] P. Mishra, V. Varadharajan, U. Tupakula, and E. S. Pilli, "A detailed investigation and analysis of using machine learning techniques for intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 686–728, 1st Quart., 2019.

[77] Y. Li, "Deep reinforcement learning: An overview," *arXiv preprint arXiv:1701.07274*, 2017.

[78] L. Chen *et al.*, "Deep mobile traffic forecast and complementary base station clustering for C-RAN optimization," *J. Netw. Comput. Appl.*, vol. 121, pp. 59–69, Nov. 2018.

[79] V. Mnih *et al.*, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2016, pp. 1928–1937.

[80] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 214–223.

[81] A. Damianou and N. Lawrence, "Deep Gaussian processes," in *Proc. Artif. Intell. Stat.*, Stellenbosch, South Africa, 2013, pp. 207–215.

[82] M. Garnelo *et al.*, "Neural processes," *arXiv preprint arXiv:1807.01622*, 2018.

[83] Z.-H. Zhou and J. Feng, "Deep forest: Towards an alternative to deep neural networks," in *Proc. 26th Int. Joint Conf. Artif. Intell.*, Nanjing, China, 2017, pp. 3553–3559.

[84] W. Mcculloch and W. Pitts, "A logical calculus of the ideas immanent in nervous activity," *Bull. Math. Biophys.*, vol. 52, nos. 1–2, pp. 99–115, 1990.

[85] 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.

[86] Y. Lecun and Y. Bengio, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*, vol. 3361. Cambridge, MA, USA: MIT Press, 1995.

[87] 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.

[88] P. Domingos, "A few useful things to know about machine learning," *Commun. ACM*, vol. 55, no. 10, pp. 78–87, 2012.

[89] I. W. Tsang, J. T. Kwok, and P.-M. Cheung, "Core vector machines: Fast SVM training on very large data sets," *J. Mach. Learn. Res.*, vol. 6, pp. 363–392, 2005.

[90] C. E. Rasmussen and C. K. Williams, *Gaussian Processes for Machine Learning*, vol. 1. Cambridge, MA, USA: MIT Press, 2006.

[91] N. Le Roux and Y. Bengio, "Representational power of restricted Boltzmann machines and deep belief networks," *Neural Comput.*, vol. 20, no. 6, pp. 1631–1649, Jun. 2008.

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

[93] F. Schroff, D. Kalenichenko, and J. Philbin, "FaceNet: A unified embedding for face recognition and clustering," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Boston, MA, USA, 2015, pp. 815–823.

[94] 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.*, Montreal, QC, Canada, 2014, pp. 3581–3589.

[95] R. Stewart and S. Ermon, "Label-free supervision of neural networks with physics and domain knowledge," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2017, pp. 2576–2582.

[96] D. J. Rezende, S. Mohamed, I. Danihelka, K. Gregor, and D. Wierstra, "One-shot generalization in deep generative models," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2016, pp. 1521–1529.

[97] R. Socher, M. Ganjoo, C. D. Manning, and A. Y. Ng, "Zero-shot learning through cross-modal transfer," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 935–943.

[98] P. Georgiev, S. Bhattacharya, N. D. Lane, and C. Mascolo, "Low-resource multi-task audio sensing for mobile and embedded devices via shared deep neural network representations," in *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol. (IMWUT)*, vol. 1, 2017, p. 50.

[99] F. Monti *et al.*, "Geometric deep learning on graphs and manifolds using mixture model CNNs," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1, 2017, p. 3.

[100] B. L. Roux and H. Rouanet, *Geometric Data Analysis: From Correspondence Analysis to Structured Data Analysis*. Dordrecht, The Netherlands: Springer, 2004.

[101] C. R. Qi, L. Yi, H. Su, and L. J. Guibas, "PointNet++: Deep hierarchical feature learning on point sets in a metric space," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 5099–5108.

[102] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2017, pp. 1–14.

[103] X. Wang, Z. Zhou, Z. Yang, Y. Liu, and C. Peng, "Spatio-temporal analysis and prediction of cellular traffic in metropolis," *IEEE Trans. Mobile Comput.*, to be published.

[104] 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.*, 2015, pp. 427–436.

[105] V. Behzadan and A. Munir, "Vulnerability of deep reinforcement learning to policy induction attacks," in *Proc. Int. Conf. Mach. Learn. Data Min. Pattern Recognit.*, 2017, pp. 262–275.

[106] P. Madani and N. Vlajic, "Robustness of deep autoencoder in intrusion detection under adversarial contamination," in *Proc. 5th ACM Annu. Symp. Bootcamp Hot Topics Sci. Security*, 2018, p. 1.

[107] D. Bau, B. Zhou, A. Khosla, A. Oliva, and A. Torralba, "Network dissection: Quantifying interpretability of deep visual representations," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 3319–3327.

[108] M. Wu *et al.*, "Beyond sparsity: Tree regularization of deep models for interpretability," in *Proc. 32nd AAAI Conf. Artif. Intell. (AAAI)*, 2018, pp. 1670–1678.

[109] S. Chakraborty *et al.*, "Interpretability of deep learning models: A survey of results," in *Proc. IEEE Smart World Congr. Workshop DAIS*, 2017, pp. 1–6.

[110] L. Perez and J. Wang, "The effectiveness of data augmentation in image classification using deep learning," *arXiv preprint arXiv:1712.04621*, 2017.

[111] C. Liu *et al.*, "Progressive neural architecture search," in *Proc. Eur. Conf. Comput. Vis. (ECCV)*, 2018, pp. 1–16.

[112] 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.

[113] 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.

[114] E. D. Coninck *et al.*, "Distributed neural networks for Internet of Things: The big-little approach," in *Proc. 2nd Int. Summit Internet Things IoT Infrastructures*, Rome, Italy, Oct. 2016, pp. 484–492.

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

[116] J. Nickolls, I. Buck, M. Garland, and K. Skadron, "Scalable parallel programming with CUDA," *Queue*, vol. 6, no. 2, pp. 40–53, 2008.

[117] S. Chetlur *et al.*, "cuDNN: Efficient primitives for deep learning," *arXiv preprint arXiv:1410.0759*, 2014.

[118] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," in *Proc. USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, vol. 16, 2016, pp. 265–283.

[119] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints abs/1605.02688*, May 2016.

[120] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," *arXiv preprint arXiv:1408.5093*, 2014.

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

[122] V. Gokhale, J. Jin, A. Dundar, B. Martini, and E. Culurciello, "A 240 G-Ops/s mobile coprocessor for deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2014, pp. 682–687.

[123] (2017). *NCNN—A High-Performance Neural Network Inference Framework Optimized for the Mobile Platform*. Accessed: Jul. 25, 2017. [Online]. Available: https://github.com/Tencent/ncnn

[124] (2017). *Huawei Announces the Kirin 970–New Flagship SoC With AI Capabilities*. Accessed: Sep. 1, 2017. [Online]. Available: http://www.androidauthority.com/huawei-announces-kirin-970-797788/

[125] (2017). *Core ML: Integrate Machine Learning Models Into Your App*. Accessed: Jul. 25, 2017. [Online]. Available: https://developer.apple.com/documentation/coreml

[126] I. Sutskever, J. Martens, G. E. Dahl, and G. E. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, vol. 28, 2013, pp. 1139–1147.

[127] J. Dean *et al.*, "Large scale distributed deep networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1223–1231.

[128] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2015, pp. 1–15.

[129] T. Kraska *et al.*, "$MLbase$: A distributed machine-learning system," in *Proc. CIDR*, vol. 1, 2013, pp. 1–7.

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

[131] H. Cui, H. Zhang, G. R. Ganger, P. B. Gibbons, and E. P. Xing, "GeePS: Scalable deep learning on distributed GPUs with a GPU-specialized parameter server," in *Proc. 11th ACM Eur. Conf. Comput. Syst.*, 2016, p. 4.

[132] X. Lin *et al.*, "All-optical machine learning using diffractive deep neural networks," *Science*, vol. 361, no. 6406, pp. 1004–1008, 2018.

[133] R. Spring and A. Shrivastava, "Scalable and sustainable deep learning via randomized hashing," in *Proc. ACM SIGKDD Conf. Knowl. Disc. Data Min.*, 2017, pp. 445–454.

[134] A. Mirhoseini *et al.*, "Device placement optimization with reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, 2017, pp. 1–11.

[135] E. P. Xing *et al.*, "Petuum: A new platform for distributed machine learning on big data," *IEEE Trans. Big Data*, vol. 1, no. 2, pp. 49–67, Jun. 2015.

[136] P. Moritz *et al.*, "Ray: A distributed framework for emerging AI applications," in *Proc. 13th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, 2018, pp. 561–577.

[137] M. Alzantot, Y. Wang, Z. Ren, and M. B. Srivastava, "RSTensorFlow: GPU enabled TensorFlow for deep learning on commodity Android devices," in *Proc. 1st ACM Int. Workshop Deep Learn. Mobile Syst. Appl.*, 2017, pp. 7–12.

[138] H. Dong *et al.*, "TensorLayer: A versatile library for efficient deep learning development," in *Proc. ACM Multimedia Conf. (MM)*, 2017, pp. 1201–1204.

[139] A. Paszke *et al.*, "Automatic differentiation in PyTorch," 2017.

[140] T. Chen *et al.*, "MXNet: A flexible and efficient machine learning library for heterogeneous distributed systems," *arXiv preprint arXiv:1512.01274*, 2015.

[141] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[142] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[143] T. Dozat, "Incorporating Nesterov momentum into Adam," in *Proc. Workshop Track (ICLR)*, 2016, pp. 1–4.

[144] M. Andrychowicz *et al.*, "Learning to learn by gradient descent by gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, 2016, pp. 3981–3989.

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

[146] Y. Zhou, S. Chen, and A. Banerjee, "Stable gradient descent," in *Proc. Conf. Uncertainty Artif. Intell.*, 2018, pp. 1–10.

[147] W. Wen *et al.*, "TernGrad: Ternary gradients to reduce communication in distributed deep learning," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1–11.

[148] F. Bonomi, R. Milito, P. Natarajan, and J. Zhu, "Fog computing: A platform for Internet of Things and analytics," in *Proc. Big Data Internet Things: A Roadmap Smart Environ.*, 2014, pp. 169–186.

[149] J. Mao, X. Chen, K. W. Nixon, C. Krieger, and Y. Chen, "MoDNN: Local distributed mobile computing system for deep neural network," in *Proc. IEEE Design Autom. Test Europe Conf. Exhibit. (DATE)*, 2017, pp. 1396–1401.

[150] M. Mukherjee, L. Shu, and D. Wang, "Survey of fog computing: Fundamental, network applications, and research challenges," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 3, pp. 1826–1857, 3rd Quart., 2018.

[151] L. M. Vaquero and L. Rodero-Merino, "Finding your way in the fog: Towards a comprehensive definition of fog computing," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 5, pp. 27–32, 2014.

[152] M. Aazam, S. Zeadally, and K. A. Harras, "Offloading in fog computing for IoT: Review, enabling technologies, and research opportunities," *Future Gener. Comput. Syst.*, vol. 87, pp. 278–289, Oct. 2018.

[153] R. Buyya *et al.*, "A manifesto for future generation cloud computing: Research directions for the next decade," *ACM Comput. Survey*, vol. 51, no. 5, pp. 1–105, 2018.

[154] V. Sze, Y.-H. Chen, T.-J. Yang, and J. S. Emer, "Efficient processing of deep neural networks: A tutorial and survey," *Proc. IEEE*, vol. 105, no. 12, pp. 2295–2329, Dec. 2017.

[155] 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. Conf. Solid-State Circuits (ISSCC)*, 2017, pp. 250–251.

[156] F. Akopyan, "Design and tool flow of IBM's TrueNorth: An ultra-low power programmable neurosynaptic chip with 1 million neurons," in *Proc. ACM Int. Symp. Phys. Design*, 2016, pp. 59–60.

[157] S. S. L. Oskouei, H. Golestani, M. Hashemi, and S. Ghiasi, "CNNdroid: GPU-accelerated execution of trained deep convolutional neural networks on Android," in *Proc. ACM Multimedia Conf.*, 2016, pp. 1201–1205.

[158] C. Cortes, X. Gonzalvo, V. Kuznetsov, M. Mohri, and S. Yang, "AdaNet: Adaptive structural learning of artificial neural networks," in *Proc. Int. Conf. Mach. Learn. (ICML)*, 2017, pp. 874–883.

[159] G. E. Hinton, S. Osindero, and Y.-W. Teh, "A fast learning algorithm for deep belief nets," *Neural Comput.*, vol. 18, no. 7, pp. 1527–1554, 2006.

[160] H. Lee, R. Grosse, R. Ranganath, and A. Y. Ng, "Convolutional deep belief networks for scalable unsupervised learning of hierarchical representations," in *Proc. 26th ACM Annu. Int. Conf. Mach. Learn.*, 2009, pp. 609–616.

[161] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *J. Mach. Learn. Res.*, vol. 11, pp. 3371–3408, Dec. 2010.

[162] D. P. Kingma and M. Welling, "Auto-encoding variational Bayes," in *Proc. Int. Conf. Learn. Representations (ICLR)*, 2014.

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

[164] S. Ji, W. Xu, M. Yang, and K. Yu, "3D convolutional neural networks for human action recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 1, pp. 221–231, Jan. 2013.

[165] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten, "Densely connected convolutional networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2017, pp. 2261–2269.

[166] F. A. Gers, J. Schmidhuber, and F. Cummins, "Learning to forget: Continual prediction with LSTM," in *Proc. 9th Int. Conf. Artif. Neural Netw. (ICANN)*, 1999, pp. 850–855.

[167] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3104–3112.

[168] S. Xingjian *et al.*, "Convolutional LSTM network: A machine learning approach for precipitation nowcasting," in *Proc. Adv. Neural Inf. Process. Syst.*, 2015, pp. 802–810.

[169] G.-J. Qi, "Loss-sensitive generative adversarial networks on Lipschitz densities," *IEEE Trans. Pattern Anal. Mach. Intell.*, 2018, pp. 1–34.

[170] A. Brock, J. Donahue, and K. Simonyan, "Large scale GAN training for high fidelity natural image synthesis," *arXiv preprint arXiv:1809.11096*, 2018.

[171] D. Silver *et al.*, "Mastering the game of GO with deep neural networks and tree search," *Nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[172] M. Hessel *et al.*, "Rainbow: Combining improvements in deep reinforcement learning," in *Proc. 32nd AAAI Conf. Artif. Intell. (AAAI)*, 2017, pp. 3215–3222.

[173] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[174] R. Collobert and S. Bengio, "Links between perceptrons, MLPs and SVMs," in *Proc. 21st ACM Int. Conf. Mach. Learn.*, Banff, AB, Canada, 2004, p. 23.

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

[176] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 971–980.

[177] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. Int. Conf. Mach. Learn.*, Long Beach, CA, USA, 2015, pp. 448–456.

[178] G. E. Hinton, "Training products of experts by minimizing contrastive divergence," *Neural Comput.*, vol. 14, no. 8, pp. 1771–1800, 2002.

[179] G. Casella and E. I. George, "Explaining the Gibbs sampler," *Amer. Stat.*, vol. 46, no. 3, pp. 167–174, 1992.

[180] T. Kuremoto, M. Obayashi, K. Kobayashi, T. Hirata, and S. Mabu, "Forecast chaotic time series data by DBNs," in *Proc. 7th IEEE Int. Congr. Image Signal Process. (CISP)*, Dalian, China, 2014, pp. 1130–1135.

[181] Y. Dauphin and Y. Bengio, "Stochastic ratio matching of RBMs for sparse high-dimensional inputs," in *Proc. Adv. Neural Inf. Process. Syst.*, 2013, pp. 1340–1348.

[182] T. N. Sainath *et al.*, "Making deep belief networks effective for large vocabulary continuous speech recognition," in *Proc. IEEE Workshop Autom. Speech Recognit. Understanding (ASRU)*, Waikoloa, HI, USA, 2011, pp. 30–35.

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

[184] M. Sakurada and T. Yairi, "Anomaly detection using autoencoders with nonlinear dimensionality reduction," in *Proc. ACM Workshop Mach. Learn. Sensory Data Anal. (MLSDA)*, 2014, p. 4.

[185] V. L. Cao, M. Nicolau, and J. McDermott, "A hybrid autoencoder and density estimation model for anomaly detection," in *Proc. Int. Conf. Parallel Probl. Solving Nat.*, 2016, pp. 717–726.

[186] V. L. L. Thing, "IEEE 802.11 network anomaly detection and attack classification: A deep learning approach," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, 2017, pp. 1–6.

[187] B. Mao *et al.*, "Routing or computing? The paradigm shift towards intelligent computer network packet transmission based on deep learning," *IEEE Trans. Comput.*, vol. 66, no. 11, pp. 1946–1960, Nov. 2017.

[188] V. Radu *et al.*, "Towards multimodal deep learning for activity recognition on mobile devices," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput. Adjunct*, 2016, pp. 185–188.

[189] V. Radu *et al.*, "Multimodal deep learning for activity and context recognition," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 4, p. 157, 2018.

[190] R. Raghavendra and C. Busch, "Learning deeply coupled autoencoders for smartphone based robust periocular verification," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Phoenix, AZ, USA, 2016, pp. 325–329.

[191] J. Li, J. Wang, and Z. Xiong, "Wavelet-based stacked denoising autoencoders for cell phone base station user number prediction," in *Proc. IEEE Int. Conf. Internet Things (iThings) IEEE Green Comput. Commun. (GreenCom) IEEE Cyber Phys. Soc. Comput. (CPSCom) IEEE Smart Data (SmartData)*, Chengdu, China, 2016, pp. 833–838.

[192] O. Russakovsky *et al.*, "ImageNet large scale visual recognition challenge," *Int. J. Comput. Vis.*, vol. 115, no. 3, pp. 211–252, 2015.

[193] Y. Jeon and J. Kim, "Active convolution: Learning the shape of convolution for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 1846–1854.

[194] J. Dai *et al.*, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, Venice, Italy, 2017, pp. 764–773.

[195] X. Zhu, H. Hu, S. Lin, and J. Dai, "Deformable ConvNets v2: More deformable, better results," *arXiv preprint arXiv:1811.11168*, 2018.

[196] Y. Bengio, P. Simard, and P. Frasconi, "Learning long-term dependencies with gradient descent is difficult," *IEEE Trans. Neural Netw.*, vol. 5, no. 2, pp. 157–166, Mar. 1994.

[197] A. Graves, N. Jaitly, and A.-R. Mohamed, "Hybrid speech recognition with deep bidirectional LSTM," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand. (ASRU)*, Olomouc, Czechia, 2013, pp. 273–278.

[198] R. Johnson and T. Zhang, "Supervised and semi-supervised text categorization using LSTM for region embeddings," in *Proc. Int. Conf. Mach. Learn. (ICML)*, New York, NY, USA, 2016, pp. 526–534.

[199] I. Goodfellow, "NIPS 2016 tutorial: Generative adversarial networks," *arXiv preprint arXiv:1701.00160*, 2016.

[200] C. Ledig *et al.*, "Photo-realistic single image super-resolution using a generative adversarial network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 105–114.

[201] J. Li *et al.*, "Perceptual generative adversarial networks for small object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 1951–1959.

[202] Y. Li, S. Liu, J. Yang, and M.-H. Yang, "Generative face completion," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Honolulu, HI, USA, 2017, pp. 5892–5900.

[203] P. Gawłowicz and A. Zubow, "ns3-gym: Extending OpenAI Gym for networking research," *arXiv preprint arXiv:1810.03943*, 2018.

[204] S. Gu, T. Lillicrap, I. Sutskever, and S. Levine, "Continuous deep Q-learning with model-based acceleration," in *Proc. Int. Conf. Mach. Learn.*, New York, NY, USA, 2016, pp. 2829–2838.

[205] M. Moravčík *et al.*, "DeepStack: Expert-level artificial intelligence in heads-up no-limit poker," *Science*, vol. 356, no. 6337, pp. 508–513, 2017.

[206] S. Levine, P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen, "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *Int. J. Robot. Res.*, vol. 37, nos. 4–5, pp. 421–436, 2018.

[207] A. El Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electron. Imag.*, vol. 19, pp. 70–76, Jan. 2017.

[208] DeepMind. (2019). *AlphaStar: Mastering the Real-Time Strategy Game StarCraft II*. Accessed: Mar. 1, 2019. [Online]. Available: https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/

[209] R. Li, Z. Zhao, X. Chen, J. Palicot, and H. Zhang, "TACT: A transfer actor–critic learning framework for energy saving in cellular radio access networks," *IEEE Trans. Wireless Commun.*, vol. 13, no. 4, pp. 2000–2011, Apr. 2014.

[210] H. A. A. Al-Rawi, M. A. Ng, and K.-L. A. Yau, "Application of reinforcement learning to routing in distributed wireless networks: A review," *Artif. Intell. Rev.*, vol. 43, no. 3, pp. 381–416, 2015.

[211] Y.-J. Liu, L. Tang, S. Tong, C. L. P. Chen, and D.-J. Li, "Reinforcement learning design-based adaptive tracking control with less learning parameters for nonlinear discrete-time MIMO systems," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 26, no. 1, pp. 165–176, Jan. 2015.

[212] L. Pierucci and D. Micheli, "A neural network for quality of experience estimation in mobile communications," *IEEE MultiMedia*, vol. 23, no. 4, pp. 42–49, Oct./Dec. 2016.

[213] Y. L. Gwon and H. T. Kung, "Inferring origin flow patterns in Wi-Fi with deep learning," in *Proc. 11th IEEE Int. Conf. Auton. Comput. (ICAC))*, 2014, pp. 73–83.

[214] L. Nie, D. Jiang, S. Yu, and H. Song, "Network traffic prediction based on deep belief network in wireless mesh backbone networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, San Francisco, CA, USA, 2017, pp. 1–5.

[215] V. Moyo *et al.*, "The generalization ability of artificial neural networks in forecasting TCP/IP traffic trends: How much does the size of learning rate matter?" *Int. J. Comput. Sci. Appl.*, vol. 4, no. 1, pp. 9–17, 2015.

[216] J. Wang *et al.*, "Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach," in *Proc. 36th Annu. IEEE Int. Conf. Comput. Commun. (INFOCOM)*, Atlanta, GA, USA, 2017, pp. 1–9.

[217] C. Zhang and P. Patras, "Long-term mobile traffic forecasting using deep spatio-temporal neural networks," in *Proc. 18th ACM Int. Symp. Mobile Ad Hoc Netw. Comput.*, Los Angeles, CA, USA, 2018, pp. 231–240.

[218] C. Zhang, X. Ouyang, and P. Patras, "ZipNet-GAN: Inferring fine-grained mobile traffic patterns via a generative adversarial neural network," in *Proc. 13th ACM Conf. Emerg. Netw. Exp. Technol.*, Incheon, South Korea, 2017, pp. 363–375.

[219] C.-W. Huang, C.-T. Chiang, and Q. Li, "A study of deep learning networks on mobile traffic forecasting," in *Proc. 28th IEEE Annu. Int. Symp. Pers. Indoor Mobile Radio Commun. (PIMRC)*, Montreal, QC, Canada, 2017, pp. 1–6.

[220] C. Zhang, H. Zhang, D. Yuan, and M. Zhang, "Citywide cellular traffic prediction based on densely connected convolutional neural networks," *IEEE Commun. Lett.*, vol. 22, no. 8, pp. 1656–1659, Aug. 2018.

[221] S. Navabi, C. Wang, O. Y. Bursalioglu, and H. Papadopoulos, "Predicting wireless channel features using neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[222] Z. Wang, "The applications of deep learning on traffic identification," in *Proc. BlackHat USA*, 2015, pp. 21–26.

[223] W. Wang, M. Zhu, J. Wang, X. Zeng, and Z. Yang, "End-to-end encrypted traffic classification with one-dimensional convolution neural networks," in *Proc. IEEE Int. Conf. Intell. Security Informat.*, Beijing, China, 2017, pp. 43–48.

[224] M. Lotfollahi, R. S. H. Zade, M. J. Siavoshani, and M. Saberian, "Deep packet: A novel approach for encrypted traffic classification using deep learning," *arXiv preprint arXiv:1709.02656*, 2017.

[225] W. Wang, M. Zhu, X. Zeng, X. Ye, and Y. Sheng, "Malware traffic classification using convolutional neural network for representation learning," in *Proc. IEEE Int. Conf. Inf. Netw. (ICOIN)*, Da Nang, Vietnam, 2017, pp. 712–717.

[226] 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, 2016, pp. 1374–1377.

[227] B. Felbo, P. Sundsøy, A. S. Pentland, S. Lehmann, and Y.-A. de Montjoye, "Using deep learning to predict demographics from mobile phone metadata," in *Proc. Workshop Track Int. Conf. Learn. Represent. (ICLR)*, 2016.

[228] N. C. Chen, W. Xie, R. E Welsch, K. Larson, and J. Xie, "Comprehensive predictions of tourists' next visit location based on call detail records using machine learning and deep learning methods," in *Proc. IEEE Int. Congr. Big Data (BigData Congr.)*, Honolulu, HI, USA, 2017, pp. 1–6.

[229] M. Yin, S. Feygin, M. Sheehan, J.-F. Paiement, and A. Pozdnoukhov, "Deep generative models of urban mobility," *IEEE Trans. Intell. Transp. Syst.*, to be published.

[230] C. Xu, K. Chang, K.-C. Chua, M. Hu, and Z. Gao, "Large-scale Wi-Fi hotspot classification via deep learning," in *Proc. 26th Int. Conf. World Wide Web Companion*, 2017, pp. 857–858.

[231] Q. Meng, K. Wang, B. Liu, T. Miyazaki, and X. He, "QoE-based big data analysis with deep learning in pervasive edge environment," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[232] L. Fang, X. Cheng, H. Wang, and L. Yang, "Mobile demand forecasting via deep graph-sequence spatiotemporal modeling in cellular networks," *IEEE Internet Things J.*, vol. 5, no. 4, pp. 3091–3101, Aug. 2018.

[233] C. Luo, J. Ji, Q. Wang, X. Chen, and P. Li, "Channel state information prediction for 5G wireless communications: A deep learning approach," *IEEE Trans. Netw. Sci. Eng.*, to be published.

[234] P. Li *et al.*, "An improved stacked auto-encoder for network traffic flow classification," *IEEE Netw.*, vol. 32, no. 6, pp. 22–27, Nov./Dec. 2018.

[235] J. Feng, X. Chen, R. Gao, M. Zeng, and Y. Li, "DeepTP: An end-to-end neural network for mobile cellular traffic prediction," *IEEE Netw.*, vol. 32, no. 6, pp. 108–115, Nov./Dec. 2018.

[236] H. Zhu, Y. Cao, W. Wang, T. Jiang, and S. Jin, "Deep reinforcement learning for mobile edge caching: Review, new features, and open issues," *IEEE Netw.*, vol. 32, no. 6, pp. 50–57, Nov./Dec. 2018.

[237] S. Liu and J. Du, "Poster: Mobiear-building an environment-independent acoustic sensing platform for the deaf using deep learning," in *Proc. 14th ACM Annu. Int. Conf. Mobile Syst. Appl. Services Companion*, Singapore, 2016, p. 50.

[238] L. Sicong *et al.*, "UbiEar: Bringing location-independent sound awareness to the hard-of-hearing people with smartphones," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 2, 2017, Art. no. 17.

[239] V. Jindal, "Integrating mobile and cloud for PPG signal selection to monitor heart rate during intensive physical exercise," in *Proc. ACM Int. Workshop Mobile Softw. Eng. Syst.*, Austin, TX, USA, 2016, pp. 36–37.

[240] E. Kim, M. Corte-Real, and Z. Baloch, "A deep semantic mobile application for thyroid cytopathology," in *Proc. Med. Imag. PACS Imag. Informat. Next Gener. Innov.*, vol. 9789. San Diego, CA, USA, 2016, Art. no. 97890A.

[241] A. Sathyanarayana *et al.*, "Sleep quality prediction from wearable data using deep learning," *JMIR mHealth uHealth*, vol. 4, no. 4, 2016, Art. no. e125.

[242] H. Li and M. Trocan, "Personal health indicators by deep learning of smart phone sensor data," in *Proc. 3rd IEEE Int. Conf. Cybern. (CYBCONF)*, Exeter, U.K., 2017, pp. 1–5.

[243] M.-P. Hosseini, T. X. Tran, D. Pompili, K. Elisevich, and H. Soltanian-Zadeh, "Deep learning with edge computing for localization of epileptogenicity using multimodal rs-fMRI and EEG big data," in *Proc. IEEE Int. Conf. Auton. Comput. (ICAC)*, Columbus, OH, USA, 2017, pp. 83–92.

[244] C. Stamate *et al.*, "Deep learning Parkinson's from smartphone data," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, Kona, HI, USA, 2017, pp. 31–40.

[245] T. Quisel, L. Foschini, A. Signorini, and D. C. Kale, "Collecting and analyzing millions of mHealth data streams," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 1971–1980.

[246] U. M. Khan, Z. Kabir, S. A. Hassan, and S. H. Ahmed, "A deep learning framework using passive WiFi sensing for respiration monitoring," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, Singapore, 2017, pp. 1–6.

[247] 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 Comput. (SEC)*, Washington, DC, USA, 2016, pp. 64–76.

[248] L. Tobías, A. Ducournau, F. Rousseau, G. Mercier, and R. Fablet, "Convolutional neural networks for object recognition on mobile devices: A case study," in *Proc. 23rd IEEE Int. Conf. Pattern Recognit. (ICPR)*, Cancún, Mexico, 2016, pp. 3530–3535.

[249] P. Pouladzadeh and S. Shirmohammadi, "Mobile multi-food recognition using deep learning," *ACM Trans. Mobile Comput. Commun. Appl.*, vol. 13, no. 3s, Aug. 2017, Art. no. 36.

[250] R. Tanno, K. Okamoto, and K. Yanai, "DeepFoodCam: A DCNN-based real-time mobile food recognition system," in *Proc. 2nd ACM Int. Workshop Multimedia Assisted Dietary Manag.*, Amsterdam, The Netherlands, 2016, p. 89.

[251] P. Kuhad, A. Yassine, and S. Shimohammadi, "Using distance estimation and deep learning to simplify calibration in food calorie measurement," in *Proc. IEEE Int. Conf. Comput. Intell. Virtual Environ. Meas. Syst. Appl. (CIVEMSA)*, Shenzhen, China, 2015, pp. 1–6.

[252] T. Teng and X. Yang, "Facial expressions recognition based on convolutional neural networks for mobile virtual reality," in *Proc. 15th ACM SIGGRAPH Conf. Virtual Real. Continuum Appl. Ind.*, vol. 1, Zhuhai, China, 2016, pp. 475–478.

[253] J. Rao, Y. Qiao, F. Ren, J. Wang, and Q. Du, "A mobile outdoor augmented reality method combining deep learning object detection and spatial relationships for geovisualization," *Sensors*, vol. 17, no. 9, 2017, Art. no. E1951.

[254] M. Zeng *et al.*, "Convolutional neural networks for human activity recognition using mobile sensors," in *Proc. 6th IEEE Int. Conf. Mobile Comput. Appl. Services (MobiCASE)*, Austin, TX, USA, 2014, pp. 197–205.

[255] B. Almaslukh, J. AlMuhtadi, and A. Artoli, "An effective deep autoencoder approach for online smartphone-based human activity recognition," *Int. J. Comput. Sci. Netw. Security*, vol. 17, no. 4, pp. 160–165, 2017.

[256] X. Li, Y. Zhang, I. Marsic, A. Sarcevic, and R. S. Burd, "Deep learning for RFID-based activity recognition," in *Proc. 4th ACM Conf. Embedded Netw. Sensor Syst. (CD-ROM)*, Stanford, CA, USA, 2016, pp. 164–175.

[257] S. Bhattacharya and N. D. Lane, "From smart to deep: Robust activity recognition on smartwatches using deep learning," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, Sydney, NSW, Australia, 2016, pp. 1–6.

[258] A. Antoniou and P. Angelov, "A general purpose intelligent surveillance system for mobile devices using deep learning," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Vancouver, BC, Canada, 2016, pp. 2879–2886.

[259] S. Wang, J. Song, J. Lien, I. Poupyrev, and O. Hilliges, "Interacting with soli: Exploring fine-grained dynamic gesture recognition in the radio-frequency spectrum," in *Proc. 29th ACM Annu. Symp. User Interface Softw. Technol.*, Tokyo, Japan, 2016, pp. 851–860.

[260] Y. Gao *et al.*, "iHear food: Eating detection using commodity Bluetooth headsets," in *Proc. IEEE 1st Int. Conf. Connected Health Appl. Syst Eng. Technol. (CHASE)*, Washington, DC, USA, 2016, pp. 163–172.

[261] J. Zhu, A. Pande, P. Mohapatra, and J. J. Han, "Using deep learning for energy expenditure estimation with wearable sensors," in *Proc. 17th IEEE Int. Conf. E-health Netw. Appl. Services (HealthCom)*, Boston, MA, USA, 2015, pp. 501–506,.

[262] P. Sundsøy, J. Bjelland, B.-A. Reme, A. M. Iqbal, and E. Jahani, "Deep learning applied to mobile phone data for individual income classification," in *Proc. ICAITA*, 2016, pp. 96–99.

[263] Y. Chen and Y. Xue, "A deep learning approach to human activity recognition based on single accelerometer," in *Proc. IEEE Int. Conf. Syst. Man Cybern. (SMC)*, 2015, pp. 1488–1492.

[264] S. Ha and S. Choi, "Convolutional neural networks for human activity recognition using multiple accelerometer and gyroscope sensors," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Vancouver, BC, Canada, 2016, pp. 381–388.

[265] M. Edel and E. Köppe, "Binarized-BLSTM-RNN based human activity recognition," in *Proc. IEEE Int. Conf. Indoor Position. Indoor Navig. (IPIN)*, 2016, pp. 1–7.

[266] S. Xue *et al.*, "AppDNA: App behavior profiling via graph-based deep learning," in *Proc. IEEE Conf. Comput. Commun.*, Honolulu, HI, USA, 2018, pp. 1475–1483.

[267] H. Liu *et al.*, "Finding the stars in the fireworks: Deep understanding of motion sensor fingerprint," in *Proc. IEEE Conf. Comput. Commun.*, Honolulu, HI, USA, 2018, pp. 126–134.

[268] T. Okita and S. Inoue, "Recognition of multiple overlapping activities using compositional CNN-LSTM model," in *Proc. ACM Int. Joint Conf. Pervasive Ubiquitous Comput. ACM Int. Symp. Wearable Comput.*, 2017, pp. 165–168.

[269] 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, 2016, pp. 940–945.

[270] L. Seidenari *et al.*, "Deep artwork detection and retrieval for automatic context-aware audio guides," *ACM Trans. Mobile Comput. Commun. Appl.*, vol. 13, no. 3s, 2017, Art. no. 35.

[271] X. Zeng, K. Cao, and M. Zhang, "MobileDeepPill: A small-footprint mobile deep learning system for recognizing unconstrained pill images," in *Proc. 15th ACM Annu. Int. Conf. Mobile Syst. Appl. Services*, 2017, pp. 56–67.

[272] H. Zou *et al.*, "DeepSense: Device-free human activity recognition via autoencoder long-term recurrent convolutional network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[273] X. Zeng, "Mobile sensing through deep learning," in *Proc. Workshop MobiSys Ph.D. Forum*, 2017, pp. 5–6.

[274] X. Wang, L. Gao, and S. Mao, "PhaseFi: Phase fingerprinting for indoor localization with a deep learning approach," in *Proc. IEEE Glob. Commun. Conf. (GLOBECOM)*, San Diego, CA, USA, 2015, pp. 1–6.

[275] X. Wang, L. Gao, and S. Mao, "CSI phase fingerprinting for indoor localization with a deep learning approach," *IEEE Internet Things J.*, vol. 3, no. 6, pp. 1113–1123, Dec. 2016.

[276] C. Feng, S. Arshad, R. Yu, and Y. Liu, "Evaluation and improvement of activity detection systems with recurrent neural network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[277] B. Cao *et al.*, "DeepMood: Modeling mobile phone typing dynamics for mood detection," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 747–755.

[278] X. Ran, H. Chen, X. Zhu, Z. Liu, and J. Chen, "DeepDecision: A mobile deep learning framework for edge video analytics," in *Proc. IEEE Int. Conf. Comput. Commun.*, Honolulu, HI, USA, 2018, pp. 1421–1429.

[279] Siri Team. (2017). *Deep Learning for Siri's Voice: On-Device Deep Mixture Density Networks for Hybrid Unit Selection Synthesis*. Accessed: Sep. 16, 2017. [Online]. Available: https://machinelearning.apple.com/2017/08/06/siri-voices.html

[280] I. McGraw *et al.*, "Personalized speech recognition on mobile devices," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Shanghai, China, 2016, pp. 5955–5959.

[281] R. Prabhavalkar, O. Alsharif, A. Bruguier, and L. McGraw, "On the compression of recurrent neural networks with an application to LVCSR acoustic modeling for embedded speech recognition," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Shanghai, China, 2016, pp. 5970–5974.

[282] T. Yoshioka *et al.*, "The NTT CHiME-3 system: Advances in speech enhancement and recognition for mobile multi-microphone devices," in *Proc. IEEE Workshop Autom. Speech Recognit. Understand. (ASRU)*, Scottsdale, AZ, USA, 2015, pp. 436–443.

[283] S. Ruan, J. O. Wobbrock, K. Liou, A. Ng, and J. Landay, "Speech is 3x faster than typing for English and mandarin text entry on mobile devices," *arXiv preprint arXiv:1608.07323*, 2016.

[284] A. Ignatov, N. Kobyshev, R. Timofte, K. Vanhoey, and L. V. Gool, "DSLR-quality photos on mobile devices with deep convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, 2017, pp. 3297–3305.

[285] Z. Lu, N. Felemban, K. Chan, and T. La Porta, "Demo abstract: On-demand information retrieval from videos using deep learning in wireless networks," in *Proc. IEEE/ACM 2nd Int. Conf. Internet Things Design Implement. (IoTDI)*, 2017, pp. 279–280.

[286] J. Lee, J. Kwon, and H. Kim, "Reducing distraction of smartwatch users with deep learning," in *Proc. 18th ACM Int. Conf. Human–Comput. Interact. Mobile Devices Services Adjunct*, 2016, pp. 948–953.

[287] T. H. Vu, L. Dung, and J.-C. Wang, "Transportation mode detection on mobile devices using recurrent nets," in *Proc. ACM Multimedia Conf.*, 2016, pp. 392–396.

[288] S.-H. Fang, Y.-X. Fei, Z. Xu, and Y. Tsao, "Learning transportation modes from smartphone sensors based on deep neural network," *IEEE Sensors J.*, vol. 17, no. 18, pp. 6111–6118, Sep. 2017.

[289] M. Zhao *et al.*, "RF-based 3D skeletons," in *Proc. ACM Conf. ACM Special Interest Group Data Commun. (SIGCOMM)*, 2018, pp. 267–281.

[290] K. Katevas, I. Leontiadis, M. Pielot, and J. Serrà, "Practical processing of mobile sensor data for continual deep learning predictions," in *Proc. 1st ACM Int. Workshop Deep Learn. Mobile Syst. Appl.*, 2017, pp. 19–24.

[291] S. Yao, S. Hu, Y. Zhao, A. Zhang, and T. Abdelzaher, "DeepSense: A unified deep learning framework for time-series mobile sensing data processing," in *Proc. 26th Int. Conf. World Wide Web*, 2017, pp. 351–360.

[292] K. Ohara, T. Maekawa, and Y. Matsushita, "Detecting state changes of indoor everyday objects using Wi-Fi channel state information," *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol.*, vol. 1, no. 3, p. 88, 2017.

[293] W. Liu, H. Ma, H. Qi, D. Zhao, and Z. Chen, "Deep learning hashing for mobile visual search," *EURASIP J. Image Video Process.*, vol. 2017, p. 17, Dec. 2017.

[294] X. Ouyang, C. Zhang, P. Zhou, and H. Jiang, "DeepSpace: An online deep learning framework for mobile big data to understand human mobility patterns," *arXiv preprint arXiv:1610.07009*, 2016.

[295] H. Yang, Z. Li, and Z. Liu, "Neural networks for MANET AODV: An optimization approach," *Clust. Comput.*, vol. 20, no. 4, pp. 3369–3377, 2017.

[296] X. Song, H. Kanasugi, and R. Shibasaki, "DeepTransport: Prediction and simulation of human mobility and transportation mode at a city-wide level," in *Proc. Int. Joint Conf. Artif. Intell.*, New York, NY, USA, 2016, pp. 2618–2624.

[297] J. Zhang, Y. Zheng, and D. Qi, "Deep spatio-temporal residual networks for citywide crowd flows prediction," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2017, pp. 1655–1661.

[298] J. V. Subramanian and M. A. K. Sadiq, "Implementation of artificial neural network for mobile movement prediction," *Indian J. Sci. Technol.*, vol. 7, no. 6, pp. 858–863, 2014.

[299] L. S. Ezema and C. I. Ani, "Artificial neural network approach to mobile location estimation in GSM network," *Int. J. Electron. Telecommun.*, vol. 63, no. 1, pp. 39–44, 2017.

[300] W. Shao *et al.*, "DePedo: Anti periodic negative-step movement pedometer with deep convolutional neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[301] Y. Yayeh *et al.*, "Mobility prediction in mobile ad-hoc network using deep learning," in *Proc. IEEE Int. Conf. Appl. Syst. Invention (ICASI)*, 2018, pp. 1203–1206.

[302] Q. Chen, X. Song, H. Yamada, and R. Shibasaki, "Learning deep representation from big and heterogeneous data for traffic accident inference," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2016, pp. 338–344.

[303] X. Song *et al.*, "DeepMob: Learning deep knowledge of human emergency behavior and mobility from big and heterogeneous data," *ACM Trans. Inf. Syst.*, vol. 35, no. 4, p. 41, 2017.

[304] D. Yao, C. Zhang, Z. Zhu, J. Huang, and J. Bi, "Trajectory clustering via deep representation learning," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, 2017, pp. 3880–3887.

[305] Z. Liu, Z. Li, K. Wu, and M. Li, "Urban traffic prediction from mobility data using deep learning," *IEEE Netw.*, vol. 32, no. 4, pp. 40–46, Jul./Aug. 2018.

[306] D. S. Wickramasuriya, C. A. Perumalla, K. Davaslioglu, and R. D. Gitlin, "Base station prediction and proactive mobility management in virtual cells using recurrent neural networks," in *Proc. IEEE Wireless Microw. Technol. Conf. (WAMICON)*, 2017, pp. 1–6.

[307] J. Tkačík and P. Kordík, "Neural Turing machine for sequential learning of human mobility patterns," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Vancouver, BC, Canada, 2016, pp. 2790–2797.

[308] D. Y. Kim and H. Y. Song, "Method of predicting human mobility patterns using deep learning," *Neurocomputing*, vol. 280, pp. 56–64, Mar. 2018.

[309] R. Jiang *et al.*, "DeepUrbanMomentum: An online deep-learning system for short-term urban mobility prediction," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2018, pp. 784–791.

[310] C. Wang, Z. Zhao, Q. Sun, and H. Zhang, "Deep learning-based intelligent dual connectivity for mobility management in dense network," in *Proc. IEEE 88th Veh. Technol. Conf.*, 2018, p. 5.

[311] R. Jiang *et al.*, "Deep ROI-based modeling for urban human mobility prediction," in *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol. (IMWUT)*, vol. 2, no. 1, 2018, p. 14.

[312] J. Feng *et al.*, "DeepMove: Predicting human mobility with attentional recurrent networks," in *Proc. World Wide Web Conf. World Wide Web*, 2018, pp. 1459–1468.

[313] 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, 2015, pp. 1666–1671.

[314] X. Wang, X. Wang, and S. Mao, "CiFi: Deep convolutional neural networks for indoor localization with 5 GHz Wi-Fi," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, 2017, pp. 1–6.

[315] X. Wang, L. Gao, and S. Mao, "BiLoc: Bi-modal deep learning for indoor localization with commodity 5GHz WiFi," *IEEE Access*, vol. 5, pp. 4209–4220, 2017.

[316] M. Nowicki and J. Wietrzykowski, "Low-effort place recognition with WiFi fingerprints using deep learning," in *Proc. Int. Conf. Autom.*, 2017, pp. 575–584.

[317] X. Zhang, J. Wang, Q. Gao, X. Ma, and H. Wang, "Device-free wireless localization and activity recognition with deep learning," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. Workshops (PerCom Workshops)*, 2016, pp. 1–5.

[318] J. Wang, X. Zhang, Q. Gao, H. Yue, and H. Wang, "Device-free wireless localization and activity recognition: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 7, pp. 6258–6267, Jul. 2017.

[319] 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.

[320] N. Anzum, S. F. Afroze, and A. Rahman, "Zone-based indoor localization using neural networks: A view from a real testbed," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–7.

[321] X. Wang, Z. Yu, and S. Mao, "DeepML: Deep LSTM for indoor localization with smartphone magnetic and light sensors," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[322] A. K. T. R. Kumar, B. Schäufele, D. Becker, O. Sawade, and I. Radusch, "Indoor localization of vehicles using deep learning," in *Proc. 17th IEEE Int. Symp. World Wireless Mobile Multimedia Netw. (WoWMoM)*, 2016, pp. 1–6.

[323] Z. Zhengj and J. Weng, "Mobile device based outdoor navigation with on-line learning neural network: A comparison with convolutional neural network," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2016, pp. 11–18.

[324] J. Vieira, E. Leitinger, M. Sarajlic, X. Li, and F. Tufvesson, "Deep convolutional neural networks for massive MIMO fingerprint-based positioning," in *Proc. 28th IEEE Annu. Int. Symp. Pers. Indoor Mobile Radio Commun.*, Montreal, QC, Canada, 2017, pp. 1–6.

[325] X. Wang, L. Gao, S. Mao, and S. Pandey, "CSI-based fingerprinting for indoor localization: A deep learning approach," *IEEE Trans. Veh. Technol.*, vol. 66, no. 1, pp. 763–776, Jan. 2017.

[326] H. Chen, Y. Zhang, W. Li, X. Tao, and P. Zhang, "ConFi: Convolutional neural networks based indoor Wi-Fi localization using channel state information," *IEEE Access*, vol. 5, pp. 18066–18074, 2017.

[327] A. Shokry, M. Torki, and M. Youssef, "DeepLoc: A ubiquitous accurate and low-overhead outdoor cellular localization system," in *Proc. 26th ACM SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst.*, 2018, pp. 339–348.

[328] R. Zhou, M. Hao, X. Lu, M. Tang, and Y. Fu, "Device-free localization based on CSI fingerprints and deep neural networks," in *Proc. 15th Annu. IEEE Int. Conf. Sens. Commun. Netw. (SECON)*, 2018, pp. 1–9.

[329] W. Zhang, R. Sengupta, J. Fodero, and X. Li, "DeepPositioning: Intelligent fusion of pervasive magnetic field and WiFi fingerprinting for smartphone indoor localization via deep learning," in *Proc. 16th IEEE Int. Conf. Mach. Learn. Appl. (ICMLA)*, 2017, pp. 7–13.

[330] A. Adege, H.-P. Lin, G. Tarekegn, and S.-S. Jeng, "Applying deep neural network (DNN) for robust indoor localization in multi-building environment," *Appl. Sci.*, vol. 8, no. 7, p. 1062, 2018.

[331] M. Ibrahim, M. Torki, and M. ElNainay, "CNN based indoor localization using RSS time-series," in *Proc. IEEE Symp. Comput. Commun. (ISCC)*, 2018, pp. 1044–1049.

[332] A. Niitsoo, T. Edelhäußer, and C. Mutschler, "Convolutional neural networks for position estimation in TDoA-based locating systems," in *Proc. IEEE Int. Conf. Indoor Position. Indoor Navig. (IPIN)*, 2018, pp. 1–8.

[333] X. Wang, X. Wang, and S. Mao, "Deep convolutional neural networks for indoor localization with CSI images," *IEEE Trans. Netw. Sci. Eng.*, to be published.

[334] C. Xiao, D. Yang, Z. Chen, and G. Tan, "3-D BLE indoor localization based on denoising autoencoder," *IEEE Access*, vol. 5, pp. 12751–12760, 2017.

[335] C.-Y. Hsu *et al.*, "Zero-effort in-home sleep and insomnia monitoring using radio signals," in *Proc. ACM Interact. Mobile Wearable Ubiquitous Technol. (IMWUT)*, vol. 1, no. 3, Sep. 2017, Art. no. 59.

[336] W. Guan *et al.*, "High-precision approach to localization scheme of visible light communication based on artificial neural networks and modified genetic algorithms," *Opt. Eng.*, vol. 56, no. 10, 2017, Art. no. 106103.

[337] P.-J. Chuang and Y.-J. Jiang, "Effective neural network-based node localisation scheme for wireless sensor networks," *IET Wireless Sensor Syst.*, vol. 4, no. 2, pp. 97–103, Jun. 2014.

[338] M. Bernas and B. Płaczek, "Fully connected neural networks ensemble with signal strength clustering for indoor localization in wireless sensor networks," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 12, 2015, Art. no. 403242.

[339] A. Payal, C. S. Rai, and B. V. R. Reddy, "Analysis of some feedforward artificial neural network training algorithms for developing localization framework in wireless sensor networks," *Wireless Pers. Commun.*, vol. 82, no. 4, pp. 2519–2536, 2015.

[340] Y. Dong, Z. Li, R. Wang, and K. Zhang, "Range-based localization in underwater wireless sensor networks using deep neural network," in *Proc. IPSN*, 2017, pp. 321–322.

[341] X. Yan *et al.*, "Real-time identification of smoldering and flaming combustion phases in forest using a wireless sensor network-based multi-sensor system and artificial neural network," *Sensors*, vol. 16, no. 8, p. 1228, 2016.

[342] B. Wang, X. Gu, L. Ma, and S. Yan, "Temperature error correction based on BP neural network in meteorological wireless sensor network," *Int. J. Sensor Netw.*, vol. 23, no. 4, pp. 265–278, 2017.

[343] K.-S. Lee, S.-R. Lee, Y. Kim, and C.-G. Lee, "Deep learning–based real-time query processing for wireless sensor network," *Int. J. Distrib. Sensor Netw.*, vol. 13, no. 5, pp. 1–10, 2017.

[344] J. Li and G. Serpen, "Adaptive and intelligent wireless sensor networks through neural networks: An illustration for infrastructure adaptation through Hopfield network," *Appl. Intell.*, vol. 45, no. 2, pp. 343–362, 2016.

[345] F. Khorasani and H. R. Naji, "Energy efficient data aggregation in wireless sensor networks using neural networks," *Int. J. Sensor Netw.*, vol. 24, no. 1, pp. 26–42, 2017.

[346] C. Li, X. Xie, Y. Huang, H. Wang, and C. Niu, "Distributed data mining based on deep neural network for wireless sensor network," *Int. J. Distrib. Sensor Netw.*, vol. 11, no. 7, 2015, Art. no. 157453.

[347] T. Luo and S. G. Nagarajany, "Distributed anomaly detection using autoencoder neural networks in WSN for IoT," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[348] D. P. Kumar, T. Amgoth, and C. S. R. Annavarapu, "Machine learning algorithms for wireless sensor networks: A survey," *Inf. Fusion*, vol. 49, pp. 1–25, Sep. 2019.

[349] N. Heydari and B. Minaei-Bidgoli, "Reduce energy consumption and send secure data wireless multimedia sensor networks using a combination of techniques for multi-layer watermark and deep learning," *Int. J. Comput. Sci. Netw. Security*, vol. 17, no. 2, pp. 98–105, 2017.

[350] S. Phoemphon, C. So-In, and D. T. Niyato, "A hybrid model using fuzzy logic and an extreme learning machine with vector particle swarm optimization for wireless sensor network localization," *Appl. Soft Comput.*, vol. 65, pp. 101–120, Apr. 2018.

[351] S. S. Banihashemian, F. Adibnia, and M. A. Sarram, "A new range-free and storage-efficient localization algorithm using neural networks in wireless sensor networks," *Wireless Pers. Commun.*, vol. 98, no. 1, pp. 1547–1568, 2018.

[352] W. Sun *et al.*, "WNN-LQE: Wavelet-neural-network-based link quality estimation for smart grid WSNs," *IEEE Access*, vol. 5, pp. 12788–12797, 2017.

[353] J. Kang, Y.-J. Park, J. Lee, S.-H. Wang, and D.-S. Eom, "Novel leakage detection by ensemble CNN-SVM and graph-based localization in water distribution systems," *IEEE Trans. Ind. Electron.*, vol. 65, no. 5, pp. 4279–4289, May 2018.

[354] A. Mehmood, Z. Lv, J. Lloret, and M. M. Umar, "ELDC: An artificial neural network based energy-efficient and robust routing scheme for pollution monitoring in WSNs," *IEEE Trans. Emerg. Topics Comput.*, to be published.

[355] M. A. Alsheikh, S. Lin, D. Niyato, and H.-P. Tan, "Rate-distortion balanced data compression for wireless sensor networks," *IEEE Sensors J.*, vol. 16, no. 12, pp. 5072–5083, Jun. 2016.

[356] A. E. Assaf, S. Zaidi, S. Affes, and N. Kandil, "Robust ANNs-based WSN localization in the presence of anisotropic signal attenuation," *IEEE Wireless Commun. Lett.*, vol. 5, no. 5, pp. 504–507, Oct. 2016.

[357] Y. Wang *et al.*, "A deep learning approach for blind drift calibration of sensor networks," *IEEE Sensors J.*, vol. 17, no. 13, pp. 4158–4171, Jul. 2017.

[358] Z. Jia *et al.*, "Continuous low-power ammonia monitoring using long short-term memory neural networks," in *Proc. 16th ACM Conf. Embedded Netw. Sensor Syst.*, 2018, pp. 224–236.

[359] L. Liu, Y. Cheng, L. Cai, S. Zhou, and Z. Niu, "Deep learning based optimization in wireless network," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, 2017, pp. 1–6.

[360] S. Subramanian and A. Banerjee, "Poster: Deep learning enabled M2M gateway for network optimization," in *Proc. 14th ACM Annu. Int. Conf. Mobile Syst. Appl. Services Companion*, 2016, p. 144.

[361] Y. He, C. Liang, F. R. Yu, N. Zhao, and H. Yin, "Optimization of cache-enabled opportunistic interference alignment wireless networks: A big data deep reinforcement learning approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, 2017, pp. 1–6.

[362] Y. He *et al.*, "Deep reinforcement learning-based optimization for cache-enabled opportunistic interference alignment wireless networks," *IEEE Trans. Veh. Technol.*, vol. 66, no. 11, pp. 10433–10445, Nov. 2017.

[363] F. B. Mismar and B. L. Evans, "Deep reinforcement learning for improving downlink mmWave communication performance," *arXiv preprint arXiv:1707.02329*, 2017.

[364] Z. Wang, L. Li, Y. Xu, H. Tian, and S. Cui, "Handover optimization via asynchronous multi-user deep reinforcement learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[365] Z. Chen and D. B. Smith, "Heterogeneous machine-type communications in cellular networks: Random access optimization by deep reinforcement learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[366] L. Chen, J. Lingys, K. Chen, and F. Liu, "AuTO: Scaling deep reinforcement learning for datacenter-scale automatic traffic optimization," in *Proc. ACM Conf. Special Interest Group Data Commun. (SIGCOMM)*, 2018, pp. 191–205.

[367] Y. Lee, "Classification of node degree based on deep learning and routing method applied for virtual route assignment," *Ad Hoc Netw.*, vol. 58, pp. 70–85, Apr. 2017.

[368] F. Tang *et al.*, "On removing routing protocol from future wireless networks: A real-time deep learning approach for intelligent traffic control," *IEEE Wireless Commun.*, vol. 25, no. 1, pp. 154–160, Feb. 2018.

[369] Q. Zhang, M. Lin, L. T. Yang, Z. Chen, and P. Li, "Energy-efficient scheduling for real-time systems based on deep Q-learning model," *IEEE Trans. Sustain. Comput.*, vol. 4, no. 1, pp. 132–141, Jan./Mar. 2019.

[370] R. Atallah, C. Assi, and M. Khabbaz, "Deep reinforcement learning-based scheduling for roadside communication networks," in *Proc. 15th IEEE Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw. (WiOpt)*, Paris, France, 2017, pp. 1–8.

[371] S. Chinchali *et al.*, "Cellular network traffic scheduling with deep reinforcement learning," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, 2018.

[372] Y. Wei, Z. Zhang, F. R. Yu, and Z. Han, "Joint user scheduling and content caching strategy for mobile edge networks using deep reinforcement learning," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2018, pp. 1–6.

[373] H. Sun *et al.*, "Learning to optimize: Training deep neural networks for wireless resource management," in *Proc. 18th IEEE Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, 2017, pp. 1–6.

[374] Z. Xu, Y. Wang, J. Tang, J. Wang, and M. C. Gursoy, "A deep reinforcement learning based framework for power-efficient resource allocation in cloud RANs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2017, pp. 1–6.

[375] P. V. R. Ferreira *et al.*, "Multi-objective reinforcement learning-based deep neural networks for cognitive space communications," in *Proc. Cogn. Commun. Aerosp. Appl. Workshop (CCAA)*, 2017, pp. 1–8.

[376] H. Ye and G. Y. Li, "Deep reinforcement learning for resource allocation in V2V communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, 2018, pp. 1–6.

[377] U. Challita, L. Dong, and W. Saad, "Proactive resource management for LTE in unlicensed spectrum: A deep learning perspective," *IEEE Trans. Wireless Commun.*, vol. 17, no. 7, pp. 4674–4689, Jul. 2018.

[378] O. Naparstek and K. Cohen, "Deep multi-user reinforcement learning for dynamic spectrum access in multichannel wireless networks," in *Proc. IEEE Glob. Commun. Conf.*, 2017, pp. 1–7.

[379] T. J. O'Shea and T. C. Clancy, "Deep reinforcement learning radio control and signal detection with KeRLym, a Gym RL agent," *arXiv preprint arXiv:1605.09221*, 2016.

[380] M. A. Wijaya, K. Fukawa, and H. Suzuki, "Intercell-interference cancellation and neural network transmit power optimization for MIMO channels," in *Proc. IEEE 82nd Veh. Technol. Conf. (VTC Fall)*, Boston, MA, USA, 2015, pp. 1–5.

[381] H. Rutagemwa, A. Ghasemi, and S. Liu, "Dynamic spectrum assignment for land mobile radio with deep recurrent neural networks," in *Proc. IEEE Int. Conf. Commun. Workshops (ICC Workshops)*, 2018, pp. 1–6.

[382] M. A. Wijaya, K. Fukawa, and H. Suzuki, "Neural network based transmit power control and interference cancellation for MIMO small cell networks," *IEICE Trans. Commun.*, vol. E99-B, no. 5, pp. 1157–1169, 2016.

[383] H. Mao, R. Netravali, and M. Alizadeh, "Neural adaptive video streaming with pensieve," in *Proc. Conf. ACM Special Interest Group Data Commun. (SIGCOMM)*, 2017, pp. 197–210.

[384] T. Oda, R. Obukata, M. Ikeda, L. Barolli, and M. Takizawa, "Design and implementation of a simulation system based on deep Q-network for mobile actor node control in wireless sensor and actor networks," in *Proc. 31st IEEE Int. Conf. Adv. Inf. Netw. Appl. Workshops (WAINA)*, 2017, pp. 195–200.

[385] T. Oda *et al.*, "Performance evaluation of a deep Q-network based simulation system for actor node mobility control in wireless sensor and actor networks considering three-dimensional environment," in *Proc. Int. Conf. Intell. Netw. Collaborative Syst.*, 2017, pp. 41–52.

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

[387] U. Challita, W. Saad, and C. Bettstetter, "Deep reinforcement learning for interference-aware path planning of cellular-connected UAVs," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–7.

[388] C. Luo, J. Ji, Q. Wang, L. Yu, and P. Li, "Online power control for 5G wireless communications: A deep Q-network approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[389] Y. Yu, T. Wang, and S. C. Liew, "Deep-reinforcement learning multiple access for heterogeneous wireless networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–7.

[390] Z. Xu *et al.*, "Experience-driven networking: A deep reinforcement learning based approach," in *Proc. IEEE Int. Conf. Comput. Commun.*, Honolulu, HI, USA, 2018, pp. 1871–1879.

[391] J. Liu, B. Krishnamachari, S. Zhou, and Z. Niu, "DeepNap: Data-driven base station sleeping operations through deep reinforcement learning," *IEEE Internet Things J.*, vol. 5, no. 6, pp. 4273–4282, Dec. 2018.

[392] Z. Zhao *et al.*, "Deep reinforcement learning for network slicing," *arXiv preprint arXiv:1805.06591*, 2018.

[393] J. Li, H. Gao, T. Lv, and Y. Lu, "Deep reinforcement learning based computation offloading and resource allocation for MEC," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, 2018, pp. 1–6.

[394] R. Mennes, M. Camelo, M. Claeys, and S. Latré, "A neural-network-based MF-TDMA MAC scheduler for collaborative wireless networks," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, 2018, pp. 1–6.

[395] Y. Zhou, Z. M. Fadlullah, B. Mao, and N. Kato, "A deep-learning-based radio resource assignment technique for 5G ultra dense networks," *IEEE Netw.*, vol. 32, no. 6, pp. 28–34, Nov./Dec. 2018.

[396] B. Mao *et al.*, "A tensor based deep learning technique for intelligent packet routing," in *Proc. IEEE Glob. Commun. Conf.*, Singapore, 2017, pp. 1–6.

[397] F. Geyer and G. Carle, "Learning and generating distributed routing protocols using graph-based deep learning," in *Proc. ACM Workshop Big Data Anal. Mach. Learn. Data Commun. Netw.*, 2018, pp. 40–45.

[398] N. C. Luong *et al.*, "Joint transaction transmission and channel selection in cognitive radio based blockchain networks: A deep reinforcement learning approach," *arXiv preprint arXiv:1810.10139*, 2018.

[399] X. Li *et al.*, "Intelligent power control for spectrum sharing in cognitive radios: A deep reinforcement learning approach," *IEEE Access*, vol. 6, pp. 25463–25473, 2018.

[400] W. Lee, M. Kim, and D.-H. Cho, "Deep learning based transmit power control in underlaid device-to-device communication," *IEEE Syst. J.*, to be published.

[401] C. H. Liu, Z. Chen, J. Tang, J. Xu, and C. Piao, "Energy-efficient UAV control for effective and fair communication coverage: A deep reinforcement learning approach," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 9, pp. 2059–2070, Sep. 2018.

[402] Y. He, F. R. Yu, N. Zhao, V. C. M. Leung, and H. Yin, "Software-defined networks with mobile edge computing and caching for smart cities: A big data deep reinforcement learning approach," *IEEE Commun. Mag.*, vol. 55, no. 12, pp. 31–37, Dec. 2017.

[403] X. Liu, Y. Xu, L. Jia, Q. Wu, and A. Anpalagan, "Anti-jamming communications using spectrum waterfall: A deep reinforcement learning approach," *IEEE Commun. Lett.*, vol. 22, no. 5, pp. 998–1001, May 2018.

[404] Q. T. A. Pham, Y. Hadjadj-Aoul, and A. Outtagarts, "Deep reinforcement learning based QoS-aware routing in knowledge-defined networking," in *Proc. Qshine EAI Int. Conf. Heterogeneous Netw. Qual. Rel. Security Robustness*, 2018, pp. 1–13.

[405] P. Ferreira *et al.*, "Multi-objective reinforcement learning for cognitive radio–based satellite communications," in *Proc. 34th AIAA Int. Commun. Satellite Syst. Conf.*, 2016, p. 5726.

[406] M. Yousefi-Azar, V. Varadharajan, L. Hamey, and U. Tupakula, "Autoencoder-based feature learning for cyber security applications," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, 2017, pp. 3854–3861.

[407] M. E. Aminanto and K. Kim, "Detecting impersonation attack in WiFi networks using deep learning approach," in *Proc. Int. Workshop Inf. Security Appl.*, 2016, pp. 136–147.

[408] Q. Feng, Z. Dou, C. Li, and G. Si, "Anomaly detection of spectrum in wireless communication via deep autoencoder," in *Proc. Int. Conf. Comput. Sci. Appl.*, 2016, pp. 259–265.

[409] M. A. Khan, S. Khan, B. Shams, and J. Lloret, "Distributed flood attack detection mechanism using artificial neural network in wireless mesh networks," *Security Commun. Netw.*, vol. 9, no. 15, pp. 2715–2729, 2016.

[410] A. A. Diro and N. Chilamkurti, "Distributed attack detection scheme using deep learning approach for Internet of Things," *Future Gener. Comput. Syst.*, vol. 82, pp. 761–768, May 2018.

[411] A. Saied, R. E. Overill, and T. Radzik, "Detection of known and unknown DDoS attacks using artificial neural networks," *Neurocomputing*, vol. 172, pp. 385–393, Jan. 2016.

[412] 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.

[413] K. Hamedani, L. Liu, R. Atat, J. Wu, and Y. Yi, "Reservoir computing meets smart grids: Attack detection using delayed feedback networks," *IEEE Trans. Ind. Informat.*, vol. 14, no. 2, pp. 734–743, Feb. 2018.

[414] R. Das, A. Gadre, S. Zhang, S. Kumar, and J. M. F. Moura, "A deep learning approach to IoT authentication," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[415] P. Jiang, H. Wu, C. Wang, and C. Xin, "Virtual MAC spoofing detection through deep learning," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

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

[417] Z. Yuan, Y. Lu, and Y. Xue, "Droiddetector: Android malware characterization and detection using deep learning," *Tsinghua Sci. Technol.*, vol. 21, no. 1, pp. 114–123, Feb. 2016.

[418] X. Su, D. Zhang, W. Li, and K. Zhao, "A deep learning approach to Android malware feature learning and detection," in *Proc. IEEE Trustcom/BigDataSE/ISPA*, Tianjin, China, 2016, pp. 244–251.

[419] S. Hou, A. Saas, L. Chen, and Y. Ye, "Deep4MalDroid: A deep learning framework for Android malware detection based on Linux kernel system call graphs," in *Proc. IEEE/WIC/ACM Int. Conf. Web Intell. Workshops (WIW)*, Omaha, NE, USA, 2016, pp. 104–111.

[420] F. Martinelli, F. Marulli, and F. Mercaldo, "Evaluating convolutional neural network for effective mobile malware detection," *Procedia Comput. Sci.*, vol. 112, pp. 2372–2381, 2017.

[421] K. K. Nguyen *et al.*, "Cyberattack detection in mobile cloud computing: A deep learning approach," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Barcelona, Spain, 2018, pp. 1–6.

[422] N. McLaughlin *et al.*, "Deep Android malware detection," in *Proc. 7th ACM Conf. Data Appl. Security Privacy*, Scottsdale, AZ, USA, 2017, pp. 301–308.

[423] Y. Chen, Y. Zhang, and S. Maharjan, "Deep learning for secure mobile edge computing," *arXiv preprint arXiv:1709.08025*, 2017.

[424] M. Oulehla, Z. K. Oplatková, and D. Malanik, "Detection of mobile botnets using neural networks," in *Proc. IEEE Future Technol. Conf. (FTC)*, San Francisco, CA, USA, 2016, pp. 1324–1326.

[425] P. Torres, C. Catania, S. Garcia, and C. G. Garino, "An analysis of recurrent neural networks for Botnet detection behavior," in *Proc. IEEE Biennial Congr. Argentina (ARGENCON)*, 2016, pp. 1–6.

[426] M. Eslahi *et al.*, "Mobile Botnet detection model based on retrospective pattern recognition," *Int. J. Security Appl.*, vol. 10, no. 9, pp. 39–54, 2016.

[427] M. Alauthaman, N. Aslam, L. Zhang, R. Alasem, and M. A. Hossain, "A P2P Botnet detection scheme based on decision tree and adaptive multilayer neural networks," *Neural Comput. Appl.*, vol. 29, no. 11, pp. 991–1004, 2018.

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

[429] L. T. Phong, Y. Aono, T. Hayashi, L. Wang, and S. Moriai, "Privacy-preserving deep learning: Revisited and enhanced," in *Proc. Int. Conf. Appl. Tech. Inf. Security*, 2017, pp. 100–110.

[430] S. A. Ossia *et al.*, "A hybrid deep learning architecture for privacy-preserving mobile analytics," *arXiv preprint arXiv:1703.02952*, 2017.

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

[432] S. A. Osia, A. S. Shamsabadi, A. Taheri, H. R. Rabiee, and H. Haddadi, "Private and scalable personal data analytics using a hybrid edge-cloud deep learning," *IEEE Comput. Mag.*, vol. 51, no. 5, pp. 42–49, May 2018.

[433] S. Servia-Rodriguez, L. Wang, J. R. Zhao, R. Mortier, and H. Haddadi, "Personal model training under privacy constraints," in *Proc. 3rd ACM/IEEE Int. Conf. Internet Things Design Implement.*, Apr 2018.

[434] B. Hitaj, G. Ateniese, and F. Perez-Cruz, "Deep models under the GAN: Information leakage from collaborative deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security*, Dallas, TX, USA, 2017, pp. 603–618.

[435] S. Greydanus, "Learning the enigma with recurrent neural networks," *arXiv preprint arXiv:1708.07576*, 2017.

[436] H. Maghrebi, T. Portigliatti, and E. Prouff, "Breaking cryptographic implementations using deep learning techniques," in *Proc. Int. Conf. Security Privacy Appl. Cryptography Eng.*, 2016, pp. 3–26.

[437] Y. Liu *et al.*, "GENPass: A general deep learning model for password guessing with PCFG rules and adversarial generation," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[438] R. Ning, C. Wang, C. Xin, J. Li, and H. Wu, "DeepMag: Sniffing mobile apps in magnetic field through deep convolutional neural networks," in *Proc. IEEE Int. Conf. Pervasive Comput. Commun. (PerCom)*, 2018, pp. 1–10.

[439] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep learning based MIMO communications," *arXiv preprint arXiv:1707.07980*, 2017.

[440] M. Borgerding, P. Schniter, and S. Rangan, "AMP-inspired deep networks for sparse linear inverse problems," *IEEE Trans. Signal Process.*, vol. 65, no. 16, pp. 4293–4308, Aug. 2017.

[441] T. Fujihashi, T. Koike-Akino, T. Watanabe, and P. V. Orlik, "Nonlinear equalization with deep learning for multi-purpose visual MIMO communications," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[442] S. Rajendran, W. Meert, D. Giustiniano, V. Lenders, and S. Pollin, "Deep learning models for wireless signal classification with distributed low-cost spectrum sensors," *IEEE Trans. Cogn. Commun. Netw.*, vol. 4, no. 3, pp. 433–445, Sep. 2018.

[443] N. E. West and T. O'Shea, "Deep architectures for modulation recognition," in *Proc. IEEE Int. Symp. Dyn. Spectr. Access Netw. (DySPAN)*, Piscataway, NJ, USA, 2017, pp. 1–6.

[444] T. J. O'Shea, L. Pemula, D. Batra, and T. C. Clancy, "Radio transformer networks: Attention models for learning to synchronize in wireless systems," in *Proc. 50th Asilomar Conf. Signals Syst. Comput.*, Pacific Grove, CA, USA, 2016, pp. 662–666.

[445] J. Gante, G. Falcão, and L. Sousa, "Beamformed fingerprint learning for accurate millimeter wave positioning," *arXiv preprint arXiv:1804.04112*, 2018.

[446] A. Alkhateeb *et al.*, "Deep learning coordinated beamforming for highly-mobile millimeter wave systems," *IEEE Access*, vol. 6, pp. 37328–37348, 2018.

[447] D. Neumann, T. Wiese, and W. Utschick, "Deep channel estimation," in *Proc. 21st Int. ITG Workshop Smart Antennas*, Berlin, Germany, 2017, pp. 1–6.

[448] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in *Proc. IEEE 18th Int. Workshop Signal Process. Adv. Wireless Commun. (SPAWC)*, Sapporo, Japan, 2017, pp. 1–5.

[449] X. Yan *et al.*, "Signal detection of MIMO-OFDM system based on auto encoder and extreme learning machine," in *Proc. IEEE Int. Joint Conf. Neural Netw. (IJCNN)*, Anchorage, AK, USA, 2017, pp. 1602–1606.

[450] T. O'Shea and J. Hoydis, "An introduction to deep learning for the physical layer," *IEEE Trans. Cogn. Commun. Netw.*, vol. 3, no. 4, pp. 563–575, Dec. 2017.

[451] J. Jagannath *et al.*, "Artificial neural network based automatic modulation classification over a software defined radio testbed," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[452] T. J. O'Shea, S. Hitefield, and J. Corgan, "End-to-end radio traffic sequence recognition with recurrent neural networks," in *Proc. IEEE Glob. Conf. Signal Inf. Process. (GlobalSIP)*, Washington, DC, USA, 2016, pp. 277–281.

[453] T. J. O'Shea, K. Karra, and T. C. Clancy, "Learning to communicate: Channel auto-encoders, domain specific regularizers, and attention," in *Proc. IEEE Int. Symp. Signal Process. Inf. Technol. (ISSPIT)*, Limassol, Cyprus, 2016, pp. 223–228.

[454] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in OFDM systems," *IEEE Wireless Commun. Lett.*, vol. 7, no. 1, pp. 114–117, Feb. 2018.

[455] F. Liang, C. Shen, and F. Wu, "Exploiting noise correlation for channel decoding with convolutional neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[456] W. Lyu, Z. Zhang, C. Jiao, K. Qin, and H. Zhang, "Performance evaluation of channel decoding with deep neural networks," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[457] S. Dörner, S. Cammerer, J. Hoydis, and S. ten Brink, "Deep learning based communication over the air," *IEEE J. Sel. Topics Signal Process.*, vol. 12, no. 1, pp. 132–143, Feb. 2018.

[458] R.-F. Liao *et al.*, "The Rayleigh fading channel prediction via deep learning," *Wireless Commun. Mobile Comput.*, vol. 2018, Jul. 2018, Art. no. 6497340.

[459] H. Hongji, Y. Jie, S. Yiwei, H. Hao, and G. Guan, "Deep learning for super-resolution channel estimation and DOA estimation based massive MIMO system," *IEEE Trans. Veh. Technol.*, vol. 67, no. 9, pp. 8549–8560, Sep. 2018.

[460] S. Huang and H. Lin, "Fully optical spacecraft communications: Implementing an omnidirectional PV-cell receiver and 8 Mb/s LED visible light downlink with deep learning error correction," *IEEE Aerosp. Electron. Syst. Mag.*, vol. 33, no. 4, pp. 16–22, Apr. 2018.

[461] R. Gonzalez, A. Garcia-Duran, F. Manco, M. Niepert, and P. Vallina, "Network data monetization using Net2Vec," in *Proc. ACM SIGCOMM Posters Demos*, Los Angeles, CA, USA, 2017, pp. 37–39.

[462] N. Kaminski *et al.*, "A neural-network-based realization of in-network computation for the Internet of Things," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Paris, France, 2017, pp. 1–6.

[463] L. Xiao, Y. Li, G. Han, H. Dai, and H. V. Poor, "A secure mobile crowdsensing game with deep reinforcement learning," *IEEE Trans. Inf. Forensics Security*, vol. 13, no. 1, pp. 35–47, Jan. 2018.

[464] N. C. Luong, Z. Xiong, P. Wang, and D. Niyato, "Optimal auction for edge computing resource management in mobile blockchain networks: A deep learning approach," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[465] A. Gulati, G. S. Aujla, R. Chaudhary, N. Kumar, and M. S. Obaidat, "Deep learning-based content centric data dissemination scheme for Internet of Vehicles," in *Proc. IEEE Int. Conf. Commun. (ICC)*, Kansas City, MO, USA, 2018, pp. 1–6.

[466] E. Ahmed *et al.*, "Recent advances and challenges in mobile big data," *IEEE Commun. Mag.*, vol. 56, no. 2, pp. 102–108, Feb. 2018.

[467] D. Z. Yazti and S. Krishnaswamy, "Mobile big data analytics: Research, practice, and opportunities," in *Proc. 15th IEEE Int. Conf. Mobile Data Manag. (MDM)*, vol. 1. Brisbane, QLD, Australia, 2014, pp. 1–2.

[468] D. Naboulsi, M. Fiore, S. Ribot, and R. Stanica, "Large-scale mobile traffic analysis: A survey," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 124–161, 1st Quart., 2016.

[469] J. Ngiam *et al.*, "Multimodal deep learning," in *Proc. 28th Int. Conf. Mach. Learn. (ICML)*, Bellevue, WA, USA, 2011, pp. 689–696.

[470] I. Alawe, A. Ksentini, Y. Hadjadj-Aoul, and P. Bertin, "Improving traffic forecasting for 5G core network scalability: A machine learning approach," *IEEE Netw.*, vol. 32, no. 6, pp. 42–49, Nov./Dec. 2018.

[471] G. Aceto, D. Ciuonzo, A. Montieri, and A. Pescapé, "Mobile encrypted traffic classification using deep learning," in *Proc. 2nd IEEE Netw. Traffic Meas. Anal. Conf.*, Vienna, Austria, 2018, pp. 1–8.

[472] 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.

[473] S. Seneviratne *et al.*, "A survey of wearable devices and challenges," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2573–2620, 4th Quart., 2017.

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

[475] 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. ACM Int. Workshop Internet Things Towards Appl.*, Seoul, South Korea, 2015, pp. 7–12.

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

[477] R. Miotto, F. Wang, S. Wang, X. Jiang, and J. T. Dudley, "Deep learning for healthcare: Review, opportunities and challenges," *Briefings Bioinformat.*, vol. 19, no. 6, pp. 1236–1246, 2017.

[478] C. A. Ronao and S.-B. Cho, "Human activity recognition with smartphone sensors using deep learning neural networks," *Expert Syst. Appl.*, vol. 59, pp. 235–244, Oct. 2016.

[479] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognit. Lett.*, vol. 119, pp. 3–11, Mar. 2019.

[480] X. Ran, H. Chen, Z. Liu, and J. Chen, "Delivering deep learning to mobile devices via offloading," in *Proc. ACM Workshop Virtual Reality Augmented Reality Netw.*, Los Angeles, CA, USA, 2017, pp. 42–47.

[481] V. V. Vyas, K. H. Walse, and R. V. Dharaskar, "A survey on human activity recognition using smartphone," *Int. J. Adv. Res. Comput. Sci. Manag. Stud.*, vol. 5, no. 3, pp. 118–125, 2017.

[482] H. Zen and A. Senior, "Deep mixture density networks for acoustic modeling in statistical parametric speech synthesis," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Florence, Italy, 2014, pp. 3844–3848.

[483] K. Zhao, S. Tarkoma, S. Liu, and H. Vo, "Urban human mobility data mining: An overview," in *Proc. IEEE Int. Conf. Big Data (Big Data)*, Washington, DC, USA, 2016, pp. 1911–1920.

[484] C. Yang, M. Sun, W. X. Zhao, Z. Liu, and E. Y. Chang, "A neural network approach to jointly modeling social networks and mobile trajectories," *ACM Trans. Inf. Syst.*, vol. 35, no. 4, 2017, Art. no. 36.

[485] A. Graves, G. Wayne, and I. Danihelka, "Neural Turing machines," *arXiv preprint arXiv:1410.5401*, 2014.

[486] S. Xia, Y. Liu, G. Yuan, M. Zhu, and Z. Wang, "Indoor fingerprint positioning based on Wi-Fi: An overview," *ISPRS Int. J. Geo Inf.*, vol. 6, no. 5, p. 135, 2017.

[487] P. Davidson and R. Piché, "A survey of selected indoor positioning methods for smartphones," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 2, pp. 1347–1370, 2nd Quart., 2017.

[488] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni, "A survey on wireless indoor localization from the device perspective," *ACM Comput. Surveys*, vol. 49, no. 2, 2016, Art. no. 25.

[489] J. Xiao, K. Wu, Y. Yi, and L. M. Ni, "FIFS: Fine-grained indoor fingerprinting system," in *Proc. 21st Int. Conf. Comput. Commun. Netw. (ICCCN)*, Munich, Germany, 2012, pp. 1–7.

[490] M. Youssef and A. Agrawala, "The Horus WLAN location determination system," in *Proc. 3rd ACM Int. Conf. Mobile Syst. Appl. Services*, Seattle, WA, USA, 2005, pp. 205–218.

[491] M. Brunato and R. Battiti, "Statistical learning theory for location fingerprinting in wireless LANs," *Comput. Netw.*, vol. 47, no. 6, pp. 825–845, 2005.

[492] J. Ho and S. Ermon, "Generative adversarial imitation learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, 2016, pp. 4565–4573.

[493] M. Zorzi, A. Zanella, A. Testolin, M. De Filippo De Grazia, and M. Zorzi, "COBANETS: A new paradigm for cognitive communications systems," in *Proc. IEEE Int. Conf. Comput. Netw. Commun. (ICNC)*, 2016, pp. 1–7.

[494] M. Roopaei, P. Rad, and M. Jamshidi, "Deep learning control for complex and large scale cloud systems," *Intell. Autom. Soft Comput.*, vol. 23, no. 3, pp. 1–3, 2017.

[495] P. V. R. Ferreira *et al.*, "Multiobjective reinforcement learning for cognitive satellite communications using deep neural network ensembles," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 5, pp. 1030–1041, May 2018.

[496] T. Schaul, J. Quan, I. Antonoglou, and D. Silver, "Prioritized experience replay," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2016.

[497] Q. Shi, M. Razaviyayn, Z.-Q. Luo, and C. He, "An iteratively weighted MMSE approach to distributed sum-utility maximization for a MIMO interfering broadcast channel," *IEEE Trans. Signal Process.*, vol. 59, no. 9, pp. 4331–4340, Sep. 2011.

[498] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 2, pp. 1153–1176, 2nd Quart., 2016.

[499] J. Wang *et al.*, "Not just privacy: Improving performance of private deep learning in mobile cloud," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, London, U.K., 2018, pp. 2407–2416.

[500] D. Kwon *et al.*, "A survey of deep learning-based network anomaly detection," *Clust. Comput.*, pp. 1–13, Aug. 2017.

[501] M. Tavallaee, E. Bagheri, W. Lu, and A. A. Ghorbani, "A detailed analysis of the KDD CUP 99 data set," in *Proc. IEEE Symp. Comput. Intell. Security Defense Appl.*, Ottawa, ON, Canada, 2009, pp. 1–6.

[502] K. Tam, A. Feizollah, N. B. Anuar, R. Salleh, and L. Cavallaro, "The evolution of Android malware and Android analysis techniques," *ACM Comput. Surveys*, vol. 49, no. 4, 2017, Art. no. 76.

[503] R. A. Rodríguez-Gómez, G. Maciá-Fernández, and P. García-Teodoro, "Survey and taxonomy of Botnet research through life-cycle," *ACM Comput. Surveys*, vol. 45, no. 4, 2013, Art. no. 45.

[504] M. Liu *et al.*, "A collaborative privacy-preserving deep learning system in distributed mobile environment," in *Proc. IEEE Int. Conf. Comput. Sci. Comput. Intell. (CSCI)*, Las Vegas, NV, USA, 2016, pp. 192–197.

[505] S. Chopra, R. Hadsell, and Y. LeCun, "Learning a similarity metric discriminatively, with application to face verification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, vol. 1. San Diego, CA, USA, 2005, pp. 539–546.

[506] A. S. Shamsabadi, H. Haddadi, and A. Cavallaro, "Distributed one-class learning," in *Proc. IEEE Int. Conf. Image Process. (ICIP)*, Athens, Greece, 2018, pp. 4123–4127.

[507] B. Hitaj, P. Gasti, G. Ateniese, and F. Perez-Cruz, "PassGAN: A deep learning approach for password guessing," *arXiv preprint arXiv:1709.00440*, 2017.

[508] H. Ye, G. Y. Li, B.-H. F. Juang, and K. Sivanesan, "Channel agnostic end-to-end learning based communication systems with conditional GAN," *arXiv preprint arXiv:1807.00447*, 2018.

[509] R. Gonzalez *et al.*, "Net2Vec: Deep learning for the network," in *Proc. ACM Workshop Big Data Anal. Mach. Learn. Data Commun. Netw.*, 2017, pp. 13–18.

[510] D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," *IEEE Trans. Evol. Comput.*, vol. 1, no. 1, pp. 67–82, Apr. 1997.

[511] Y. Cheng, D. Wang, P. Zhou, and T. Zhang, "Model compression and acceleration for deep neural networks: The principles, progress, and challenges," *IEEE Signal Process. Mag.*, vol. 35, no. 1, pp. 126–136, Jan. 2018.

[512] N. D. Lane *et al.*, "Squeezing deep learning into mobile and embedded devices," *IEEE Pervasive Comput.*, vol. 16, no. 3, pp. 82–88, 2017.

[513] J. Tang, D. Sun, S. Liu, and J.-L. Gaudiot, "Enabling deep learning on IoT devices," *Computer*, vol. 50, no. 10, pp. 92–96, 2017.

[514] J. Wang *et al.*, "Deep learning towards mobile applications," in *Proc. 38th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2018, pp. 1385–1393.

[515] F. N. Iandola *et al.*, "SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5MB model size," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.

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

[517] X. Zhang, X. Zhou, M. Lin, and J. Sun, "ShuffleNet: An extremely efficient convolutional neural network for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Salt Lake City, UT, USA, Jun. 2018, pp. 6848–6856.

[518] Q. Zhang, L. T. Yang, X. Liu, Z. Chen, and P. Li, "A tucker deep computation model for mobile multimedia feature learning," *ACM Trans. Multimedia Comput. Commun. Appl.*, vol. 13, no. 3s, Aug. 2017, Art. no. 39.

[519] Q. Cao, N. Balasubramanian, and A. Balasubramanian, "MobiRNN: Efficient recurrent neural network execution on mobile GPU," in *Proc. 1st ACM Int. Workshop Deep Learn. Mobile Syst. Appl.*, Niagara Falls, NY, USA, 2017, pp. 1–6.

[520] C.-F. Chen, G. G. Lee, V. Sritapan, and C.-Y. Lin, "Deep convolutional neural network on iOS mobile devices," in *Proc. IEEE Int. Workshop Signal Process. Syst. (SiPS)*, Dallas, TX, USA, 2016, pp. 130–135.

[521] S. Rallapalli *et al.*, "Are very deep neural networks feasible on mobile devices," *IEEE Trans. Circuits Syst. Video Technol.*, to be published.

[522] 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, 2016, pp. 1–12.

[523] L. N. Huynh, R. K. Balan, and Y. Lee, "Demo: DeepMon: Building mobile GPU deep learning models for continuous vision applications," in *Proc. 15th ACM Annu. Int. Conf. Mobile Syst. Appl. Services*, 2017, p. 186.

[524] J. Wu, C. Leng, Y. Wang, Q. Hu, and J. Cheng, "Quantized convolutional neural networks for mobile devices," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 4820–4828.

[525] S. Bhattacharya and N. D. Lane, "Sparsification and separation of deep learning layers for constrained resource inference on wearables," in *Proc. 14th ACM Conf. Embedded Netw. Sensor Syst. CD-ROM*, Stanford, CA, USA, 2016, pp. 176–189.

[526] M. Cho and D. Brand, "MEC: Memory-efficient convolution for deep neural network," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, 2017, pp. 815–824.

[527] J. Guo and M. Potkonjak, "Pruning filters and classes: Towards on-device customization of convolutional neural networks," in *Proc. 1st ACM Int. Workshop Deep Learn. Mobile Syst. Appl.*, Niagara Falls, NY, USA, 2017, pp. 13–17.

[528] S. Li *et al.*, "FitCNN: A cloud-assisted lightweight convolutional neural network framework for mobile devices," in *Proc. 23rd IEEE Int. Conf. Embedded Real Time Comput. Syst. Appl. (RTCSA)*, Hsinchu, Taiwan, 2017, pp. 1–6.

[529] H. Zen, Y. Agiomyrgiannakis, N. Egberts, F. Henderson, and P. Szczepaniak, "Fast, compact, and high quality LSTM-RNN based statistical parametric speech synthesizers for mobile devices," *arXiv preprint arXiv:1606.06061*, 2016.

[530] G. Falcao, L. A. Alexandre, J. Marques, X. Fraz ao, and J. Maria, "On the evaluation of energy-efficient deep learning using stacked autoencoders on mobile GPUs," in *Proc. 25th IEEE Euromicro Int. Conf. Parallel Distrib. Netw. Based Process.*, St. Petersburg, Russia, 2017, pp. 270–273.

[531] B. Fang, X. Zeng, and M. Zhang, "NestDNN: Resource-aware multi-tenant on-device deep learning for continuous mobile vision," in *Proc. 24th ACM Annu. Int. Conf. Mobile Comput. Netw.*, New Delhi, India, 2018, pp. 115–127.

[532] M. Xu, M. Zhu, Y. Liu, F. X. Lin, and X. Liu, "DeepCache: Principled cache for mobile deep vision," in *Proc. 24th ACM Annu. Int. Conf. Mobile Comput. Netw.*, New Delhi, India, 2018, pp. 129–144.

[533] S. Liu *et al.*, "On-demand deep model compression for mobile devices: A usage-driven model selection framework," in *Proc. 16th ACM Annu. Int. Conf. Mobile Syst. Appl. Services*, Munich, Germany, 2018, pp. 389–400.

[534] T. Chen *et al.*, "TVM: An automated end-to-end optimizing compiler for deep learning," in *Proc. 13th USENIX Symp. Oper. Syst. Design Implement. (OSDI)*, Carlsbad, CA, USA, 2018, pp. 578–594.

[535] S. Yao *et al.*, "FastDeepIoT: Towards understanding and optimizing neural network execution time on mobile and embedded devices," in *Proc. 16th ACM Conf. Embedded Netw. Sensor Syst.*, Shenzhen, China, 2018, pp. 278–291.

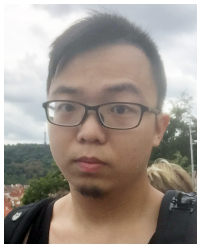
[536] D. Li, X. Wang, and D. Kong, "DeepRebirth: Accelerating deep neural network execution on mobile devices," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, New Orleans, LA, USA, 2018, pp. 2322–2330.

[537] S. Teerapittayanon, B. McDanel, and H. T. Kung, "Distributed deep neural networks over the cloud, the edge and end devices," in *Proc. 37th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Atlanta, GA, USA, 2017, pp. 328–339.

[538] S. Omidshafiei *et al.*, "Deep decentralized multi-task multi-agent reinforcement learning under partial observability," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, 2017, pp. 2681–2690.

[539] B. Recht, C. Ré, S. J. Wright, and F. Niu, "Hogwild: A lock-free approach to parallelizing stochastic gradient descent," in *Proc. Adv. Neural Inf. Process. Syst.*, Granada, Spain, 2011, pp. 693–701.

[540] P. Goyal *et al.*, "Accurate, large minibatch SGD: Training ImageNet in 1 hour," *arXiv preprint arXiv:1706.02677*, 2017.
[541] R. Zhang, S. Zheng, and J. T. Kwok, "Asynchronous distributed semi-stochastic gradient optimization," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, Phoenix, AZ, USA, 2016, pp. 2323–2329.
[542] C. Hardy, E. L. Merrer, and B. Sericola, "Distributed deep learning on edge-devices: Feasibility via adaptive compression," in *Proc. 16th IEEE Int. Symp. Netw. Comput. Appl. (NCA)*, Cambridge, MA, USA, 2017, pp. 1–8.
[543] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. Y. Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Proc. 20th Int. Conf. Artif. Intell. Stat.*, vol. 54. Fort Lauderdale, FL, USA, Apr. 2017, pp. 1273–1282.
[544] K. Bonawitz *et al.*, "Practical secure aggregation for privacy preserving machine learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Security (CCS)*, Dallas, TX, USA, Nov. 2017, pp. 1175–1191.
[545] S. Gupta, W. Zhang, and F. Wang, "Model accuracy and runtime tradeoff in distributed deep learning: A systematic study," in *Proc. IEEE 16th Int. Conf. Data Min. (ICDM)*, Barcelona, Spain, 2016, pp. 171–180.
[546] B. McMahan and D. Ramage, *Federated Learning: Collaborative Machine Learning Without Centralized Training Data*, Google Res. Blog, 2017.
[547] K. Bonawitz *et al.*, "Towards federated learning at scale: System design," *arXiv preprint arXiv:1902.01046*, 2019.
[548] A. Fumo, M. Fiore, and R. Stanica, "Joint spatial and temporal classification of mobile traffic demands," in *Proc. IEEE Conf. Comput. Commun.*, Atlanta, GA, USA, 2017, pp. 1–9.
[549] Z. Chen and B. Liu, "Lifelong machine learning," in *Synthesis Lectures on Artificial Intelligence and Machine Learning*, vol. 10, Morgan & Claypool, 2016, pp. 1–145.
[550] S.-W. Lee *et al.*, "Dual-memory deep learning architectures for lifelong learning of everyday human behaviors," in *Proc. Int. Joint Conf. Artif. Intell.*, New York, NY, USA, 2016, pp. 1669–1675.
[551] A. Graves *et al.*, "Hybrid computing using a neural network with dynamic external memory," *Nature*, vol. 538, no. 7626, pp. 471–476, 2016.
[552] G. I. Parisi, J. Tani, C. Weber, and S. Wermter, "Lifelong learning of human actions with deep neural network self-organization," *Neural Netw.*, vol. 96, pp. 137–149, Dec. 2017.
[553] C. Tessler, S. Givony, T. Zahavy, D. J. Mankowitz, and S. Mannor, "A deep hierarchical approach to lifelong learning in minecraft," in *Proc. Nat. Conf. Artif. Intell. (AAAI)*, San Francisco, CA, USA, 2017, pp. 1553–1561.
[554] D. López-Sánchez, A. G. Arrieta, and J. M. Corchado, "Deep neural networks and transfer learning applied to multimedia Web mining," in *Proc. 14th Int. Conf. Distrib. Comput. Artif. Intell.*, vol. 620. Porto, Portugal, 2018, pp. 124–131.
[555] E. Baştuğ, M. Bennis, and M. Debbah, "A transfer learning approach for cache-enabled wireless networks," in *Proc. 13th IEEE Int. Symp. Model. Optim. Mobile Ad Hoc Wireless Netw. (WiOpt)*, 2015, pp. 161–166.
[556] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.
[557] M. Palatucci, D. Pomerleau, G. E. Hinton, and T. M. Mitchell, "Zero-shot learning with semantic output codes," in *Proc. Adv. Neural Inf. Process. Syst.*, Vancouver, BC, Canada, 2009, pp. 1410–1418.
[558] O. Vinyals, C. Blundell, T. Lillicrap, K. Kavukcuoglu, and D. Wierstra, "Matching networks for one shot learning," in *Proc. Adv. Neural Inf. Process. Syst.*, Barcelona, Spain, 2016, pp. 3630–3638.
[559] S. Changpinyo, W.-L. Chao, B. Gong, and F. Sha, "Synthesized classifiers for zero-shot learning," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Las Vegas, NV, USA, 2016, pp. 5327–5336.
[560] J. Oh, S. P. Singh, H. Lee, and P. Kohli, "Zero-shot task generalization with multi-task deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn. (ICML)*, Sydney, NSW, Australia, 2017, pp. 2661–2670.
[561] H. Wang, F. Xu, Y. Li, P. Zhang, and D. Jin, "Understanding mobile traffic patterns of large scale cellular towers in urban environment," in *Proc. ACM Internet Meas. Conf.*, 2015, pp. 225–238.
[562] C. Marquez *et al.*, "Not all Apps are created equal: Analysis of spatiotemporal heterogeneity in nationwide mobile service usage," in *Proc. 13th ACM Conf. Emerg. Netw. Exp. Technol.*, Incheon, South Korea, 2017, pp. 180–186.
[563] G. Barlacchi *et al.*, "A multi-source dataset of urban life in the city of Milan and the Province of Trentino," *Sci. Data*, vol. 2, Oct. 2015, Art. no. 150055.
[564] L. Liu, W. Wei, D. Zhao, and H. Ma, "Urban resolution: New metric for measuring the quality of urban sensing," *IEEE Trans. Mobile Comput.*, vol. 14, no. 12, pp. 2560–2575, Dec. 2015.
[565] D. Tikunov and T. Nishimura, "Traffic prediction for mobile network using holt-winter's exponential smoothing," in *Proc. 15th IEEE Int. Conf. Softw. Telecommun. Comput. Netw.*, 2007, pp. 1–5.
[566] H.-W. Kim, J.-H. Lee, Y.-H. Choi, Y.-U. Chung, and H. Lee, "Dynamic bandwidth provisioning using ARIMA-based traffic forecasting for mobile WiMAX," *Comput. Commun.*, vol. 34, no. 1, pp. 99–106, 2011.
[567] R. Q. Charles, H. Su, M. Kaichun, and L. J. Guibas, "Pointnet: Deep learning on point sets for 3D classification and segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2017, pp. 77–85.
[568] A. Ioannidou, E. Chatzilari, S. Nikolopoulos, and I. Kompatsiaris, "Deep learning advances in computer vision with 3D data: A survey," *ACM Comput. Surveys*, vol. 50, no. 2, 2017, Art. no. 20.
[569] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Trans. Neural Netw.*, vol. 20, no. 1, pp. 61–80, Jan. 2009.
[570] Y. Yuan, X. Liang, X. Wang, D.-Y. Yeung, and A. Gupta, "Temporal dynamic graph LSTM for action-driven video object detection," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Venice, Italy, 2017, pp. 1819–1828.
[571] M. Usama *et al.*, "Unsupervised machine learning for networking: Techniques, applications and research challenges," *arXiv preprint arXiv:1709.06599*, 2017.
[572] C. Zhou and R. C. Paffenroth, "Anomaly detection with robust deep autoencoders," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 665–674.
[573] M. Abadi and D. G. Andersen, "Learning to protect communications with adversarial neural cryptography," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2017.
[574] D. Silver *et al.*, "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.



Chaoyun Zhang received the B.Sc. degree from the School of Electronic Information and Communications, Huazhong University of Science and Technology, China, and the M.Sc. degree in artificial intelligence from the University of Edinburgh, with a focus on machine learning, where he is currently pursuing the Ph.D. degree with the School of Informatics. His current research interests include the application of deep learning to problems in computer networking, including traffic analysis, resource allocation, and network control.

Paul Patras (M'11–SM'18) received the M.Sc. and Ph.D. degrees from the Universidad Carlos III de Madrid in 2008 and 2011, respectively. He is a Lecturer (Assistant Professor) and a Chancellor's Fellow with the School of Informatics, University of Edinburgh, where he leads the Internet of Things Research Programme. His research interests include performance optimization in wireless and mobile networks, applied machine learning, mobile traffic analytics, security and privacy, prototyping, and test beds.

Hamed Haddadi is a Senior Lecturer (Associate Professor) and the Deputy Director of Research with the Dyson School of Design Engineering, and an Academic Fellow with the Data Science Institute, Faculty of Engineering, Imperial College London. He is interested in user-centered systems, human–data interaction, applied machine learning, and data security and privacy. He enjoys designing and building systems that enable better use of our digital footprint, while respecting users' privacy.