Computer Communications 170 (2021) 19–41

Contents lists available at ScienceDirect

Computer Communications

journal homepage: www.elsevier.com/locate/comcom

Review

Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey

Mahmoud Abbasi [a], Amin Shahraki [b,c,*], Amir Taherkordi [b]

[a] Department of Computer Sciences, Islamic Azad University, Mashhad, Iran
[b] Department of Informatics, University of Oslo, Oslo, Norway
[c] Faculty of Computer Sciences, Østfold University College, Halden, Norway

ARTICLE INFO

Keywords:
Network Traffic Monitoring and Analysis
Network management
Deep learning
Machine learning
Survey
NTMA
Edge Intelligence
IoT
QoS

ABSTRACT

Modern communication systems and networks, e.g., Internet of Things (IoT) and cellular networks, generate a massive and heterogeneous amount of traffic data. In such networks, the traditional network management techniques for monitoring and data analytics face some challenges and issues, e.g., accuracy, and effective processing of big data in a real-time fashion. Moreover, the pattern of network traffic, especially in cellular networks, shows very complex behavior because of various factors, such as device mobility and network heterogeneity. Deep learning has been efficiently employed to facilitate analytics and knowledge discovery in big data systems to recognize hidden and complex patterns. Motivated by these successes, researchers in the field of networking apply deep learning models for Network Traffic Monitoring and Analysis (NTMA) applications, e.g., traffic classification and prediction. This paper provides a comprehensive review on applications of deep learning in NTMA. We first provide fundamental background relevant to our review. Then, we give an insight into the confluence of deep learning and NTMA, and review deep learning techniques proposed for NTMA applications. Finally, we discuss key challenges, open issues, and future research directions for using deep learning in NTMA applications.

Contents

* Corresponding author at: Department of Informatics, University of Oslo, Oslo, Norway.
E-mail address: am.shahraki@ieee.org (A. Shahraki).

https://doi.org/10.1016/j.comcom.2021.01.021
Received 10 September 2020; Received in revised form 11 December 2020; Accepted 19 January 2021
Available online 3 February 2021

1. Introduction

During the last years, NTMA have received much attention as a significant research topic in supporting the performance of networking [1]. As common solutions in network management, NTMA techniques have been introduced both by industry and academia [2,3]. Although different NTMA techniques have been introduced, emerging networking technologies and paradigms have made establishment of efficient networks complex. New networks with thousands of nodes e.g. Internet of Things (IoT) need to be monitored on a regular basis to maintain their performance [4]. Different purposes in networking trigger the network managers to evaluate a network in terms of, e.g. security challenges, supporting Quality of Service (QoS) requirements, and resource consumption improvement, to name a few [5]. These purposes are satisfied by applying NTMA techniques e.g. anomaly detection, network traffic classification, fault management, and traffic prediction.

NTMA techniques are classified into two main groups: (1) active methods, and (2) passive methods [6]. Active methods involve generating and injecting probe traffic into a network in order to learn about the state of the network. More precisely, test traffic data is injected into the network based on scheduled sampling, and then different network performance metrics will be measured. Examples of the metrics include network throughput, packet loss ratio, latency, and jitter (or delay variation). Since active monitoring methods present a real-time insight about performance, they are the primary methods to control Service Level Agreement (SLA) based services. In contrast, passive methods are mainly used to monitor and analyze real network traffic in the network. Passive methods have received much interest from industry for managing and planning purposes [7,8]. Passive methods, unsurprisingly, do not need another site in the network to be involved. These methods can be used to carefully monitor traffic, particularly in post-event situations e.g. fault tolerance and troubleshooting. Moreover, they are ideally suited for obtaining deep insights into the user's Quality of Experience (QoE). The applications of active and passive methods are summarized in Table 1 [9].

The growth of the communication systems and networks in terms of the number of users and the amount of generated traffic, poses different daily challenges to NTMA, including: (1) storing and analyzing traffic data, (2) using traffic data for business goals through gaining insight, (3) traffic data integration, (4) traffic data validation, (5) traffic data security, and (6) traffic data acquisition [10,11]. The unprecedented increase in the number of connected nodes and the volume of data amplify the network complexity, calling for continuing studies to analyze and monitor the networking performance [12,13]. Furthermore, the availability of massive and heterogeneous amount of traffic data necessitates adopting new approaches for monitoring and analyzing the network management data. Due to these challenges, most works focus specifically on one aspect of NTMA, e.g. anomaly detection, traffic classification, or QoS [14].

Among the challenges mentioned above, traffic data acquisition presents enormous technical difficulties in the field of NTMA, particularly for active measurements as one has to use *probes* to evaluate the progression of crucial network parameters over time. *Probes* are among the most efficient methods to obtain insights into the end-to-end performance experienced by the end-users. Active and passive probes are two common strategies that can improve the performance of end-to-end measurement and determine QoS and QoE by delivering granular traffic data [15]. An active probe tries to emulate the network traffic and then send emulated traffic within the network to measure the end-to-end performance (e.g., latency). In comparison to the active probes, passive probes present a distinct viewpoint of the network. Passive probes are placed on links in the network, and they inquire all the traffic that transmits through the connection being monitored.

Regarding particular network scenarios and the purposes of traffic data gathering (e.g., traffic classification and intrusion detection), one can define different requirements for traffic data acquisition. In other words, it is not required to acquire all available data from a network in a traffic acquisition task. Hence, the network packets are commonly regarded as the central targets that should be examined in the traffic data acquisition tasks. To monitor the network traffic to evaluate its performance, there are two fundamental methods including Shallow Packet Inspection (SPI) and Deep Packet Inspection (DPI). As the former refers to gather information from headers of packets of network traffic, the latter processes all contents of a packet including user's data. Probes can use both techniques to gather network information but DPI has some disadvantages including:

- Analyzing users' data can jeopardize the privacy of users.
- Processing the whole packet needs more time and resources compared to processing the header.
- In some types of network traffic, e.g. Virtual Private Network (VPN) and encrypted network traffics, DPI is unusable.

Based on the challenges mentioned above for using DPI, most of probes in new NTMA techniques use SPI. After providing a background to the probes, we highlight that one of the significant challenges in NTMA is acquiring a large amount of reliable traffic data. To deal with this challenge, some tools, e.g., [16] and [17], as data acquisition and data collection architectures have been proposed during recent years. However, an adaptive and efficient data acquisition approach that can be pervasively used within heterogeneous and large-scale modern networks is still missing [18].

The network packets are yet the most widespread data format for network traffic collection. However, the majority of the network packet collection methods are confronted by the packet loss problem, particularly when they confront a large amount of traffic [19]. Moreover, these methods have difficulties with high-speed links and become ineffective due to their low capability. The flow-based data gathering techniques are another popular data collection mechanism. A flow network is a set of network packets with the same features, such as source/destination IP address and source/destination ports. Compared to the packet-based mechanisms, the flow-based data gathering techniques can decline the number of needed tasks for packet analysis and provide a better performance, especially in Gigabit Networks. Nonetheless, packet and flow filtering can seriously challenge these techniques. Several survey papers about traffic data acquisition methods, approaches, and architectures have been published e.g. [19–21].

Modern networking solutions are under pressure of new phenomenon, known as big data [1]. This fact is based on special characteristics of network management data e.g. high volume, high velocity, high veracity, and high variety [22]. Network management data refers to all data that reflects the network situation, mainly extracted from headers of packets (packet-level feature) e.g. packet delay, time stamps, and type of packet. NTMA techniques can be considered as one of the main consumers of big data. Besides, it is becoming a critical field of study in the context of big data analytics due to data complexity. Conventional data processing techniques for NTMA include:

- Mathematical and statistical methods (e.g. regression for time-series analysis)

Table 1
Categories of active/passive methods applications.

Active methods	Passive methods
Direct and end-to-end analysis	Comprehensive traces for troubleshooting
Quality of service (QoS)	Quality of experience (QoE)
Real-time monitoring	Diagnosis of protocol issues
Network and service performance monitoring	Non-real-time monitoring
Real-time monitoring of end-to-end transport processes	Service and customer experience monitoring

- Machine Learning (ML) algorithms and big data processing approaches (e.g. supervised learning for intrusion detection)

NTMA techniques should perform a sequence of steps for transforming raw traffic data into useful information. Using the conventional methods for big data analytics faces multiple challenges and issues, including accuracy, high-speed analytics, and effective processing of big data in a real-time manner [23]. Furthermore, based on new paradigms like Internet of Things (IoT) [24], a high number of connected devices produce a massive volume of raw data every day, and thus, we need more effective methodologies to monitor and analyze such massive amount of raw data in a more efficient way in terms of processing time and space.

As mentioned above, ML techniques have received much attention in NTMA techniques. ML techniques are grouped into four groups as mentioned below: (1) Supervised Learning, (2) Semi-supervised Learning, (3) Unsupervised Learning, and (4) Reinforcement Learning. Among various ML techniques, Deep Learning (DL) is a key step to considerably ease the analytics and knowledge discovery in the big data field [25]. DL has been used in many fields, including computer vision, healthcare, transportation, and smart farming. In addition, DL has also gained attention from technology-based companies (TBC). Large companies such as Twitter, YouTube, and Facebook produce huge amounts of data every day, and hence, it is crucially important for them to handle this big data [26]. DL algorithms are utilized to analyze produced data and extract meaningful information because for traditional data processing techniques it is almost impossible to handle such a huge amount of data. This paper gives an insight into the confluence of two emerging technologies, i.e. NTMA and deep learning. The paper does not cover all the possible NTMA applications as there is a long list of NTMA applications in the literature. We only investigate four key applications, including traffic classification, traffic prediction, fault management and network security as the main topics of NTMA, see Fig. 8.

1.1. Contributions

This work is intended for the vertical domains and researchers in the field of communication systems and networks, who want to use AI-based analytics systems on top of their communication infrastructures. The main contributions of our work in this paper include the following:

- We review the well-known literature that has surveyed traditional learning-based techniques for NTMA and highlights their differences with our work.
- To use deep models for NTMA, we discuss the key characteristics and applications of NTMA.
- We review the advanced DL techniques used in the main NTMA applications and their applicability in the NTMA domain.
- We survey the literature that uses DL techniques in four fields of NTMA including traffic classification, fault management, traffic prediction, and network security.
- We present the challenges and future research directions regarding NTMA and DL.

2. Related work

To the best of our knowledge, this paper is the first that investigates the relation between NTMA and DL and reviews the applications of DL models in NTMA. A few works exist in the literature that have focused on data mining applications and traditional ML models in NTMA. It is noteworthy to mention that a limited number of papers present DL models for some NTMA applications, e.g., traffic classification. The work presented in [27] by Rezaei et al. surveyed DL models for encrypted traffic classification. This paper addressed different DL-based classification models for network traffic classification. Nevertheless, it did not review other NTMA applications, which are the focus of our paper.

In [28], Aniello et al. surveyed basic ML models (supervised, unsupervised, and semi-supervised learning) in the context of malware analysis. Moreover, the related challenges and issues are discussed in this paper. However, the authors did not investigate the importance of DL in malware analysis and detection.

The work in [29] by Conti et al. conducts an in-depth survey on network traffic analysis. They categorize the relevant works into three criteria: (1) the aim of the analysis, (2) the point in the network where the traffic is monitored, and (3) the selected mobile platforms. They reviewed several algorithms, such as Naive Bayes, C4.5 decision tree, Random forest, k-means, to name a few. The work focuses on mobile devices and compares analysis methods, validation techniques, and achieved results. In addition, the focus of [29] was mainly on conventional ML algorithms, whereas our work targets DL models.

In [30], DL models and architectures for network traffic control systems have been studied by Fadlullah et al. Since this paper primarily covers the network infrastructure, it differs from our survey that targets the use of deep models in NTMA.

D'Alconzo et al. [1] addressed a big data approach for NTMA. The authors surveyed the works that employ big data approaches to understand network traffic data in that work. Moreover, they briefly reviewed big data analytics (e.g., traditional machine learning) for four main NTMA applications, i.e., traffic classification, traffic prediction, fault management, and network security. As the main difference, DL models have not been considered in this paper.

Finally, Verma et al. [31] surveyed real-time analysis of big IoT data. In this work, the authors reviewed the latest network data analytics methods, which are appropriate for real-time IoT network data analytics. Moreover, in that paper, the foundations of real-time IoT analytics, use cases, and software platforms are discussed. Similar to the works mentioned above, that paper did not study DL models for data analytics purposes.

3. Overview of NTMA

NTMA refers to a range of techniques to monitor the network traffic at an appropriate level of granularity (e.g., at the packet level). The NTMA techniques obtain deep insight into the operation and performance of the network and users' behavior [32]. In the context of communication systems and networking, NTMA has played a critical role in:

1. understanding how networks work and monitoring the performance of networks
2. how consumers are using resources and optimizing the use of resources
3. how to effectively control and manage the telecommunication infrastructures to provide SLA

Due to the explosive growth of connected devices and the volume of traffic data, more advanced NTMA techniques are needed to ensure communication systems' stability and availability. In the following, we focus on the general framework for NTMA, comprising five steps. Most existing research works follow all or part of the framework based on Fig. 1.

The first step towards NTMA is to clearly define the goals of NTMA. As mentioned above, the typical goals include traffic classification, traffic prediction, fault management, and network security. Depending on the target goal, one may need to work towards different sub-goals in order to serve the primary goal. For example, if the aim of NTMA is to classify network traffic, the sub-goal can be the categorization of traffic data into different classes based on their labels, such as VPN and non-VPN traffic or Firefox and Chrome. The second step is to gather network management data by using passive or active monitoring methods. Due to the fact that these two methods provide different views

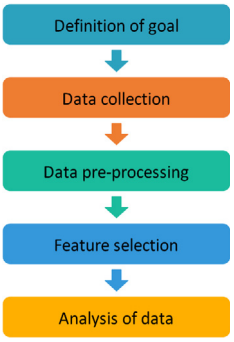


Fig. 1. General framework for NTMA procedure.

of the network status, they can be used in cooperation to take benefit of both methods.

Data preprocessing and cleaning can greatly affect the performance of NTMA, especially ML-based approaches. In a network, the distribution of packet-level features may change by some common activities, such as packet retransmission and duplicated ACKs. Hence, removing such network management data can improve NTMA applications' performance, e.g., traffic prediction [33]. Normalization is another preprocessing technique towards improving the performance of NTMA applications, especially it is crucially important for ML- and DL-based approaches.

Then, after data preprocessing, NTMA has to go through a feature selection step to select the most informative features for serving the goal. Feature selection can be performed automatically or manually. In the former, feature selection algorithms are used to extract the most relevant features, and the latter uses domain knowledge to perform feature selection.

After the above steps, data analysis experts perform an in-depth analysis of the preprocessed data in order to extract meaningful information. As mentioned in the introduction section, mathematical and statistical methods, ML algorithms, and big data approaches [34] are the traditional approaches to retrieve meaningful knowledge from the raw data. Selecting the most appropriate model or technique from the existing approaches is important for a reliable and reproducible statistical inference. ML-based approaches overcome the mathematical and statistical methods by their abilities to discover hidden patterns about the raw data. In communication systems and networks, ML methods have been used in many applications such as Intrusion Detection System (IDS) [35], anomalies detection [36], monitoring [37], and pattern discovery [38].

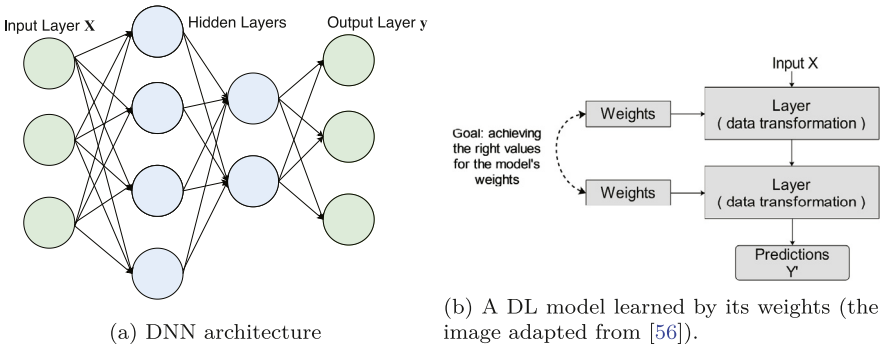
The enormous volumes of human- and machine-generated traffic, e.g., web surfing and IoT networks, calls for the design of scalable algorithms and tools to handle such a huge amount of data in a short time. Fortunately, big data frameworks such as Hadoop [39] and Spark [40] are introduced to process a huge volume of data in a short time. This is mainly due to their distributed architecture and the possibility to accelerate the process through parallel processing and moving the computation program to the node, which generates the data. [41].

Network management data is different from conventional big data. Hence, to ease understanding the requirements for analysis of the data, we find it necessary to investigate the characteristics of the network management data and highlight its major differences with conventional big data. The network management data possesses several characteristics in common, including:

- *Heterogeneity*: The set of devices served in a communication system and network can be vastly heterogeneous, and these devices consume or generate different types of data resulting in heterogeneity in both network traffic and network management data. Smartphones, vehicles, sensors, smart appliances, and IoT devices are examples of devices that can benefit from being served by communication systems and networks.
- *Time and space correlation*: The pattern of network traffic, especially cellular network, shows a very complex behavior because of various factors, including device mobility and heterogeneity, different communication protocols, patterns of usage, and user requirements. Moreover, recent works propose to use temporal and spatial features of network traffic and network management data to obtain a finer insight into the complex pattern hidden in network traffic data [42]. This is mainly because many applications and services are provided for specific locations, and thus temporal and spatial information is attached to the traffic data and network management data.
- *Noisy data*: By noise, we mean any unwanted change reshapes the values of data. In the context of networking, noise may be created by some common events, caused by e.g. faults, attacks, etc. For example, in multi-hop routing in IoT networks, inefficient queuing management in middle nodes can cause jitter.
- *High-speed rate and large-scale streaming data*: One of the distinguishing characteristics of network traffic data is streaming and high data rates, particularly in services such as streaming media, P2P applications and live game streaming. In this case, network management data can be affected by the volume and velocity of streams.
- *Implications of data protection*: The emergence of network traffic encryption protocols has considerably enhanced the privacy and security of communication. Using encryption technologies guarantees, to some extent, that third-parties will not have access to data. Nevertheless, increasing the popularity of network traffic encryption poses some new challenges to NTMA. For example, encrypted traffic can reduce the performance of IDSs in identifying malicious traffic. Many Internet services and applications use encryption protocols, such as Hypertext transfer protocol secure (HTTPS), for secure communication. Consequently, a small amount of information stays visible in network packets or inadequate information is available. Under these scenarios, performing NTMA tasks, such as traffic classification and fault management, is not trivial. For instance, as a classification technique, DPI [43] runs into challenges with encrypted traffic and privacy policy restrictions (see Section 5.1). Flow-based NTMA applications will face less encryption-related challenges because data is sequenced and lower transmission is required. However, more obscure packet content is inevitable, which makes the network traffic analysis more restricted. Moreover, in some situations, IP layer encryption may be applied, which obscures the TCP/UDP headers, and consequently, it is almost impossible to know the original port numbers (see Section 5.1, port-based techniques).

Despite this fact that extracting hidden patterns and knowledge from big data is critical [44–46], it is not a complicated task as it looks. For such a difficult and demanding task that needs capabilities beyond the conventional learning-based mechanisms, novel learning approaches, learning models, and techniques are required [47].

In the past few years, many NTMA applications, such as traffic classification, traffic prediction, and network security, gained attention of the academia and industry. The reason behind using NTMA consists in this fact that NTMA applications play an important role in network and resource management, network auditing methods, and intrusion detection [48]. Deep learning is one of the powerful AI-driven techniques for gaining insights into communication systems and networks. DL has been employed for many NTMA applications in recent years, e.g., traffic classification and prediction [49,50]. Motivated by this fact that classical ML algorithms are not able to effectively meet the emerging analytic requirements of communication systems and networks, DL achieves increasing popularity among scholars to address these requirements.



(a) DNN architecture

(b) A DL model learned by its weights (the image adapted from [56]).

Fig. 2. Deep learning network architecture and learning process.

Generally, DL algorithms represent two significant improvements upon the classical machine learning methods.

- They eliminate the need for the feature engineering phase via deploying automatic feature learning [51]. Hence, some useful and meaningful features that might not be apparent to manual feature engineering approaches can be obtained easily by DL algorithms.
- DL algorithms improve learning performance in terms of accuracy and loss through learning hidden and high-level patterns from data. It can be achieved by feeding a huge volume of traffic data into DL models.

In this work, we explain a broad spectrum of DL architectures and survey the NTMA use cases that take advantage of DL models. This paper focuses on four main NTMA applications that can be utilized for different services and vertical domains.

4. Deep learning models

AI has attracted lots of interest in recent years for many use cases, such as self-driving cars [52], chatbots [53], virtual assistants [54], etc. [55]. AI's history goes back to the 1950s, when researchers tried to automate intellectual tasks that humans normally perform. For a very long time, many experts were arguing that by formulating a large set of explicit rules for manipulating knowledge, they can realize human-like artificial intelligence. This approach, also known as symbolic AI, was a dominant method for achieving human-level artificial intelligence between the 1950s to the late 1980s. Despite this fact that symbolic AI successfully dealt with well-defined tasks, such as playing chess, it encountered difficulty with solving more complex tasks, such as speech recognition and image classification. To address this challenge, machine learning has arisen as a new approach.

The emergence of machine learning introduces a new paradigm in programming. In the paradigm of symbolic AI, human-agent enters rules (a program) and data to be manipulated according to these rules, and yield results. In contrast, in machine learning, the human agent enters data and the expected results from the data, and then the learning model yields the rules. Then, these rules are applied to new data in order to achieve original results. Machine learning systems are trainable rather than explicitly programmable. This means a massive amount of data feed into these systems to find meaningful features in this data. Then, these features can be used to produce rules for automating the task. Machine learning usually struggles with big and sophisticated datasets, such as image datasets with thousands or even millions of instances. For the classical statistical analysis, such as Bayesian analysis, it is almost impossible to handle such big datasets. Consequently, machine learning and particularly DL shows relatively little theory of mathematics and is an engineering-oriented approach.

DL is a specific sub-field of ML, in which Deep Neural Network (DNN) is used to find data representation at each layer [56]. The deep in DL definition refers to the idea of successive layers of representations. Moreover, the number of layers for modeling the data is known as the depth of the model. For complex tasks such as image recognition, DL models often have tens or even hundreds of successive layers of representations. In contrast to DL, other machine learning models often involve one or two layers for the data representation. A DNN architecture is presented in Fig. 2a.

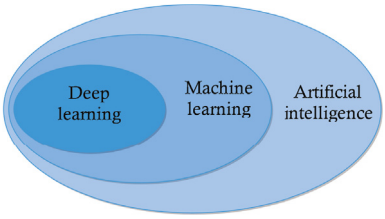


Fig. 3. Difference between artificial intelligence, machine learning, and deep learning.

As a general definition, one may argue that machine learning is mapping inputs data (e.g., video and images) to targets (e.g., the label "dog"), which is achieved by exposing the model to many instances of input and targets. Similarly, one sees that DL performs the input-to-target mapping through deep successive layers of data transformations. The DL model learns these transformations by observing many examples of input/target.

In a DL model, the layer's weights, also known as parameters, determine what transformations would be performed to a layer's input data. According to a simple definition of 'weights', they are a set of numbers (see Fig. 2b). In the context of DL, learning refers to finding a set of correct values for the weights of all layers in a model so that the model will precisely map inputs to their related targets. Due to the fact that DL models may have tens of millions of parameters (weights), determining the correct value for all of these parameters is a challenging task. Fig. 3 shows the relationship between AI, machine learning, and DL in summary. In the following, we investigate the major DL models in detail.

4.1. Multi-layer perceptron

A well-known category of a DL model is the feed-forward deep network or multilayer perceptron (MLP). An MLP model is an artificial neural network mapping some examples of input data to target values [57]. The network is formed by composing multiple simple layers (at least three layers). We can consider the application of each layer as providing a new representation of each data point.

The main objective of an MLP model is to approximate some function f^*. For instance, in a classifier model, $y = f^*(x)$ maps an input data x to a label y. An MLP defines a mapping $y = f(x;\theta)$ and finds the correct values for parameters θ that lead to the closest

function approximation. In the feed-forward deep networks, the feed-forward definition refers to the idea that input data goes through the function being evaluated from x, then flows through the intermediate computational units employed to define f, and finally flows to the output y. One must note that in an MLP model, there are no feedback connections to feedback the outputs of the model into itself.

An MLP has at least three layers, in which computational units (or neurons) are densely connected to the next layer units (see Fig. 2a). We assume an input data vector x and a standard MLP network. Given these settings, the MLP carries out the following operation:

$$y = \sigma(W.x + b). \tag{1}$$

In this expression, y is the output of the layer, W denotes the learning weights, and b indicates the bias neurons. Also, $\sigma(.)$ is an activation function that aims to improve the model's training by allowing the non-linearity of it. The most common non-linear activation functions are as follows:

- Sigmoid (or logistic), Where $sigmoid(x) = \frac{1}{1+e^{-x}}$,
- Tanh (or hyperbolic tangent), Where $tanh(x) = \frac{e^{x}-e^{-x}}{e^{x}+e^{-x}}$,
- ReLU (or Rectified Linear Unit), Where $ReLU(x) = max(x, 0)$,
- Leaky ReLU, Where $LeakyReLU(x) = max(\alpha * x, x)$, and α is a small constant, e.g., 0.1.

ReLU and Leaky ReLU activation functions are proposed to deal with a critical problem in other activation functions, called gradient vanishing. The problem refers to when the loss function gradients will be vanishingly small and cannot propagate through layers.

4.2. Convolutional networks

Convolutional networks, also called Convolutional neural networks (CNNs), are a specific type of neural network that specialize in processing grid-like data [58]. Examples of this data type are time-series and images, which can be regarded as a 1-dimensional grid and 2-dimensional grid of pixels, respectively. Convolutional networks have been extensively used in diverse real-world problems, such as Natural Language Processing (NLP), computer vision, speech recognition, etc. The term "convolutional" in convolutional neural networks supports this idea that CNNs use a mathematical operation called convolution. In its most common form, the convolution operator is a specific type of linear operation that performs the integral of the product of two functions/signals. In other words, CNNs are neural networks that employ convolution operators instead of general matrix multiplication in at least one of their network layers. CNNs apply three key principles that can be applied to promote the performance of an ML system by reducing a model parameter space: parameters or weights sharing, sparse interactions, and equivariant representations.

The large dimensionality is an obvious disadvantage of DNN architecture, especially when the input data is too large and complicated, e.g., images. Towards dealing with this problem, the convolution operator (or convolution layer) has been introduced as an alternative for full connectivity in the DNN architecture. A graphical description of deep CNN architecture is presented in Fig. 4. The CNN accepts multi-channeled images (e.g., cars and ships) as the input for training purposes. The CNN takes the benefit of multiple convolution layers with non-linear activation functions to alleviate the input's complexity (i.e., images) and produce the output, i.e., the probability of each image belonging to a class (or category). In CNN, each input zone is connected to a neuron in the output, aka local connectivity. Each layer employs various filters to recognize abstract concepts, e.g., the boundary of a vehicle. The CNN can learn higher-level features, such as different vehicle parts, at the deeper layers. Filters are not defined beforehand in a CNN; instead, it automatically learns each filter's value during the training phase. Moreover, the CNN uses the pooling layer as a method for down sampling. In the output layer, a classifier is applied to use the high-level features for the classification task. The interested readers are referred to [59] for more details.

4.3. Recurrent neural networks

Recurrent neural networks (also known as RNNs), are a category of artificial neural networks appropriate for analyzing sequential data [60]. Unlike CNNs that are designed to work with the grid-like topology data, e.g., images, RNNs are neural networks that have specialized characteristics for operating on a sequence of values $x_1, x_2, \ldots, x_t$. In addition, most RNNs are able to handle variable-length sequences. The clever idea behind the recurrent networks and some other machine learning and statistical methods are to share parameters over different layers of a model to extend the use of the model for data instances with different forms. The parameter sharing task is especially crucial when a particular item of data may appear at multiple positions within the sequence. This optimization technique typically leads to significant savings of memory in machine learning models [61]. It is also possible to employ RNNs for 2-dimensional spatial data such as images. The key advantage of using recurrent networks over conventional neural networks is that RNN is able to handle sequence of data so that each sample can be considered to be dependent on previous ones.

As mentioned, RNNs are specialized to model sequences, where there is a strong sequential correlation among the sequence samples. At each time step, RNN uses the given input and the information related to what has been observed as yet (i.e., state) to generate output. Note, this information is transferred through recurrent connections between units, as shown in Fig. 5a. Assume we have a sequence of input elements $\mathbf{x} = (x_1, x_2, \ldots, x_t)$. Under this setting, a RNN conducts the following computations:

$$S_t = \sigma_s(W_x x_t + W_s s_{t-1} + b_s)$$

$$h_t = \sigma_h(W_h s_t + b_h)$$

where S_t is the state of the RNN at time step t and it acts as a memory unit for the RNN. To compute the value of S_t, a function of the input value at time t (i.e., x_t) and previous state of the RNN, i.e., s_{t-1}, has been calculated. Moreover, W_x and W_h are weights to be learned during the training process, and b_s and b_h are biases. In the RNN, the Backpropagation Through Time (BPTT) algorithm [62] is used to update the weights or train the network.

4.4. Long short-term memory

RNN can use self-loops to store the gradient of recent input events for long durations. This is the core idea of long short-term memory (LSTM) model [63]. This feature is potentially important for a wide spectrum of applications, such as speech recognition [64], handwriting recognition [65], machine translation [66], handwriting generation [67], image captioning [68] and parsing [69]. LSTM has been introduced to deal with two serious problems, i.e., gradient vanishing and gradient blow up, in the former techniques. More specifically, by using the conventional gradient-based learning methods such as BPTT and real-time recurrent learning (RTRL), error signals may reduce or increase when they back-propagate over the model. LSTM network is proposed to solve the problems of error signals back-flow, by introducing the idea of using a collection of gates. LSTM has been successfully applied to many problems, such as speech recognition and text classification. A graphical illustration of the structure of an LSTM is presented in Fig. 5b. In this structure, 'forget gate' decides what information from the cell state will forget as they are unrepresentative. Indeed, the forget gate makes this decision through a sigmoid layer. The forget gate performs the following operation:

$$f_t = \sigma(W_{xf} X_t + W_{hf} H_{t-1} + W_{cf} \odot C_{t-1} + b_f),$$

In this expression, '$\odot$' operation is Hadamard or element-wise product, C_t represents the cell state outputs, H_t denotes the hidden states. Forget gate alleviates the gradient vanishing and gradient blow up and significantly promotes the performance of LSTM than RNN.

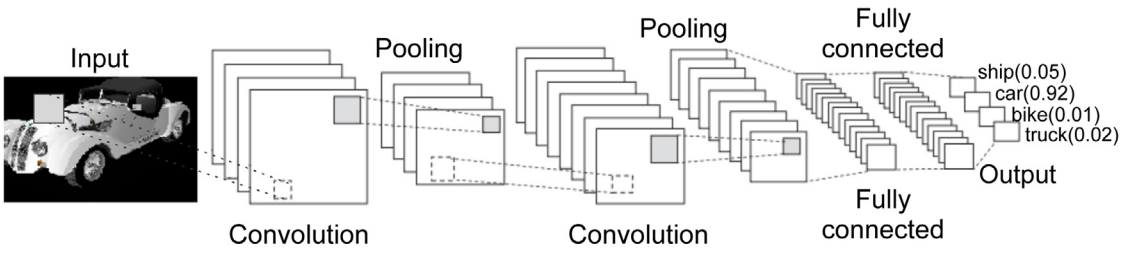
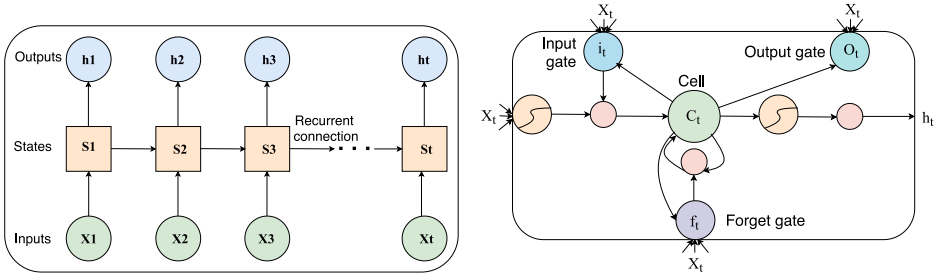


Fig. 4. CNN architecture.



(a) A recurrent network (b) The internal structure of an LSTM

Fig. 5. Typical structures of RNN and LSTM.

Another essential function of the LSTM is to decide what new information should be stored in the cell state. Towards this end, input gates i_t decides which information will be updated, and this information will provide an update to the old cell state (i.e., C_{t-1}).

$$i_t = \sigma(W_{xi}X_t + W_{hi}H_{t-1} + W_{ci} \odot C_{t-1} + b_f),$$

$$C_t = f_t \odot C_{t-1} + i_t \odot tanh(W_{xc}X_t + W_{hc}H_{t-1} + b_c),$$

And the final step for the LSTM is to decide what should go to output, based on the cell state. It can be done by output gates (i.e., o_t), which decides what information of the cell state will go to output. The cell state also goes through a tanh and then multiply by the output gates.

$$o_t = \sigma(W_{xo}X_t + W_{ho}H_{t-1} + W_{co} \odot C_t + b_o),$$

$$H_t = o_t \odot tanh(C_t),$$

4.5. Auto-encoders

In the most general sense, an auto-encoder or AE is a neural network that is used to efficiently learn how to copy its inputs to its outputs. AE has a hidden layer, called **h**, which is responsible for describing a **code** that stands for the input. An AE network consists of two main components: an encoding function $h = f(x)$ and a decoding function $r = g(x)$. A graphical description of the structure of an AE is depicted in Fig. 6. AEs are not designed for this purpose to copy their inputs to their outputs. Instead, they attempt to copy only essential aspects of the inputs that contain useful properties of the data. Assume there is a training set of $\{x^1, x^2, x^3, \ldots, x^n\}$ where for each data sample we have $x^i \in R^n$. The objective of the AE is to reconstruct the network input by reducing the reconstruction error, i.e., $y^i = x^i$ for $i \in \{1, 2, 3, \ldots, n\}$. In other words, the AE attempts to learn a compressed representation of the input data. Given this objective, the AE tries to minimize the following loss function:

$$\Gamma(W, b) = \| x - F_{W,b}(x) \|^2,$$

in which W and b are the vectors of the network weights and biases, respectively, and $F_{W,b}(x)$ is the identity function that the AE tries to learn. AEs are primarily employed as an unsupervised framework for the automatic feature extraction process. More specifically, the outputs of AE's output layers can be assumed as an abstract set of discriminative features for the categorization task, especially for high dimensional data.

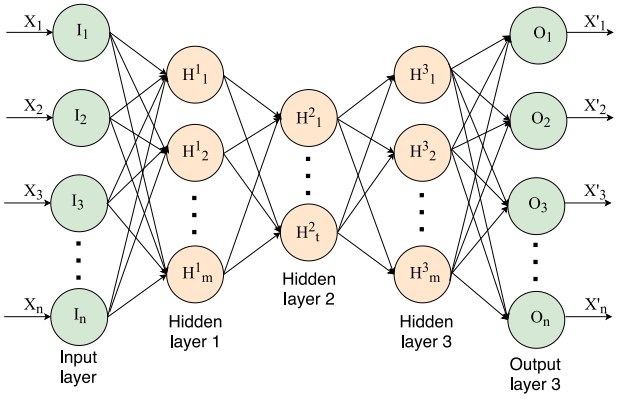


Fig. 6. The general structure of an AE.

4.6. Deep generative models

Deep generative models or generative deep learning is an effective learning mechanism for any input data distribution through unsupervised learning. There are several kinds of generative models, such as Boltzmann machines [70], restricted Boltzmann machines [71], deep belief networks (DBNs) [72], deep Boltzmann machines [73], and Boltzmann machines for real-valued data [74]. According to a broad definition, a deep generative model characterizes how a specific dataset is generated with regard to a probabilistic model. Through sampling from this model, one can produce new data. Deep generative models attempt for integrating the interpretable representations and quantified uncertainty (UQ) provided by probabilistic models, into the scalability and flexibility of deep learning.

Generally, most machine learning models are discriminative models in nature [75]. Discriminative models do not care about how the

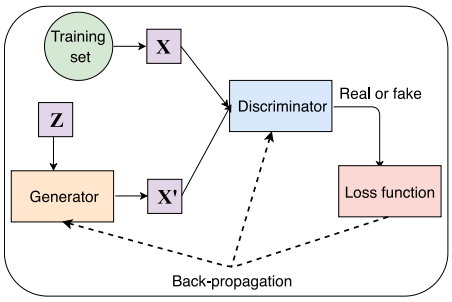


Fig. 7. Typical GANs architecture.

data was produced; they categorize a given input data. In contrast, generative models specify how the data was produced in order to categorize input data. Another critical difference between discriminative and generative modeling consists in the fact that in the former, each instance in the training dataset has a label. Hence, discriminative models are considered synonymous with supervised learning. In contrast, generative models usually use unlabeled dataset; however, they can also be employed with a labeled dataset in order to learn how to produce data instances from each distinct class label.

Generative Adversarial Network (GAN) is a widespread method for DL-based generative modeling. GAN is a supervised training framework that simultaneously trains two sub-models: the generator model G that tries to create new instances from training data and the discriminative model D that attempts to categorize instances into real (from the training data domain) or fake (generated). Both G and D are trained through playing in a zero-sum game. More specifically, G tries to produce new instances as real as possible and to maximize the probability of D to be confused in recognizing these instances. The responsibility of D is to differentiate between real instances and fake instances. In a GAN network, the overall goal is to solve a two-player minimax problem. The overall structure of a GAN has been presented in Fig. 7

The aforementioned deep models, their attributes, and characteristics are summarized in Table 2.

5. DL and NTMA

Machine learning techniques, especially DL algorithms, are among the most popular techniques for network traffic data processing. This is arguably explained by the fact that modern communication systems and networks, e.g., IoT and cellular networks, have distinguishing characteristics that fit DL algorithms. These features include big data generation, complexity, multimodal data, being large-scale, the growing number of protocols in such networks, etc. The traditional methods for NTMA have their own problems; for example, they are inaccurate or highly dependent on human experts. Unlike the traditional methods, DL-based techniques have some advantages to be used as NTMA techniques listed below:

- DL models do not require considerable human effort and they are not dependent on the choice of features. DL models can employ different representative layers and efficient algorithms to extract hidden knowledge from massive amounts of traffic data without feature engineering. This advantage of the DL models is very efficient for NTMA techniques as most of the network management data is unlabeled or semi-labeled [76].
- DL models (e.g., LSTM) are capable of working with temporal–spatial data, capturing related dependencies. Most network management data gathered as time-series datasets can fit to be analyzed by DL models with high accuracy. Deploying accurate and effective techniques for different NTMA applications is paramount of importance. For example, accurate mobile traffic prediction is important for traffic engineering (e.g., on-demand resource allocation), saving energy, and user mobility analytics in cellular networks (e.g., movement forecasting).
- In new computing paradigms, e.g. Fog and Edge, involved devices are equipped with high-performance computational equipment e.g. Graphical Processing Units (GPUs) to process data [77]. As these computing paradigms are widely used to perform NTMA, DL techniques can be implemented by e.g. Fog and Edge equipment to monitor the network. In addition, new machine learning paradigms, e.g. federated learning are mainly designed to implement deep learning techniques in a distributed manner [78,79]. Implementing DL models by the new ML paradigms enables the DL to train its model separately in each machine. It is considered as a great advantage as NTMA techniques need to gather network management information from different machines to a central point. Using the distributed machine learning techniques, DL models can be trained separately in each machine, reducing the network overhead and jeopardization of security and privacy.

In the next subsections, we explore the abilities of DL models for four main NTMA applications, as shown in Fig. 8.

5.1. DL for traffic classification

In its broadest definition, network traffic classification refers to a system in which a program assigns traffic flows to the sources (e.g., applications and protocols) that produce them. Traffic classification has attracted ever-increasing interest over the years as a crucial step towards the network management process. Moreover, traffic classification covers a wide variety of applications in QoS purposes, pricing in Internet service providers (ISPs), anomaly detection, etc. Due to the continuing growth in Internet-based applications and the number of connected devices, applying efficient traffic classification methods is critically important. Generally speaking, one can categorize network traffic classification techniques into three basic classes as listed below [80]:

- port-based: These techniques simply associate services/applications to registered port numbers, e.g. HTTP port, and categorize the traffic according to the used port number. Port-based techniques are among the earliest traffic classification methods. Despite the advantages of port-based techniques such as simplicity on implementation, deploying new communication methods such as tunneling and random ports assignments techniques cause serious difficulties and affect the performance and applicability of them.
- payload-based: Payload-based methods, also known DPI, closely investigate the content of the captured packet, especially the application layer-related information, in order to associate the packet to a specific service/application. In order to make a prediction, this methodology usually leverages predefined signatures or patterns for each communication protocol, and then discover these patterns to differentiate the traffic flows from each other. Payload-based classification techniques suffer from three main problems in conventional networking paradigms as listed below:

  1. They run into difficulties with encrypted traffic classification.
  2. Privacy policies may limit access to the contents of the packets.
  3. Payload methods impose heavy computational overhead on communication systems

  As a result of these difficulties, new traffic classification methods have been proposed to dispose the need for inspecting packets contents.

Table 2
A summary of deep learning models.

Class	Approach	Learning technique	Input data	Characteristics
MLP	Discriminative	Supervised	Various	• Non-linearity • Adaptive learning • MLP is highly fault tolerant
CNN	Discriminative	Supervised	2-D (image, video, etc.)	• Requiring a huge training data for visual tasks (images and videos) • Processes sequential data by internal memory
RNN	Discriminative	Supervised	Sequence data, time-series	• Suitable for NTMA use cases with time-sensitive data • Processes sequential data by internal memory
AE	Generative	Unsupervised	Various	• AE can work with unlabeled data instances • Useful for feature extraction and dimensionality reduction • The output is a reconstruction of input data
Generative models	Generative	Unsupervised	Various	• Ability to produce new data similar to existing data
LSTM	Discriminative	Supervised	Sequence data, time-series, long time dependent data	• Fine performance in applications with long time lag input data compared to its predecessors • Work with unlabeled and labeled datasets

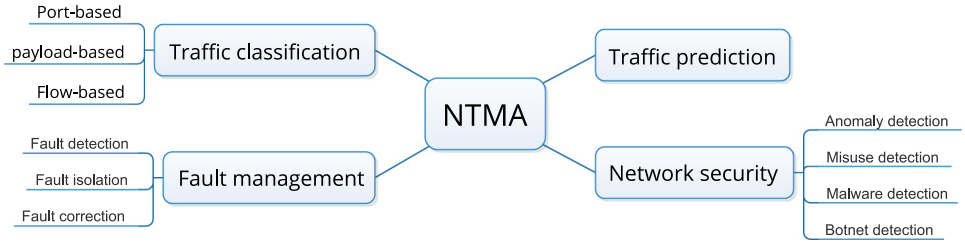


Fig. 8. NTMA sub-fields.

- Flow-based: The underlying assumption behind the flow-based methods is that traffic associated with each application/service has almost unique statistical/time series characteristics. Hence, a flow-based classifier can handle both encrypted and normal traffic. Flow-based methods generally use traditional ML models, such as decision tree, logistic regression, and Support Vector Machine (SVM) for traffic classification. Despite this fact that ML models achieve a remarkable accuracy level, they need a massive amount of fully labeled data for modeling purposes.

With the rapid increase in the number of DL models, researchers have recently investigated these models for traffic classification and consequently reported great accuracy [49]. Motivated by the DL models proliferation, we provided a comprehensive review of traffic classification works.

Due to the complexity and low accuracy of MLP networks, pure MLP implementation has rarely been employed for network traffic classification. Pure MLP suffers from the disadvantages that it needs to tune some hyper-parameters, such as the number of hidden neurons and layers and sensitivity to feature scaling. A combination of MLP with other DL methods and pure MLP has been studied in some works, including in [81–92].

Aceto et al. [81] studied DL-based models for mobile traffic classification. They reproduced several DL classifiers, e.g., MLP, LSTM, CNN, and SAE, from the traffic classification literature in order to make a comprehensive evaluation for showing the accuracy of these classifiers. Among DL-based classifiers, the best performance is related to 1D-CNN with 76.37%/85.70% accuracy and the F-measure of 75.56%/78.78% on FB-FBM and Android dataset, respectively. The authors acknowledge the fact that classical ML algorithms that employ experts- and manually-based methods for feature extraction are not appropriate for modern networks due to: (1) handheld devices' massive deployment, such as smartphones and tablets, considerably increases mobile traffic volume, (2) the massive adoption of the encrypted network protocols, e.g., Transport Layer Security (TLS), reduces the effectiveness of DPI techniques based on ML algorithms, and (3) considering the ever-increasing development of mobile applications and the changing nature of mobile traffic, implementing up-to-date and accurate traffic classifiers through classical ML algorithms is challenging.

Wang et al. [82] also developed different DL-based traffic classifiers. Motivated by the recent advances in DL-based traffic classification and the weaknesses of the available traffic classification techniques, e.g., DPI, in giving real-time application awareness for encrypted network traffic, the authors used DL-based models, i.e., MLP, SAE, and CNN, to categorize traffic in the smart home use case. They used an open dataset with 200,000 encrypted data points from 15 applications to evaluate the models. The experimental results reveal the applicability of the evaluated models for smart home networks. More specifically, the average results of Precision, Recall and F1-Score on DataNet dataset are MLP = 0.9657%, 0.9653%, and 0.9653%, SAE = 0.9883% 0.9881%, and 0.9882%, CNN = 0.9847%, 0.9842%, and 0.9843%, respectively. The authors of [83] focused on media traffic classification through DL. They applied CNN and MLP methods to classify four types of media traffic, i.e., video, audio, image, and text. According to the results, MLP shows good performance in terms of accuracy (0.9983%) and training time (0.019 s) under different scenarios.

In [84], IDS is considered by Ferreira and Shinoda since intrusion detection is a serious challenge in the context of NTMA. The authors introduced a new intrusion detection dataset and employed several traffic classification algorithms, such as MLP, J48, and Bayesian networks, to evaluate the dataset. Similarly, works in [85–92] proposed to use an MLP for traffic classification in IDS.

Despite difficulties with using pure MLP, some works use this model for traffic classification. For example, in [90], Miller et al. used MLP to categorize encrypted VPN and non-VPN network traffic. The simulation results show 92% and 93% accuracy for VPN and non-VPN traffic classifiers, respectively. Similarly, Sahay et al. deployed MLP neural

networks as a classification tool to detect misappropriation attacks in Low power and Lossy Networks (LLNs) [91]. The authors claim that the proposed method can also find the nodes affected by the attack and determine the malicious nodes. The pure MLP model has also been adopted in the context of IDS. Wang et al. used the MLP network in combination with the sequential feature selection technique in order to detect distributed denial of service (DDoS) attack [92]. They utilized these techniques to select the optimal features during the training phase. Moreover, to show the effectiveness of the proposed methodology ($\approx 98\%$ accuracy), they compared it with some papers in the literature.

As mentioned, one of the main advantages of CNNs compared to conventional neural networks is *the automatic detection* of the important features and hierarchical feature extraction. A simple CNN model proposed in [93] for the categorization of encrypted traffic. This paper is one of the first works leveraging CNNs in the context of traffic classification, in which encrypted traffic is transformed into two-dimensional images, and then the images fed into the CNN model to be classified. The authors reported an accuracy of 1D-CNN = 1%, 82%, 98%, and 86%, and 2D-CNN = 1%, 80%, 97%, and 84% for four different experiments, respectively. The main advantages of the method presented in this work over the existing traffic classifiers, such as classical ML classifiers, include (1) integrating feature extraction/selection/classification phases into an end-to-end framework; (2) categorization of the encrypted network traffic which is a challenging task for the traditional classifiers. In [94], the authors also adopted the CNN model for IP traffic classification. They converted sequences into images that fully represent the patterns of different applications, such as Facebook and Instagram. Then, the CNN model is employed to classify the images to different applications. Rezaei and Liu proposed a one-dimensional CNN-based semi-supervised approach to categorize five Google applications [95]. To reduce the need for large labeled traffic datasets, first, the model is pre-trained on a big unlabeled training test where the time series characteristics of a few samples of packets are considered as the input. The proposed method's performance is evaluated with different sampling techniques (i.e., fixed step sampling, random sampling, and incremental sampling) on three different datasets, including the QUIC dataset, Unlabeled Waikato dataset, and Ariel dataset. The proposed pre-trained method achieved higher accuracy than its non-pre-trained counterpart, with 81.50%, 81.27%, and 80.76% on the QUIC dataset for the sampling techniques. As mentioned, the authors use a 1D-CNN as a classifier because they believe the using of new applications and network encryption techniques have considerably raised the complexity of the traffic classification tasks, mainly when one uses classical ML-based methods.

In [96], a novel IDS, namely, HAST-IDS, is proposed, in which CNN and LSTM models are used to learn the low-level features of spatial information of network traffic and high-level features of temporal information, respectively. No feature engineering phase is used in the proposed system since the deep neural models automatically learn the key features. To measure the effectiveness of the system, DARPA1998 and ISCX2012 datasets have been used by the authors, where HAST-IDS outperformed its competitions in terms of training and testing time and accuracy in both datasets. For example, in the DARPA1998 dataset, the training and testing time is 58 min and 1.7 min, respectively, and accuracy on the ISCX2012 dataset is $\approx 99.5\%$. Yeo et al. [97] applied CNN to malware detection tasks in an automated fashion. The authors claim that the introduced method can detect malware that uses unpredictable port numbers and protocols. This is mainly due to the fact that the model employs 35 different features captured from the packet flow, instead of features extracted from packets such as the port numbers and protocols. Besides, conventional networks have been used as traffic classifiers in IoT networks, where traffic classification can help distinguish between traffic/behavior of heterogeneous devices and services in these networks [98]. In this work, the authors combined CNN and RNN models to achieve the best detection results, around 97% accuracy when they use all features. The proposed method shows excellent performance in terms of detection scores, even under a highly unbalanced dataset. Compared to the classical ML techniques, the proposed DL models in [98] do not need to go through the feature engineering phase thanks to the convolutional layers that extract complex features automatically from the input data.

Tong et al. [99] provided the novel traffic classification based on CNN to categorize QUIC protocol traffic. They focus on the networks that use Google's QUIC protocol since the traffic generated by such systems imposes several challenges for traffic classification tasks because this protocol decreases network traffic visibility. As a result, port- and payload-based traffic classification methods cannot be used for QUIC-based communications. To deal with this problem, CNN has been proposed, utilizing the flow- and packet-based features for further improvement. CNNs have also been adopted for malware traffic classification [100]. In this work, first, the network traffic is transformed into two-dimensional images. The convolutional network is then used to classify these images into different categories, such as Skype, FTP, and Outlook, , and the authors reported the average accuracy of 99.41%. Despite the advantages of the proposed method, the authors highlighted some limitations of their work, including (1) the size of the used dataset and classes number are fixed, while in the real-world use cases is not undoubtedly true, (2) the proposed method only utilized network traffic spatial features, while classical ML-based classification methods utilize different temporal features and show high accuracy.

For network traffic classification, RNN models are usually used with other DL models. For instance, in [98], both the RNN and CNN models are used for traffic classification. Different DL models are implemented in this work, where a particular combination of CNN/RNN achieved the highest degree of accuracy. Radford et al. proposed a creative method in [101] for network anomaly detection through RNN. They converted network flow into sequences of words that form sentences, then these sentences are considered as the language model of a specific network. RNN is used to identify network activities that are malicious with respect to the model.

Auto-encoders are mainly used as an unsupervised technique to do automatic feature extraction and selection. More specifically, the output of the encoding part of an AE network can be used as a high-level set of discriminative features for a classification problem. Auto-encoders models have also been applied to classification problems, e.g., in [49] Lotfollahi et al. adopted an Stacked Autoencoders (SAE) model, called Deep Packet, for encrypted traffic classification. The SAE stacks several AEs to form a deep structure to obtain a better performance. The authors used the UNB ISCX VPN-nonVPN dataset to assess the performance of the introduced method. Deep Packet outperformed all of the introduced and compared classification methods on the used dataset, including two classical ML algorithms, i.e., k-NN and C4.5, an accuracy of 0.98% is compared to 0.94% and 0.90%, respectively. Moreover, given the increasing interactions between different components on the Internet and, consequently, the network's considerable complexity and diversity, DL algorithms are necessary to perform traffic classification tasks. In [102], Zhao et al. deployed AE to extract and aggregate features from traffic data. Then, they used the n-gram embedding strategy and k-means clustering to classify unknown traffic, i.e., network traffic generated by previously unknown applications or services. The authors have targeted network flow classification in [103]. They proposed an improved SAE, in which several basic Bayesian auto-encoders are stacked to understand the complex relations between the multi-source network flows. Moreover, the proposed SAE is trained through the back-propagation learning algorithm and in a supervised learning manner in order to learn the complex relations between the network flows. The simulation results show the improved SAE outperforms its ancestor in terms of accuracy (83.2 percent accuracy versus 82.9 percent). Last but not least, in [104] a comparison between the classical machine learning classification method and the DL method, i.e., SAE, has been made. The experiments revealed that

DL model provides better accuracy (with 99.20%) than the classical ML model (with 95.22%). Furthermore, the authors claimed that in highly distributed networks, such as IoT systems, the traditional techniques such as classical ML techniques for NTMA purposes (e.g., attack detection) have less scalability. As a result, they proposed edge-based deep learning to deal with modern communication systems' distributed and complex nature. The vast amount of data generated by IoT edge devices allow DL models to learn more useful than classical ML models.

In the context of network traffic classification, deep generative models can be used to deal with the imbalanced dataset problem. An imbalanced dataset refers to the situation in which the number of instances available for different data classes is considerably different. In such situations, predicting the classes with few instances is usually challenging for classical ML models. To alleviate this problem, oversampling and undersampling are two frequent and easy techniques. In the former, oversampling can be realized through duplicating instances of minor label classes, whereas by deleting some instances from major classes, one can implement an undersampling technique. In [105], a deep generative model, namely Auxiliary Classifier GANs (AC-GAN), is proposed to address the problem of imbalanced classes of network data. More precisely, Generative Adversarial Network (GAN) has been deployed for the generation of synthesized data instances to create a balance between the minor and the major label classes. In [106], Alom et al. used Deep Belief Neural Network (DBNN), a well known generative model, for intrusion detection. Furthermore, they compared the proposed method with some existing methods, such as SVM and DBNN-SBM. The proposed methods outperformed all these methods in terms of classification accuracy by achieving $\approx 97\%$ accuracy. The authors announced that their method is not only able to detect threats, but also categorize them in five classes with the accuracy of detection. Another advantage of the provided DL model is that it can detect any unknown attack that has not been considered in the training dataset. Iliyasu et al. introduced a semi-supervised learning technique by Deep Convolutional Generative Adversarial Network (DCGAN) for the classification of encrypted network traffic [107]. The main idea behind this method is to use DCGAN for instance generation, as well as utilizing unlabeled traffic data to increase the accuracy of the learner, even when a small number of labeled data is available for training purposes. The authors deployed QUIC and ISCX VPN-NonVPN datasets to demonstrate the accuracy of their model, where the model delivered 89% and 78% classification accuracy on both QUIC and ISCX VPN-NonVPN datasets, respectively. As another positive point, the proposed deep method can alleviate the problems connected with extensive dataset collecting and labeling, which are problematic for both classical ML and DL models.

A summary of the papers reviewed in this section is provided in Table 3.

5.2. DL for network traffic prediction

Network Traffic Prediction (NTP) refers to the understanding of the future status of links in a network. Network traffic prediction and modeling are two key metrics to measure telecommunications systems' performance as they attract much attention [108]. In the context of cellular networks, making an accurate prediction on the dynamic of cellular network traffic is a key step towards improving network performance. Considering the rapid evolution towards deployment of the 5G cellular networks, the telecommunication systems and networks are expected to be more intelligent and self-organized [109]. A Self-organizing Network (SON) had to adapt itself to dynamic patterns of usage and perform preemptive actions for planning, configuration, management, and optimization of the network. Towards this end, prediction and understanding of the future of dynamicity of the mobile traffic is crucially important to support smart and automated management features [110].

From the point of view of an IoT service provider, traffic prediction is highly valuable since it can provide information on the probability distribution of IoT devices connectivity [4,13]. The information can be used to prepare the software and hardware infrastructures needed to minimize the risk of interruption of extremely significant services and related devices. Moreover, it is highly useful to know in advance the status of IoT devices connectivity in order to decline the impact of possible connectivity congestion in a network.

In recent years, it is becoming more and more apparent that NTP is a challenging task. The volume of mobile data traffic has experienced an enormous increase in the last few decades [111]. In addition, technological advances in the field of communication systems and networking lead to a proliferation of the number of devices connecting to the cellular network, as well as emerging social networks such as Instagram and Facebook have further added to the network traffic volume [112]. For example, Xu et al. [110] demonstrate that a considerable portion of mobile traffic is unpredictably random. They analyze the traffic patterns of more than 9000 Base Stations (BSs) in a metropolitan area. In [113], some challenges for ML in network traffic prediction, such as data acquisition, class imbalance, concept drift, and big data setting, have been listed. In [114], significant spatial and temporal variations in cellular network traffic are referred to as a severe challenge to accurate cellular traffic prediction.

Despite all the difficulties mentioned above, various methods for NTP have been proposed in the literature. Generally, one can categorize them into two main groups, including classic prediction methods (e.g., ARMA) and ML-based methods. The most commonly adopted linear methods are ARIMA/SARIMA models and HoltWinters algorithm [115–119]. Whereas, the most commonly used non-linear methods are traditional and deep neural networks [120]. The performance of different linear methods such as ARMA, ARIMA, and HoltWinters and non-linear methods such as traditional neural networks were investigated [121,122]. In the majority of cases, the non-linear methods have performed better than linear methods. Broadly speaking, the best prediction technique can be selected based on considering some measurement factors, such as computational cost, lower mean error, and characteristics of the traffic matrix to name a few. One of the serious limitations of linear methods (e.g., ARIMA) is their low robustness to the sudden changes of the time-series. This is due to the fact that the model tends to over recreate the average of the previously observed instances [42]. Adding new services or unforeseen changes in the current service settings (e.g., the running of new bandwidth-hungry use cases) presents significant challenges to these methods. Moreover, these methods provide poor performance with non-homogeneous time-series, where the input and the prediction are not within the same set of data points.

DL has been used successfully in many use cases, such as visual recognition and Spatio-temporal forecasting problems [123], as well as is considered as one of the most cutting edge achievements in AI. Different types of DL models have been applied in the context of NTP, e.g. CNNs and RNN in cellular networks to capture spatial and temporal properties [114]. In the following, the state-of-the-art DL models for traffic prediction are reviewed.

Azari et al. [116] provided a comparative evaluation of LSTM and ARIMA. They studied the effect of different parameters on the models on the effectiveness of the predictions. Their simulation results prove the superiority of LSTM over ARIMA, particularly when the training time series is long enough. Nevertheless, in some scenarios, ARIMA gives performance near the optimal with a lower level of complexity. In a similar way, in [124], authors made a comparison between three well-known traffic prediction models, i.e., RNN, ARIMA, and Wavelet Transform (DWT). They referred to this fact that NTP is very helpful for many applications, such as congestion control, anomaly detection, and bandwidth allocation. Andreoletti et al. [125] proposed a novel method for traffic forecasting through Convolutional Recurrent Neural Network (DCRNN). They employed DCRNN to predict the amount of expected traffic and to forecast network congestion. In addition, the authors compared the proposed method with other famous methods,

Table 3
A summary of works on network traffic classification.

Reference	Category	DL model	Key contribution
Aceto et al. [81]	Traffic classification	MLP, CNN, LSTM, SAE	Comprehensive evaluations of different DL models
Wang et al. [82]	SDN traffic classification	MLP, SAE, CNN	Application-aware SDN-home gateway (HGW) framework is introduced for smart home networks
Lyu et al. [83]	Media traffic classification (e.g. video and audio)	MLP, CNN	Precise classification of different types of media traffic
Ferreira et al. [84]	Intrusion detection	MLP, Bayesian networks, Decision Tables, IBK, Naïve Bayes,J48	Investigation about the creation of a IDS dataset
Pwint et al. [85]	Anomaly detection	MLP, Decision tree, Naïve Bayes, Random forest, Logistic Regression	Introduces multi-class network attack anomaly detection system by Apache Spark's framework.
Salek et al. [86]	Intrusion detection	MLP, RBF, PNN	Evaluates different DL and ML models for intrusion detection
Salih et al. [87]	Intrusion detection	MLP, Naïve Bays, KNN	Finds that a high level of attacks classification accuracy can be achieved by combining best different features selection.
Sreekesh et al. [88]	Intrusion detection	MLP+ Reinforcement Learning (RL)	Introduces two tier architecture in order to increase the system security.
Efferen et al. [89]	Anomaly detection	MLP, J48	Shows the importance of right feature selection.
Miller et al. [90]	Encrypted vpn traffic classification	MLP	Proposes a framework based on a MLP model to classify VPN and non-VPN traffic
Sahay et al. [91]	Attacks detection in IoT	MLP	Introduces a mechanism to detect Misappropriation attacks in the IoT LLNs.
Wang et al. [92]	Attack detection	MLP	Provides an interactive approach to combine feature selection with MLP model in order to detect DDoS attack.
Wang et al. [93]	Encrypted traffic classification	CNN	Uses an end-to-end deep learning approach to conduct encrypted traffic classification.
Chen et al. [94]	IP traffic classification	CNN	Employs a compact nonparametric kernel embedding based technique to transform traffic flow sequences into images, and then categorize these images.
Rezaei et al. [95]	QUIC protocol classification	CNN	Introduces a semi-supervised method that uses large quantities of unlabeled data and just a few labeled instances.
Wang et al. [96]	Intrusion detection	CNN+LSTM	Proposes a system learns spatial–temporal features of network traffic flow.
Yeo et al. [97]	Malware detection	CNN, MLP, RF, SVM	Introduces a more robust and accurate malware detection method through features extracted from packet flow.
Lopez et al. [98]	IoT traffic classification	RNN+CNN	One of the first works that uses an RNN combined with CNN for traffic classification task.
Radford et al. [101]	Anomaly detection	LSTM RNN	Proves that LSTM RNN can detect patterns of malicious traffic without the help of labeled data instances and without insight into each node's internal state.
Lotfollahi et al. [49]	Encrypted traffic classification	SAE+CNN	It is able to do both traffic characterization and application identification.
Zhao et al. [102]	Features extraction unknown traffic identification	AE+ KNN+ n-gram embeddings	Presents a method for identification unknown network traffic to address the issue of zero-day applications.
Li et al. [103]	Traffic flow classification	Bayesian SAE	Uses Bayesian probability in order to achieve a posteriori distribution of model parameters.
Abeshu et al. [104]	Attack detection for IoT applications	SAE, classical ML	Provides a novel DL approach for attack detection in fog-to-things computing.
Vu et al. [105]	Traffic classification	GAN	Uses GAN to address imbalanced dataset problem in traffic classification tasks.
Alom et al. [106]	Intrusion detection	DBNN	First comprehensive method for intrusion detection using DL model.
Iliyasu et al. [107]	Encrypted traffic classification	DCGAN	Utilizes DCGAN to generate data instances and unlabeled data instances to improve the classification accuracy.
Tong et al. [99]	QUIC traffic classification	CNN	Leverages the convolutional network to classify encrypted traffic by QUIC protocol and achieve good performance than the available methods

(*continued on next page*)

Table 3 (*continued*).

Reference	Category	DL model	Key contribution
Wang et al. [100]	Traffic classification	CNN	One of the earliest papers that use CNN for traffic classification by transforming network traffic into images and applying the CNN network.

such as LSTM and Fully-Connected Neural Networks. For instance, the Root Mean Squared Error (RMSE), Mean Absolute Percentage Error (MAPE), and the Mean Absolute Error (MAE) for DCRNN are equal to 497.1 (Mb/s), 43.2%, and 92.5 (Mb/s), while these figures for LSTM are 525.21, 210.34%, and 142.43, respectively. The result of the simulations reveals that the DCRNN outperform the referenced counterparts with regard to prediction ability and network congestion prediction. The authors provided the remarkable insight that it is not straightforward to use classical ML algorithms for NTP. This is mainly due to the fact that classical ML algorithms are adopted to be used for data in the Euclidean space, while the data in communication systems and networks are usually graph-structured data. Hence, they deployed DCRNN as a graph-based DL algorithm for the NTC task.

One of the DL models, gaining the reputation to learn long-range dependencies time series is LSTM. Today, studies and applications of LSTM for time series forecasting in the context of communication systems and networks are proliferating. For example, in [42], the authors used LSTM to model and predict Spatio-temporal data in cellular networks. This paper directly challenged Support Vector Regression (SVR) and ARIMA as two widespread methods for time-series prediction. The paper reveals that ARIMA is not a useful technique for NTP due to its bias to concentrate on the historical data's mean values, making it powerless to catch the fast variational underlying network traffic data. Regarding the SVR, it refers to this fact that one has to determine the different model's parameters. Simultaneously, there is no structured method for selecting the most suitable values of the parameters.

Fen et al. addressed the cellular traffic prediction task through a deep traffic prediction, called DeepTP [126]. DeepTP comprises two primary components: a feature extractor to model spatial dependencies of cellular traffic, encode the external related information, and a sequential module for modeling important temporal changes. The authors reported that DeepTP outperforms the latest methods of traffic forecasting by more than 12.31%. The authors used DL to solve the cellular traffic prediction problem as they argue that the performance of available techniques is still low because of the following difficulties: (1) complex temporal variations in the network traffic, e.g., traffic burstiness, (2) dependencies to different impact components such as daytime and day of the week, and (3) spatial dependencies caused by user mobility. Motivated by the importance of traffic prediction to telecommunication providers in order to manage their resources in an efficient and futuristic manner, Dalgkitsis et al. in [127] also introduced an LSTM-based approach for traffic prediction in cellular networks. They compared the proposed technique with different prediction methods, such as RBF, ARIMA, and SARIMAX. The proposed method shows more outstanding performance than other methods in terms of prediction error. The MSE achieved by LSTM = 1.685, SARIMAX = 11.26, ARIMA = 6.53, and RBF = 3.24.

Fang et al. investigated one of the big problems in cellular networks, i.e., per-cell demand forecasting [128]. The authors referred to the fact that the nonuniform spatial distribution of cells poses a serious challenge towards the modeling of spatial relevancy, mostly when one employs the neural networks that accept the grid-based input data. The authors used the dependency graph based on convolutional networks and LSTM to model the spatial dependence among cells to address this problem. The convolutional networks are responsible for modeling traffic data's spatial features, whereas the LSTM can model temporal aspects. Zhang and Patras focused on traffic prediction in mobile networks through DL [129]. This paper proposed a Spatio-temporal neural network architecture based on CNN and LSTM models to forecast cellular traffic in dense urban areas, where there is extreme spatio-temporal variability in mobile traffic because of the mobility of users. One of the main advantages of the proposed architecture is that it only needs a small number of ground truth samples. The simulation results demonstrate the proposed method's provides better performance than its counterpart in terms of prediction error in different prediction durations or steps. Under 1-step setting, For example, the proposed method achieved a Normalized Root Mean Square Error (NRMSE) of 0.19, while this figure for ARIMA = 0.20, MLP = 0.23, SVM = 0.39, and AE + LSTM = 0.24.

One of the less-investigated DL models in the field of NTP is SAE. One of the first works of the traffic prediction through SAE is conducted by Oliveira et al. in [130]. They employed and compared two DL models, i.e., MLP and SAE, for the Internet traffic prediction. To evaluate their models, they use real traffic data that indicate that the proposed models are able to cope with complex traffic flow prediction tasks with reasonable accuracy and effectiveness. Another advantage of using the SAE is the unsupervised training nature of this DL model. Besides, compared to the classical ML algorithms, DL algorithms support adding considerable complexity to the prediction model due to several data representation layers. In [42], a new deep technique based on auto-encoder is proposed for spatial modeling. Also, in this paper, LSTM is used to model temporal information. The auto-encoder model comprises of a Global Stacked AutoEncoder (GSAE) and several Local SAEs. The main idea behind using multiple SAEs is that they can provide better representations of input data and decline the model size. Bega et al. proposed DeepCog, as a DL-based data analytics tool for traffic forecasting in network slicing [131]. The main objective of DeepCog is to predict the respective resource demands for each slice for resource allocation purposes. DeepCog takes benefit from a DL architecture specifically established to predict a network slice's future needed capacity. The architecture comprises two main modules, including encoder and decoder. The encoder accepts the cellular traffic data as input and then maps spatial/temporal features of data onto a low-dimensional space. Then, the decoder processes this low-dimensional data representation to produce the ultimate slice capacity prediction. Furthermore, similar work has been conducted by the authors in [132], where they proposed AZTEC, a framework for automatic allocation of capacity to different network slices. The proposed framework utilizes DL architectures (CNNs) and a traditional numerical optimization algorithm to provide the best performance, i.e., minimizing management's costs.

DL models demonstrate remarkable ability to capture the complex and non-linear dependence hidden in wireless communications and raised as the biggest competitors to classical linear models in traffic forecasting. Among DL models, CNN is one of the most powerful ones that has been successfully used in a wide range of applications, such as computer vision and NLP, and traffic prediction is no exception. Zhang et al. [133] introduced a novel approach for citywide traffic prediction through CNN. More specifically, they employed CNN for modeling the spatial and temporal dependence of traffic in different network cells. While many studies have been conducted to examine the dynamic characteristics of mobile network traffic (e.g., ARIMA and ML), the pattern of mobile network traffic is too complicated because of the different factors (e.g., UEs mobility and diversity). As a result, it soon becomes evident that these linear models did not work in such a complex network, and adopting novel models based on DL is necessary. Work in [134] targeted network traffic forecasting in data centers by gated recurrent unit (GRU) model and interactive temporal recurrent convolution network (ITRCN) model. CNN is a part of the ITRCN model that is responsible for learning network traffic in the form of images

to discover the network-wide services' correlations. To evaluate the performance of their method, authors used datasets from Yahoo and the results of experiments demonstrate the superiority of the proposed method over GRU and CNN by 13.0% and 14.3% in RMSE, respectively. Motivated by the potential applications of traffic prediction in network planning and routing configurations, and consequently, QoS for users, Nie et al. developed a network traffic prediction approach based on DBNN and spatiotemporal compressive sensing technique [135]. They first used a discrete wavelet transform in order to extract the low-pass component of network traffic, and then DBNN is adopted as a prediction model to categorize network traffic.

The authors in [136] proposed an attention-based convolutional network to forecast of wireless network traffic, called LA-ResNet. Their method can involve both the temporal and spatial features of traffic in the prediction process. To extract spatial characteristics of the traffic, they use a residual network, where RNN is deployed to capture temporal features. The RMSE calculated for the proposed method and its well-known counterparts, including 3DCNN (5.02), ARIMA (7.98), LSTM (6.12), GRU (6.48), and CNN + RNN (11.03), showed the superiority of the proposed method (4.5).

Wan et al. [137] conducted a detailed investigation into cellular network traffic in large-scale deployments. In their paper, the authors first provide a useful insight into cellular traffic in large cities, temporal/spatial dynamics of cellular traffic in such environments, and the source causes of these dynamics. They also deployed a graph-based DL method for cellular traffic forecasting. The simulation result reveals the superiority of the proposed method over time-series based techniques. The paper's novelty is that the authors modeled the spatial/temporal features of cellular traffic in an urban area employing a directed graph. Then, they used a graph-based DL model that can learn from the modeled graph.

A summary of the papers reviewed in this section is provided in Table 4.

5.3. DL for fault management

Fault management refers to the group of tasks to detect, isolate, and then correct abnormal situations of a network. It also includes any operations needed to determine the source of an abnormal situation, also called failure. Failure happens when a network cannot successfully offer a service, where a fault is the root cause of a failure. In traditional communication systems and networks, link and node failures are the most common type of failures, and faults are mostly related to software bugs (e.g., failed web service) or hardware-related crashes (e.g., routers) [138].

Fault management plays a crucial role in today's network management procedure. The recent rapid growth of interests in IoT and the proliferation of mobile devices has further strengthened the importance of fault management. Faults in communication systems and networks are no exception and tend to happen more frequently. Along with common network faults, recently trendy phenomena, such as IoT and the Internet of Everything (IoE), have to cope with faults related to unreliable hardware, limited battery life, connectivity failure, harsh environmental condition, etc. Hence, in order to guarantee QoS and high performance in communication systems and networks, fault detection and performing immediate and effective actions to heal and recover the systems from failure are crucially important. A set of functions and a cyclic of the process are introduced specifically for this purpose that categorized under the umbrella of fault management systems [9].

In the context of cellular networks, high demands, and the ever-increasing dependency of people on these networks are the primary motivations for designing better fault management systems. Nowadays, cellular networks have become an essential part of the communication infrastructures, where users can use their mobile device anytime and almost everywhere. This makes fault management as one of the central aspects of network management in the context of cellular networks [139].

Despite the importance of fault management, there are some difficulties in realizing an effective fault management system. For example, IoT sensors and some IoT devices usually have limited resources of power, like a battery. It is expected that these smart physical objects operate autonomously on their environment for some periods of time, ranging from days to years. Moreover, these objects may not be easily accessible to change their batteries since IoT devices may be deployed in specific locations, such as forests and volcanic areas. Because of these reasons, faults may happen more frequently and unexpectedly in IoT networks than conventional networks. Besides, although the distributed nature of communication systems and networks brings scalability and resiliency to failures, it makes it challenging to perform fault management for such complex systems both due to the handling a substantial number of devices and also due to the different vendor-specific characteristics [140,141].

Fault management can be considered as a cyclic process, i.e., operates on a continuous cycle and actively seeks for abnormal conditions on a network. Although every fault management system may have different steps, the general fault management cycle includes fault detection, localization (or fault diagnosis), and mitigation (or fault resolving) steps. First, a fault management system examines the network and discovers one or more failures that affect the network's performance. As instances of fault, one may refer to filled switch capacity, disabled link, and disabled switch [142]. The next step in the fault management cycle is to localize the source of fault (s). This step calls for determining the physical location of the fault (s) on the network and pinpointing the reason for the fault (s). Furthermore, finally, fault mitigation attempts to repair or fix the network fault (is). This step may performed in an automatic or manual way.

Fault prediction is another important concept in the context of fault management and aims to prevent network failures by forecasting them and establishing resolving procedures in order to minimize the negative effects of the failures. DL-based approaches have been introduced to deal with these challenges and improve functionality in the steps mentioned above. Addressing complex problems is one of the key advantages of ML, especially deep learning [143]. Regarding the ever-increasing amount of traffic in communication systems and networks, using DL models to analyze such a massive amount of traffic is a promising technique to produce helpful insights for fault management. In the following, the DL models that have been proposed for these significant challenges for fault management are reviewed.

Huang et al. in [144] first reviewed fault detection mechanisms in IoT systems and then proposed a fault-detection architecture for Self-Driving Network (SelfDN)-enabled IoT. Towards this end, they also introduced an algorithm, namely, Gaussian Bernoulli restricted Boltzmann machines auto-encoder (GBRBM-DAE) in order to convert the fault-detection task into a classification task. The result of experiments reveals that the provided algorithm shows better detection accuracy (82.95%) than the other commonly used ML algorithms, such as SVM (77.4%), linear regression (64.9%), quadratic discriminant analysis (64.8%), and linear discriminant analysis (68.9%). The authors discussed this exciting point that IoT devices produce a massive amount of data at the edge of the network, which causes a high computational load for online processing in edge servers. As a result, using classification techniques based on the traditional ML models is ineffective or almost incapable. DL techniques can process a massive amount of data with a large number of features without the need for a hand-engineered data preprocessing phase and engineering techniques. Mulvey et al. in [145] targeted the sleeping cell problem in the cellular networks through DL techniques. More specifically, they adopted RNN to diagnose cell radio performance degradation and complete cell outages in a cellular network. The proposed method achieves greater sensitivity than traditional ML-based techniques, e.g., SVM, while reducing the demand for some preprocessing phases such as dimensionality reduction. Masood et al. in [146] provided an auto-encoder-based framework for self-governed detection of sleeping cells in mobile networks. Sleeping

Table 4
A summary of works on network traffic prediction.

Reference	Category	DL model	Key contribution
Assem et al. [114]	Capture spatial and temporal properties	CNN, RNN	For the first time, extracted urban patterns considered in network demand prediction.
Azari et al. [116]	Cellular traffic prediction	LSTM, ARIMA	Provides a comparative evaluation of LSTM and ARIMA.
Madan et al. [124]	Traffic prediction	RNN, ARIMA and DWT	Proposes and compares three methods for the network traffic forecasting.
Andreoletti et al. in [125]	Traffic prediction	DCRNN	Introduces a method that is able to learn a representation of a network that considers both the properties, e.g. the load on links and the structure of the network, e.g. topology.
Wang et al. [42]	Cellular spatiotemporal prediction	LSTM+AE	Deploys an AE to model spatial correlations and an LSTM to model temporal correlation.
Feng et al. [126]	Cellular traffic prediction	LSTM	Spatial and temporal dependencies extraction by separate modules.
Dalgkitsis et al. [127]	Cellular traffic prediction	LSTM	Demanding part of the prediction task can be offloaded to a server side.
Oliveira et al. [130]	the Internet traffic prediction	SAE, MLP	Uses two types of artificial neural network models for the Internet traffic prediction.
Zhang et al. [133]	Citywide traffic prediction	CNN	Applying CNN to model spatial and temporal dependence of traffic in different cells.
Cao et al. [134]	Traffic prediction in data centers.	GRU, ITRCN	First work to use the image-based method for network traffic forecasting in large-scale data centers.
Nie et al. [135]	Traffic prediction in wireless mesh network	DBNN	First work to use DBNN for network traffic prediction in wireless mesh networks.
Wan et al. [137]	Cellular traffic prediction	Graph-based DL	Delivers insight into cellular traffic characteristics and proposes a graph-based DL model for traffic prediction in cellular networks.
Bega et al. [131]	Network slice capacity prediction	AE	Designs a novel architecture based on DL algorithms to forecast the future network slice capacity.
Bega et al. [132]	Network slice capacity prediction	CNN	Establishes a framework based on DL architectures and a classical optimization algorithm to automatically predict a network slice capacity in advance.
Fang et al. [128]	Mobile demand prediction	CNN+LSTM	Combines the graph-based CNNs and LSTM to predict the per-cell demand in cellular networks.
Zhang and Patras [129]	Cellular traffic prediction	CNN+LSTM	It provides a novel method that employs DL algorithms for mobile traffic forecasting by modeling Spatio-temporal features of traffic.
Li et al. [136]	Wireless network traffic prediction	CNN+residual+LSTM	Proposes a DL method based on an attention approach to consider Spatio-temporal features of network traffic data for prediction.

cell is a severe problem in mobile networks as it does not produce any alarm when a breakdown occurs in the hardware/software of BSs. The authors reported that their approach shows higher detection accuracy than one-class SVM, with 99% accuracy compared to 94% accuracy provided by SVM.

As mentioned, fault localization is the second step in the fault management cycle. Dusia et al. in [147] provide a comprehensive survey on fault localization in computer networks. They reviewed the latest advances in fault localization techniques for communication networks as well as they discussed the fault localization in complex communication systems. Note, the main focus of the paper is not on DL-based approaches, whereas three major categories of fault localization methods, i.e., AI-based methods, model traversing and graph-theoretic methods, have been discussed in this study. Gupta et al. in [148] proposed a hybrid framework, namely, HYPER-VINES, using classical ML and DL algorithms with a mix of supervised and unsupervised learning for detection and localization faults in Network Function Virtualization (NFV). The proposed framework is able to handle both fault detection and localization with a reasonable level of accuracy (>95%). The authors in [149] use RNN in order to perform fault detection and localization in distributed systems (DS). To be more specific, they employed a two-dimensional CNN model in the form of a denoising AE along with RNN to simultaneously perform fault detection and diagnosis for a distributed system. The main reasons behind proposing DNNs for this problem consist in DNN models' ability to handle such systems' high dimensionality and uncertainty.

The last step in the fault management cycle is fault mitigation. Khunteta et al. in [150] adopted a method based on DL for link failure mitigation in 5G networks. They used RNN models in order to continuously track Reference Signals Received Power (RSRP) and Reference Signal Received Quality (RSRQ) as signal conditions, consequently, predict and mitigate link failure events. The simulation results show that the proposed method can mitigate the failures with user equipment (UE) autonomous decisions in link failure events. Compared with statistical models such as Hidden Markov Model (HMM) and N-gram model, DL models (e.g., LSTM) show better performance for predicting future failures due to their better data representation ability utilizing various features effectively. The proposed LSTM provided an MSE of 0.01 on the training data and 0.013 on the testing data. Ding et al. in [151] developed an LSTM-based method for detecting and mitigating faults, such as spikes and off-sets in cyber–physical systems. Moreover, the authors proposed an online fault mitigation approach. The mitigation is performed by replacing the detected faults with the forecasted values. The authors point out that DL-based techniques have exceeded conventional techniques in terms of performance as the size of data and complexity rise.

In the context of fault management, some studies consider fault management as a whole. They usually try to deal with fault management by the implementation of SON. For example, Mismar and Evans in [152] introduced a deep Q-learning algorithm for SON automate fault management. They used exploration and exploitation concepts in order to improve the downlink the interference and noise ratio (SINR). Their simulation reveals that the proposed algorithm is able to promote the performance of the cellular network in terms of downlink SINR and downlink throughput. In comparison with the classical ML techniques, supervised algorithms, the proposed deep RL-based approach in that paper offers two significant advantages, (1) it is not dependent on human supervision, and (2) it does not need data for training. In

addition, network self-healing is a novel concept in the area of network fault management. Self-healing is a subset of SON, and it refers to resolving issues in cellular networks such as cell outages, without the need for human interventions [153]. A combination of self-healing and cutting-edge DL algorithms can be used to forecast failures in advance. Such insight is beneficial due to the fact that it allows us to perform preventive operations and consequently lower unexpected costs and maintain the acceptable level of QoS.

System failure forecasting in mission-critical IT environments has been studied in [154] by Zhang et al. The authors criticized the current system management mechanisms for being labor-intensive, mainly when they utilize logs records. Moreover, they recognized that automated mechanisms based on text mining methods result in a high-dimensional feature space. To deal with these challenges, they proposed an automatic mechanism capable of parsing logs record in order to forecast system failures in IT systems. To lower the feature space's dimensionality, they first cluster the logs and then consider each cluster as a word. Besides, as the number of labeled samples is usually rare in such systems, they adopted an LSTM model to alleviate this issue. The LSTM can notably capture the long-term dependency over sequences.

A summary of the works reviewed in this section is provided in Table 5.

5.4. DL for network security

Communication systems and networks have ever-increasing impacts on people's lives, making cybersecurity a significant research area. According to [155], cybersecurity refers to the sets of policies, approaches, technologies, and processes that closely collaborate to guard computer systems, networks, programs, and data from attack, unauthorized access, and malicious changes. Cyber-defense tools mostly include firewalls, anti-virus software, and IDS. These tools are aimed at protecting communication systems and networks from internal and external threats. IDS is one of the underlying mechanisms for preventing attacks and detecting security breaches in networks.

During the past decades, there have been considerable efforts in academia and industry to enable communication systems and networks in order to respond to rapidly increasing demands [156,157]. New International Telecommunications Union (ITU) statistics show that at the end of 2019, the number of Internet users has reached 53.6% of the global population, or 4.1 billion people [158]. At the same time, according to the report in [159], cyberattacks are increasing in size, sophistication, and cost. For example, it is expected that cyber-attacks will lose about 6$ trillion annually by 2021 in the world. Thus, it is crucially important to strengthen the security of communication systems and networks against cyber threats, particularly due to the fact that people become increasingly dependent on wireless networks, such as cellular networks and WiFi for everyday life activities (e.g., online shopping, online banking, and the Internet-based business).

As mentioned, IDS are underlying mechanisms for providing security of communication systems and networks. An IDS is a hardware tool or software program that monitors a network/systems for malicious activities, attacks, violations of the security policies, etc. Many different types of IDS can be used to aid the security of networks. However, based on intrusive behaviors, network-based intrusion detection systems (NIDS) and host-based intrusion detection systems (HIDS) are the most widespread ones. Network IDS either are hardware- or software-based systems, which can be placed on different network mediums such as Fiber Distributed Data Interface (FDDI) and switches in order to inspect and analyze network traffic to protect a system from attacks and possible threats. One can classify network IDS works into two major categories, namely, misuse and anomaly detection. Misuse detection methods, also called signature-based, allow detecting previously known attacks through matching the signatures of these attacks with the analyzed data [160]. In contrast, anomaly detection methods aim to distinguish abnormal traffic patterns from normal ones by monitoring

Fig. 9. Different types of intrusion detection systems (IDSs).

the network data. On the other hand, host IDS refers to the systems which use network behavior through the different log files on the local host computer to identify attacks. Host IDS is more focused on detecting internal attacks, e.g., file permission changes. Malware and botnet detection are the two most prevalent categories of host IDS [9]. Malware detection is a technique that its objective is to protect the system by detecting malicious behavior. A botnet is a special type of malware that consists of a large number of bots and may use for malicious activities such as DDoS attacks. A graphical description of IDS techniques is provided in Fig. 9.

When it comes to applying the deep learning for security in communication systems and networks, we have realized that a considerable number of studies in the literature have focused on the application of DL for intrusion and anomaly detection. In the sequel, we present a literature review of DL-based techniques for cybersecurity applications, and we classify the papers into two main categories, namely, network IDS (misuse detection and anomaly detection) and host IDS (malware detection and botnet detection).

The use of DL seems to be widespread for network security, especially for network IDS. For example, Papamartzivanos et al. in [161] proposed a self-adaptive and autonomous misuse detection system based on DL. In this work, AE and sparse AE is used as a part of the proposed system, in which the authors combined self-taught learning and MAPE-K (Monitor-Analyze-Plan-Execute over a shared Knowledge) frameworks to provide a scalable, self-adaptive and autonomous misuse IDS. The proposed method achieved an accuracy of 77.99%, compared to 59.71% accuracy acquired by the static technique. Given the massive scale of modern communication systems and networks and the complexity of big traffic data, the IDS task surpasses the limits of traditional techniques capabilities, such as classical ML. Jiang et al. provided a virtual MAC spoofing detection technique by means of a convolutional network for the environments that use virtualization technologies [162]. In this work, CNN has been utilized to derive physical features from channel state information (gathered from packet transmissions) order to catch virtual MAC spoofing attacks.

Naseer et al. in [163] investigated the applicability of DL models for anomaly detection systems. Towards this end, they proposed different DL-based anomaly detection methods, including CNN, AE, and RNN, as well as classical ML models such as the nearest neighbor, decision-tree, random-forest, and SVM. They used NSLKDD [164] dataset for evaluation of the proposed methods. The simulation results reveal the suitability of the DL-based anomaly method for real-world applications. The authors stated that the paper's main objective is to examine DL models' suitability for anomaly detection purposes compared to the shallow models. DL models give better performance regarding different classification metrics (e.g., accuracy and precision) than shallow models. However, DL models spend more time on training and test purposes than shallow models. Similarly, Malaiya et al. in [165] examined the anomaly detection techniques based on DL. They implement different DL models, including Fully Connected Networks (FCNs), Variational AutoEncoder (VAE), and Sequence-to-Sequence (Seq2Seq) for this purpose. Moreover, the authors referred to this fact that non-linearity in network traffic data is one of the main reasons that challenge the classical ML techniques (e.g., SVM) for the anomaly detection task. Anomaly detection in cloud data-center networks is targeted by the authors in [166]. They employed the Gray Wolf Optimization (GWO) algorithm and CNN for anomaly detection. The authors claimed that their method is suitable for analyzing the network log big data for real-time anomaly detection. DARPA'98, KDD'99, and synthetic datasets are used to evaluate the efficacy of the introduced approach, which

Table 5
A summary of works on fault management.

Reference	Category	DL model	Key contribution
Huang et al. [144]	Fault detection in IoT networks	GBRBM+DAE	Surveys fault detection approaches in IoT networks and then introduce a fault-detection DL-based architecture for SelfDN-enabled IoT.
Mulvey et al. [145]	Cell coverage degradation detection	RNN	Investigates the using RNN to create a more sensitive fault detector than conventional methods, such as SVM.
Masood et al. [146]	Detection of sleeping cells	AE	Minimizes the effect of cell outage through DL-based framework which deploys Minimization of Drive Tests (MDT) functionality.
Gupta et al. [148]	Faults detection and localization in NFV	Classical ML and sparse autoencoders	Uses of shallow and deep models to achieve high degree of accuracy in detection and localization.
Qi et al. [149]	Fault detection and diagnosis in DS	RNN+CNN+AE	Proposes an automated fault detection and diagnosis method for general distributed systems.
Khunteta et al. [150]	Link failure mitigation in 5G	RNN	Uses signal conditions and RNNs to classify fail or success events in advance.
Ding et al. [151]	Faults detection and mitigation	LSTM	Introduces an on-line fault detection and mitigation approaches for common faults of cyber–physical systems through DL.
Mismar et al.[152]	Automatic fault management	Deep Q-learning	First work that utilizes deep Q-learning for fault management in SON.
Zhang et al.[154]	Automatic fault forecasting	LSTM	Develop a DL-based approach for automated failure forecasting in IT environments.

shows the superiority of the introduced approach over the existing approaches in the literature. The method introduced in this paper achieved an accuracy of 97.92% on DARPA'98 dataset and 98.42% accuracy on KDD'99. In this work, the authors made this interesting point that the current anomaly detection methods are inefficient since they cause computational complexity and produce high false-positives, especially for real-time anomaly detection in big data. The work has been conducted by Yousefi-Azar et al. showed the abilities of AE for cybersecurity applications, such as anomaly and malware detection [167]. The authors used a single AE model with the same topology for an anomaly and malware detection task and achieved promising results as the AE is an unsupervised generative model and can learn the original latent representation of traffic data. The proposed AE gave 83.34% of accuracy and outperformed its classical ML competitors, including Gaussian Naïve Bayes = 82.02%, Fuzzy classifier = 82.74%, and Decision Tree = 80.14%.

SAEs were proposed in 2017 by Vrizlynn [168] to perform anomaly detection and attack categorization tasks in the IEEE 802.11 network. To this end, the author first investigated the threats and attacks in the IEEE 802.11 network and then listed the challenges in gaining high accuracy in threat/attack classification in this network type. Finally, a DL approach based on SAE is proposed for anomaly detection/classification. The proposed method is able to learn the key features of data by itself and achieve an accuracy of 98.66%.

Chen et al. discussed security in Mobile Edge Computing (MEC) when used in transportation system applications [169]. The main reason behind this study is that the volume of communications has substantially risen among the connected edge devices. Hence, communication security is arising as a severe challenge. To address this challenge, the authors proposed a method for feature learning based on a deep belief network to identify unknown attacks in MEC. The proposed method is compared with the other four classical ML algorithms, including Decision Tree, Random Forest, Softmax Regression, and SVM, outperforming them in accuracy. Similarly, Nguyen et al. focused on Mobile Cloud Computing (MCC) security issues, such as availability of services, data integrity, and users' privacy, [170]. To this end, they adopted a preventive method based on DL to detect and isolate cyberattacks in MCC. In the paper, a Gaussian Binary Restricted Boltzmann Machine (GRBM) network is used for learning purposes, and NSL-KDD/UNSW-NB15/KDDcup is utilized for evaluating the proposed method. The experiment result shows the superiority of the proposed method over nine classical ML algorithms, including Decision Tree, K-means, K-NN, Logistic Regression, Gaussian Naive Bayes, Multinomial Naive Bayes, Bernoulli Naive Bayes, Random Forest Classifier, and SVM by 2%–16% in accuracy. The work in [171] developed a IDS for Vehicular Ad hoc Network (VANET) [172] based on DL. More specifically, the authors in this work leveraged deep GANs and distributed SDN in order to develop a collaborative IDS for VANETs. They claim that the proposed method can detect abnormal behaviors in the entire network, unlike the traditional methods that can detect only abnormal activities in local sub-networks. Moreover, they evaluate their method's performance under both independent identically distributed (IID) and non-IID scenarios. For example, on NSL-KDD and the KDD99 dataset and IID scenario, they achieved 0.951%–0.977% and 0.977%–0.984% accuracy under different settings, respectively.

Malware detection is an essential class of host IDS because malware (short for malicious software) can significantly impact a huge number of connected devices. Viruses, worms, trojans, spyware, bots, rootkits, and ransomware are prominent types of malware. Since the number and variety of malware are increasing, malware detection techniques need to be improved to protect communication systems and networks. To that end, DL-based Malware Detection Systems (MDSs) are promising techniques to deal with sophisticated malware attacks. For example, Zhong and Gu [173] proposed a multi-level DL-based system for malware detection to further enhance the performance of DL-based MDSs in terms of scalability and handling more complex data distributions of malware datasets. The authors adopted three DL models (i.e., CNN, LSTM, and RNN) as three structures of their systems. Each model is responsible for learning a specific data distribution of a particular class of malware. The authors compared the proposed method's performance (i.e., accuracy) with other methods such as ensemble DL, single CNN and RNN, SVM, and decision tree. The results demonstrate the supremacy of the provided system relative to its counterparts. Hardy et al. in [174] designed a DL architecture based on the SAEs model to detect malware intelligently. In this architecture, the SAEs performed as greedy layer-wise unsupervised pre-training operation, and then the supervised parameter fine-tuning idea is used to decline the error of prediction. The results indicate that the proposed architecture is able to improve the overall performance of malware detection by providing 0.956% accuracy. In other words, although shallow learning methods such as Naive Bayes, SVM, and decision tree provide excellent results in terms of accuracy, yet disappointing for malware detection tasks to some degree. Hence, the DL has been proposed to tackle this concern. Authors of [175] introduced a technique to detect Internet Of Battlefield Things (IoBT) malware by Operational Code (OpCode) sequence and deep Eigenspace learning. They transformed OpCodes into a vector space and then and adopt deep Eigenspace learning to

categorize malware. The authors declared that their method is robust and sustainable against malicious and benign applications.

Last but not least, botnet detection is another important class of host IDS. A botnet is a network of Internet-connected computers infected by bots. The bots are able to communicate with the attacker, also known as a botmaster. Botnets can be used to perform a wide range of subversive activities, such as DDoS attacks, send spam, click fraud, and bitcoin mining, to name a few. Different botnet detection approaches have been used in literature that are mostly based on passive monitoring of the network traffic. These approaches typically use ML algorithms to categorize network traffic and mainly depend on feature engineering and feature selection. ML-based methods need hard effort and domain knowledge to extract high-level network traffic patterns and the most relevant features. DL alleviates this problem as DL algorithms are able to automatically detect the important features and hierarchically extract features. As a result, DL has gained lots of interest in recent years for botnet detection purposes. For example, Roosmalen et al. [176] apply DL on packet flows to detect botnet traffic. More specifically, they use flows of TCP/UDP/IP-packets as inputs for a stacked denoising auto-encoders (SDAs) network to extract traffic features automatically, and then feed-forward supervised DNN is used for fine-tuning. Therefore, the proposed method does not need a feature engineering or a feature selection phase. The authors achieve a P2P-botnet detection accuracy of 99.7%. Pektas and Acarman [177] combine CNN and LSTM to detect botnets. The proposed approach consists of three phases, including feature extraction, building model, and evaluation phase. In this approach, network flows are used as the input in the first phase. Moreover, the authors use two datasets publicly available to evaluate their method, which achieves a classification accuracy of 99%. The work in [178] targets IoT-based botnet attacks. To be specific, the authors in this work deploy deep auto-encoders for botnet attacks detection. Unlike previous papers on IoT botnets detection, the authors use real IoT traffic data to evaluate their method. The IoT traffic data is collected from multiple commercial IoT devices infected by botnets. Analysis of results demonstrates the applicability of the proposed method for IoT botnets detection. The work of Torres et al. [179] performed an analysis of RNNs for botnet detection. They first referred to these facts that traditional botnet detection techniques are not efficient because botnets continually evolve and the behavioral analysis-based techniques are more applicable in modern communication systems. As a result, they assessed the viability of network traffic behavior recognition through RNN. During the performance evaluation of the RNN, the authors considered two common issues in such problems, i.e., imbalance network data and optimal length of sequences. The simulation results reveal that the RNN can categorize the network traffic with a high accuracy rate and a low false alarm rate.

A summary of the works reviewed in this section is provided in Table 6.

6. Future direction and open issues

In this section, we discuss future direction and open issues based on our findings after the literature review in terms of using DL in NTMA.

1. **Lack of labeled data:** The success of ML techniques especially DL algorithms, highly depends on the quality and quantity of the data used for training. Applying DL-based techniques for building prediction models in NTMA, calls for having huge amounts of network traffic. Although dramatic advances in communication systems and networks, such as IoT and 5G, lead to the production of massive volume of raw data every day, data labeling is a costly task in terms of time, computational overhead, and human effort. In real-world applications of networking, most of the data is unlabeled or semi-labeled [76]. Integrating DL techniques with other ML techniques that are able to work with unlabeled or semi-labeled data can be considered as a great solution. Active Learning has received much attention in addressing the challenge of labeling data [180].
2. **Difficulties in using DL for structured data:** Structured data refers to a standardized format that organizes data in tables with rows and columns, e.g., network traffic data. However, the most initial and successful applications of DL techniques are reported on problems with data that is unstructured, such as video, images, text, and audio. Some machine learning experts are against using DL for structured data because they believe that labeled structured datasets are not large enough for training DL algorithms. Moreover, they argue that classical ML algorithms, such as KNN and SVM, are much more simple and understandable than complex DL algorithms (e.g. GANs). Hence, they are the right choice for classical tasks, such as traffic classification and intrusion detection.

   Despite these objectives, one can refer to huge amounts of produced data by communication systems and networks that can be used for training DL algorithms but at the costs of labeling. In addition, thanks to the popularity of DL models in different domains, multiple frameworks have been established to facilitate using of DL. As prime examples of DL frameworks, we can refer to TensorFlow [181], PyTorch and Keras [182].
3. **Lack of successful or full exploitation of DL in some NTMA applications:** As is clear from Section 5.3, the lack of work on fault management is notable, and a considerable number of the reviewed papers have used DL for other tasks, such as network traffic classification and forecasting. However, the traditional methods for fault management, e.g. rule-based systems and algorithmic approaches suffer from serious disadvantages [139]. For example, in rule-based systems, network domain knowledge is needed to formulate and maintain the rule-sets; Or the functionalities of the algorithmic approaches are restricted to a specific problem area, and these approaches have to adopt a wide range of algorithms to support the complete problem space in the context of fault management. To overcome these disadvantages, one can exploit the capabilities of ML-based methods, especially DL for fault management. Specifically, ML-based methods can support a diverse range of problem areas, as well as eliminate the need for knowledge from domain experts through learning from fault data during the training phase [183,184].
4. **Resource-constrained networks:** Most of DL algorithms are designed to be trained and used by devices with sufficient resources. Training a DL algorithm with a large number of training samples and parameters, requires resource-rich devices in terms of computation, memory and power. This is in direct contradiction to the growing interest in the deployment of resource-constrained devices (e.g., IoT devices) equipped with AI- and learning-based technologies. The techniques in [185–189] have been proposed as ways to respond to this challenge. Regarding the enormous advances in the resource-constrained IoT devices and significant growth of the data produced at the edge of the network, it seems that the techniques based on federated learning will be more adaptive in the future [186,188]. In federated learning, a device can use local data in order to participate in the training process through updating the current model downloaded from a cloud and then only send back this update (weights). This learning approach allows using DL algorithms in resource-constrained devices as the training data is distributed among all participant devices and they use a shared model. As a result, with limited power, memory, and computational resources the devices can achieve great performance.
5. **Retraining challenge:** Unlike other applications of ML, networking suffers from high dynamicity and consequently it needs to retrain the models to be adapted with the new situations in a network. In a network, models should be retrained frequently because of the following events:

Table 6
A summary of works on network security.

Reference	Category	DL model	Key contribution
Papamartzivanos et al. [161]	Misuse detection in modern networks	AE+ sparse AE + MAPE-K	Introduces a scalable, self-adaptive and autonomous method for misuse detection for modern large-scale modern networks by leveraging DL.
Naseer et al. [163]	Anomaly detection system	CNN, AE and RNN	Designs and implements anomaly detection models based on different DL algorithms. Also, evaluates these models through standard classification metrics.
Malaiya et al.[165]	Anomaly detection	FCNs, VAE, and Seq2Seq	Examines multiple DL models for anomaly detection, including FCN, VAE, and LSTM.
Garg et al. [166]	Anomaly detection cloud datacenter	CNN+GWO	Proposes a robust hybrid method based on CNN and GWO for network anomaly detection in cloud environments, especially for streaming data.
Zhong and Gu [173]	Malware detection	CNN, RNN and LSTM	Introduces a multi-level DL system by using different DL models for malware detection.
Hardy et al. [174]	Malware detection	SAEs	Introduces a two-phase framework for malware detection based on SAEs model.
Azmoodeh et al. [175]	Malware detection for IoBT	OpCode+ deep Eigenspace learning	Proposes the first work based on OpCode deep learning technique for IoT and IoBT malware detection.
Roosmalen et al. [176]	Botnet detection	SDAs+ feed-forward supervised DNN	Discusses the application of DL for botnet detection and proposes a DL-based approach for botnet detection which utilizes TCP/UDP/IP packet flows as inputs.
Pektas and Acarman [177]	Botnet detection	CNN+LSTM	Provides a botnet detection method, in which both network flow information and DL are used. Moreover, it uses graph structure for feature extraction purposes.
Meidan et al. [178]	Botnet detection for IoT networks	Autoencoders	The first work that uses autoencoders in IoT networks for detecting botnet attacks. Also, unlike previous papers that use emulated or simulated data, this paper deploys real IoT traffic data for evaluation its method.
Yousefi-Azar et al. [167]	Anomaly and malware detection	Autoencoders	Provides an unsupervised feature learning method based on AE for cybersecurity purposes, e.g., anomaly and malware detection.
Vrizlynn [168]	Anomaly detection in IEEE 802.11 network	SAE	One of the few articles that consider anomaly detection in the IEEE 802.11 network through DL.
Chen et al. [169]	Attack detection in MEC	DBNs	Provides a feature learning model based on deep belief network to detect attacks in MEC.
Shu et al. [171]	Anomaly detection in VANETs	GANs	Proposes a collaborative methods by leveraging deep generative models and distributed SDN to detect anomalies in VANETs.
Torres et al. [179]	Botnet detection	RNN	Analyzes the performance of RNN for botnet detection purposes through the behavioral analysis of network traffic.
Nguyen et al. [170]	Attacks detection in MCC	GRBM	Leverages the GRBM network to develop an attack detection method for mobile cloud environments.
Jiang et al. [162]	Virtual MAC spoofing detection	CNN	Proposes a DL based detection system for MAC spooling attacks detection in virtualized environments.

- Daily training
- To be adapted with the new changes upon network manager's requests
- Triggered by detecting some events, e.g. security breaches, network behavior changes, or starting new packet streams.

Generally, DL models suffer from high complexity in the training phase as they consume plenty of resources and time. Most of the networking applications are time-consuming, thereby the time complexity of DL models to be retrained can be considered as a great challenge as DL models should be optimized in terms of time complexity and resource consumption.

6. **Theory of network:** In Internet Engineering Task Force 97 (IETF97), the challenge is introduced as networks suffer from the lack of a unified theory that can be applied to all networks. It means that the behavior of networks is heterogeneous based on their different topologies, equipment, scale, applications, etc. It causes an important problem that ML techniques should be trained for each network separately. The accuracy of ML techniques that are trained by public datasets can be reduced in different networks. There are some efforts to provide representative datasets, but it seems that the challenge increases the need for ML techniques that can label the data and re-train frequently for each network separately. Therefore, rather than public datasets, the DL techniques should be trained by exclusive datasets gathered from the target network and labeled with high accuracy. To solve this challenge, DL techniques should be learned for each network separately, but as mentioned above, retraining the DL models for each network is a time and resource consuming task.

7. Conclusion

This work has presented a comprehensive literature review of the applications of deep learning in network traffic monitoring and analysis. To this end, we have first given an introduction to deep learning, and then reviewed some of the related survey papers to highlight the differences between them and our paper. Afterwards, we have reviewed and discussed the advantages/disadvantages of the deep learning techniques used for NTMA applications. The applications include network traffic classification and prediction, fault management, and network security. Finally, we have discussed key challenges, open issues, and future directions based on our findings in the literature review.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] Alessandro D Alconzo, Idilio Drago, Andrea Morichetta, Marco Mellia, Pedro Casas, A survey on big data for network traffic monitoring and analysis, IEEE Trans. Netw. Serv. Manag. 16 (3) (2019) 800–813.

[2] Chakchai So-In, A survey of network traffic monitoring and analysis tools, in: Cse 576m computer system analysis project, Washington University in St. Louis, 2009.

[3] Alisha Cecil, A summary of network traffic monitoring and analysis techniques, Comput. Syst. Anal. (2006) 4–7.

[4] Amin Shahraki, Hamed Taherzadeh, Øystein Haugen, Last significant trend change detection method for offline poisson distribution datasets, in: 2017 International Symposium on Networks, Computers and Communications (ISNCC), IEEE, 2017, pp. 1–7.

[5] Amin Shahraki, Amir Taherkordi, Øystein Haugen, Frank Eliassen, Clustering objectives in wireless sensor networks: A survey and research direction analysis, Comput. Netw. 180 (2020) 107376.

[6] Jun Liu, Feng Liu, Nirwan Ansari, Monitoring and analyzing big traffic data of a large-scale cellular network with hadoop, IEEE Netw. 28 (4) (2014) 32–39.

[7] Se-Hee Han, Myung-Sup Kim, Hong-Taek Ju, James Won-Ki Hong, The architecture of NG-MON: A passive network monitoring system for high-speed IP networks, in: International Workshop on Distributed Systems: Operations and Management, Springer, 2002, pp. 16–27.

[8] Marco Ehrlich, Alexander Biendarra, Henning Trsek, Emanuel Wojtkowiak, Jürgen Jasperneite, Passive flow monitoring of hybrid network connections regarding quality of service parameters for the industrial automation, in: 8. Jahreskolloquium "Kommunikation in der Automation–KommA, 2017.

[9] Raouf Boutaba, Mohammad A Salahuddin, Noura Limam, Sara Ayoubi, Nashid Shahriar, Felipe Estrada-Solano, Oscar M Caicedo, A comprehensive survey on machine learning for networking: evolution, applications and research opportunities, J. Internet Serv. Appl. 9 (1) (2018) 16.

[10] Alexandros Labrinidis, Hosagrahar V. Jagadish, Challenges and opportunities with big data, Proc. VLDB Endow. 5 (12) (2012) 2032–2033.

[11] Uthayasankar Sivarajah, Muhammad Mustafa Kamal, Zahir Irani, Vishanth Weerakkody, Critical analysis of big data challenges and analytical methods, J. Bus. Res. 70 (2017) 263–286.

[12] Amin Shahraki, Marius Geitle, Øystein Haugen, A comparative node evaluation model for highly heterogeneous massive-scale internet of things-mist networks, Trans. Emerg. Telecommun. Technol. 31 (12) (2020) e3924.

[13] Amin Shahraki, Øystein Haugen, An outlier detection method to improve gathered datasets for network behavior analysis in IoT, Academy Publisher, 2019.

[14] Amin Shahraki, Amir Taherkordi, Øystein Haugen, Frank Eliassen, A survey and future directions on clustering: From WSNs to IoT and modern networking paradigms, IEEE Trans. Netw. Serv. Manag. (2020).

[15] Enrico Masala, Antonio Servetti, Simone Basso, Juan Carlos De Martin, Challenges and issues on collecting and analyzing large volumes of network data measurements, in: New Trends in Databases and Information Systems, Springer, 2014, pp. 203–212.

[16] Simone Basso, Antonio Servetti, Juan Carlos De Martin, Rationale, design, and implementation of the network neutrality bot, in: Congresso AICA 2010 (L'Aquila), 2010.

[17] Srikanth Sundaresan, Walter De Donato, Nick Feamster, Renata Teixeira, Sam Crawford, Antonio Pescapè, Broadband internet performance: a view from the gateway, ACM SIGCOMM Comput. Commun. Rev. 41 (4) (2011) 134–145.

[18] Donghao Zhou, Zheng Yan, Yulong Fu, Zhen Yao, A survey on network data collection, J. Netw. Comput. Appl. 116 (2018) 9–23.

[19] C. Morariu, B. Stiller, Dicap: Distributed packet capturing architecture for high-speed network links, in: 2008 33rd IEEE Conference on Local Computer Networks (LCN), 2008, pp. 168–175.

[20] Sihyung Lee, Kyriaki Levanti, Hyong S. Kim, Network monitoring: Present and future, Comput. Netw. 65 (2014) 84–98.

[21] Adam Oliner, Archana Ganapathi, Wei Xu, Advances and challenges in log analysis: Logs contain a wealth of information for help in managing systems., Queue 9 (12) (2011) 30–40.

[22] Yuri Demchenko, Cees De Laat, Peter Membrey, Defining architecture components of the big data ecosystem, in: 2014 International Conference on Collaboration Technologies and Systems (CTS), IEEE, 2014, pp. 104–112.

[23] Reihaneh H. Hariri, Erik M. Fredericks, Kate M. Bowers, Uncertainty in big data analytics: survey, opportunities, and challenges, J. Big Data 6 (1) (2019) 44.

[24] M Mazhar Rathore, Anand Paul, Won-Hwa Hong, HyunCheol Seo, Imtiaz Awan, Sharjil Saeed, Exploiting IoT and big data analytics: Defining smart digital city using real-time urban data, Sustain. Cities Soc. 40 (2018) 600–610.

[25] Maryam M Najafabadi, Flavio Villanustre, Taghi M Khoshgoftaar, Naeem Seliya, Randall Wald, Edin Muharemagic, Deep learning applications and challenges in big data analytics, J. Big Data 2 (1) (2015) 1.

[26] M. Gheisari, G. Wang, M.Z.A. Bhuiyan, A survey on deep learning in big data, in: 2017 IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), Vol. 2, 2017, pp. 173–180.

[27] Shahbaz Rezaei, Xin Liu, Deep learning for encrypted traffic classification: An overview, IEEE Commun. Mag. 57 (5) (2019) 76–81.

[28] Daniele Ucci, Leonardo Aniello, Roberto Baldoni, Survey of machine learning techniques for malware analysis, Comput. Secur. 81 (2019) 123–147.

[29] Mauro Conti, QianQian Li, Alberto Maragno, Riccardo Spolaor, The dark side(-channel) of mobile devices: A survey on network traffic analysis, 2017.

[30] Zubair Md Fadlullah, Fengxiao Tang, Bomin Mao, Nei Kato, Osamu Akashi, Takeru Inoue, Kimihiro Mizutani, State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems, IEEE Commun. Surv. Tutor. 19 (4) (2017) 2432–2455.

[31] Shikhar Verma, Yuichi Kawamoto, Zubair Md Fadlullah, Hiroki Nishiyama, Nei Kato, A survey on network methodologies for real-time analytics of massive IoT data and open research issues, IEEE Commun. Surv. Tutor. 19 (3) (2017) 1457–1477.

[32] Fabio Ricciato, Traffic monitoring and analysis for the optimization of a 3g network, IEEE Wirel. Commun. 13 (6) (2006) 42–49.

[33] Ran Dubin, Amit Dvir, Ofir Pele, Ofer Hadar, I know what you saw last minute–encrypted http adaptive video streaming title classification, IEEE Trans. Inform. Forensics Secur. 12 (12) (2017) 3039–3049.

[34] Andrew Pavlo, Erik Paulson, Alexander Rasin, Daniel J Abadi, David J DeWitt, Samuel Madden, Michael Stonebraker, A comparison of approaches to large-scale data analysis, in: Proceedings of the 2009 ACM SIGMOD International Conference on Management of Data, 2009, pp. 165–178.

[35] Amin Shahraki, Mahmoud Abbasi, Øystein Haugen, Boosting algorithms for network intrusion detection: A comparative evaluation of real adaboost, gentle adaboost and modest adaboost, Eng. Appl. Artif. Intell. 94 (2020) 103770.

[36] Terran Lane, Carla E. Brodley, An application of machine learning to anomaly detection, in: Proceedings of the 20th National Information Systems Security Conference, Vol. 377, Baltimore, USA, 1997, pp. 366–380.

[37] Arian Bär, Alessandro Finamore, Pedro Casas, Lukasz Golab, Marco Mellia, Large-scale network traffic monitoring with dbstream, a system for rolling big data analysis, in: 2014 IEEE International Conference on Big Data (Big Data), IEEE, 2014, pp. 165–170.

[38] Robert Cooley, Bamshad Mobasher, Jaideep Srivastava, Web mining: Information and pattern discovery on the world wide web, in: Proceedings Ninth IEEE International Conference on Tools with Artificial Intelligence, IEEE, 1997, pp. 558–567.

[39] Hadoop, Apache hadoop, 2020, https://hadoop.apache.org/.

[40] Spark, Apache spark, 2020, http://spark.apache.org/.

[41] Lizhe Wang, Jie Tao, Rajiv Ranjan, Holger Marten, Achim Streit, Jingying Chen, Dan Chen, G-hadoop: Mapreduce across distributed data centers for data-intensive computing, Future Gener. Comput. Syst. 29 (3) (2013) 739–750.

[42] Jing Wang, Jian Tang, Zhiyuan Xu, Yanzhi Wang, Guoliang Xue, Xing Zhang, Dejun Yang, Spatiotemporal modeling and prediction in cellular networks: A big data enabled deep learning approach, in: IEEE INFOCOM 2017-IEEE Conference on Computer Communications, IEEE, 2017, pp. 1–9.

[43] Justine Sherry, Chang Lan, Raluca Ada Popa, Sylvia Ratnasamy, Blindbox: Deep packet inspection over encrypted traffic, in: Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, 2015, pp. 213–226.

[44] Eiman Al Nuaimi, Hind Al Neyadi, Nader Mohamed, Jameela Al-Jaroodi, Applications of big data to smart cities, J. Internet Serv. Appl. 6 (1) (2015) 25.

[45] Kai Hwang, Min Chen, Big-data analytics for cloud, IoT and cognitive computing, John Wiley & Sons, 2017.

[46] A Tawalbeh Lo'ai, Rashid Mehmood, Elhadj Benkhlifa, Houbing Song, Mobile cloud computing model and big data analysis for healthcare applications, IEEE Access 4 (2016) 6171–6180.

[47] Xue-Wen Chen, Xiaotong Lin, Big data deep learning: challenges and perspectives, IEEE Access 2 (2014) 514–525.

[48] Pedro Casas, Alessandro D Alconzo, Tanja Zseby, Marco Mellia, Big-DAMA: big data analytics for network traffic monitoring and analysis, in: Proceedings of the 2016 Workshop on Fostering Latin-American Research in Data Communication Networks, 2016, pp. 1–3.

[49] Mohammad Lotfollahi, Mahdi Jafari Siavoshani, Ramin Shirali Hossein Zade, Mohammdsadegh Saberian, Deep packet: A novel approach for encrypted traffic classification using deep learning, Soft Comput. 24 (3) (2020) 1999–2012.

[50] Zhidan Liu, Zhenjiang Li, Kaishun Wu, Mo Li, Urban traffic prediction from mobility data using deep learning, IEEE Netw. 32 (4) (2018) 40–46.

[51] Yann LeCun, Yoshua Bengio, Geoffrey Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.

[52] Rusul Abduljabbar, Hussein Dia, Sohani Liyanage, Saeed Asadi Bagloee, Applications of artificial intelligence in transport: An overview, Sustainability 11 (1) (2019) 189.

[53] Aggeliki Androutsopoulou, Nikos Karacapilidis, Euripidis Loukis, Yannis Charalabidis, Transforming the communication between citizens and government through AI-guided chatbots, Gov. Inf. Q. 36 (2) (2019) 358–367.
[54] Christina L McDowell Marinchak, Edward Forrest, Bogdan Hoanca, The impact of artificial intelligence and virtual personal assistants on marketing, in: Encyclopedia of Information Science and Technology, Fourth Edition, IGI global, 2018, pp. 5748–5756.
[55] Stuart Russell, Peter Norvig, Artificial intelligence: a modern approach, 2002.
[56] Ian Goodfellow, Yoshua Bengio, Aaron Courville, Deep Learning, MIT press, 2016.
[57] Bruce W. Suter, The multilayer perceptron as an approximation to a Bayes optimal discriminant function, IEEE Trans. Neural Netw. 1 (4) (1990) 291.
[58] Yann LeCun, et al., Generalization and network design strategies. Connectionism in perspective, Elsiever, Zurich, Switzerland, 1989.
[59] Tuukka Salmi, Jussi Kiljander, Daniel Pakkala, Stacked boosters network architecture for short term load forecasting in buildings, 2020, arXiv preprint arXiv:2001.08406.
[60] David E. Rumelhart, Geoffrey E. Hinton, Ronald J. Williams, Learning internal representations by error propagation, Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.
[61] Jeeheh Oh, Jiaxuan Wang, Shengpu Tang, Michael Sjoding, Jenna Wiens, Relaxed weight sharing: Effectively modeling time-varying relationships in clinical time-series, 2019, arXiv preprint arXiv:1906.02898.
[62] Audrunas Gruslys, Rémi Munos, Ivo Danihelka, Marc Lanctot, Alex Graves, Memory-efficient backpropagation through time, in: Advances in Neural Information Processing Systems, 2016, pp. 4125–4133.
[63] Sepp Hochreiter, Jürgen Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[64] Alex Graves, Navdeep Jaitly, Towards end-to-end speech recognition with recurrent neural networks, in: International Conference on Machine Learning, 2014, pp. 1764–1772.
[65] Alex Graves, Jürgen Schmidhuber, Offline handwriting recognition with multidimensional recurrent neural networks, in: Advances in Neural Information Processing Systems, 2009, pp. 545–552.
[66] Ilya Sutskever, Oriol Vinyals, Quoc V. Le, Sequence to sequence learning with neural networks, in: Advances in Neural Information Processing Systems, 2014, pp. 3104–3112.
[67] Alex Graves, Generating sequences with recurrent neural networks, 2013.
[68] Ryan Kiros, Ruslan Salakhutdinov, Richard S. Zemel, Unifying visual-semantic embeddings with multimodal neural language models, 2014, arXiv preprint arXiv:1411.2539.
[69] S. Ren, K. He, R. Girshick, J. Sun, Advances in neural information processing systems, in: Faster R-CNN: towards real-time object detection with region proposal networks, 2015, pp. 91–99.
[70] Scott E. Fahlman, Geoffrey E. Hinton, Terrence J. Sejnowski, Massively parallel architectures for al: NETL, thistle, and Boltzmann machines, in: National Conference on Artificial Intelligence, AAAI, 1983.
[71] Tijmen Tieleman, Training restricted Boltzmann machines using approximations to the likelihood gradient, in: Proceedings of the 25th International Conference on Machine Learning, 2008, pp. 1064–1071.
[72] Geoffrey E. Hinton, Ruslan R. Salakhutdinov, Reducing the dimensionality of data with neural networks, Science 313 (5786) (2006) 504–507.
[73] Ruslan Salakhutdinov, Geoffrey Hinton, Deep boltzmann machines, in: Artificial Intelligence and Statistics, 2009, pp. 448–455.
[74] Geoffrey E. Hinton, Training products of experts by minimizing contrastive divergence, Neural Comput. 14 (8) (2002) 1771–1800.
[75] Tony Jebara, Machine learning: discriminative and generative, Vol. 755, Springer Science & Business Media, 2012.
[76] Mohammad Abu Alsheikh, Dusit Niyato, Shaowei Lin, Hwee-Pink Tan, Zhu Han, Mobile big data analytics using deep learning and apache spark, IEEE Netw. 30 (3) (2016) 22–29.
[77] Mehdi Mohammadi, Ala Al-Fuqaha, Sameh Sorour, Mohsen Guizani, Deep learning for IoT big data and streaming analytics: A survey, IEEE Commun. Surv. Tutor. 20 (4) (2018) 2923–2960.
[78] Qiang Yang, Yang Liu, Tianjian Chen, Yongxin Tong, Federated machine learning: Concept and applications, ACM Trans. Intell. Syst. Technol. (TIST) 10 (2) (2019) 1–19.
[79] Jakub Konečnỳ, H. Brendan McMahan, Felix X. Yu, Peter Richtárik, Ananda Theertha Suresh, Dave Bacon, Federated learning: Strategies for improving communication efficiency, 2016, arXiv preprint arXiv:1610.05492.
[80] Alberto Dainotti, Antonio Pescape, Kimberly C. Claffy, Issues and future directions in traffic classification, IEEE Netw. 26 (1) (2012) 35–40.
[81] Giuseppe Aceto, Domenico Ciuonzo, Antonio Montieri, Antonio Pescapé, Mobile encrypted traffic classification using deep learning: Experimental evaluation, lessons learned, and challenges, IEEE Trans. Netw. Serv. Manag. 16 (2) (2019) 445–458.
[82] Pan Wang, Feng Ye, Xuejiao Chen, Yi Qian, Datanet: Deep learning based encrypted network traffic classification in sdn home gateway, IEEE Access 6 (2018) 55380–55391.
[83] Qing Lyu, Xingjian Lu, Effective Media Traffic Classification Using Deep Learning, in: Proceedings of the 2019 3rd International Conference on Compute and Data Analysis, 2019, pp. 139–146.
[84] Ed Wilson Tavares Ferreira, Ailton Akira Shinoda, The development and evaluation of a dataset for testing of IDS for wireless networks, IEEE Latin Amer. Trans. 14 (1) (2016) 404–410.
[85] Phyo Htet Pwint, Thanda Shwe, Network traffic anomaly detection based on apache spark, in: 2019 International Conference on Advanced Information Technologies (ICAIT), IEEE, 2019, pp. 222–226.
[86] Zahra Salek, Fariborz Mousavi Madani, Reza Azmi, Intrusion detection using neuarl networks trained by differential evaluation algorithm, in: 2013 10th International ISC Conference on Information Security and Cryptology (ISCISC), IEEE, 2013, pp. 1–6.
[87] Azar Abid Salih, Maiwan Bahjat Abdulrazaq, Combining best features selection using three classifiers in intrusion detection system, in: 2019 International Conference on Advanced Science and Engineering (ICOASE), IEEE, 2019, pp. 94–99.
[88] Manasa Sreekesh, et al., A two-tier network based intrusion detection system architecture using machine learning approach, in: 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT), IEEE, 2016, pp. 42–47.
[89] Lennart Van Efferen, Amr M.T. Ali-Eldin, A multi-layer perceptron approach for flow-based anomaly detection, in: 2017 International Symposium on Networks, Computers and Communications (ISNCC), IEEE, 2017, pp. 1–6.
[90] S. Miller, K. Curran, T. Lunney, Multilayer perceptron neural network for detection of encrypted vpn network traffic, in: 2018 International Conference on Cyber Situational Awareness, Data Analytics and Assessment (Cyber SA), 2018, pp. 1–8.
[91] R. Sahay, G. Geethakumari, K. Modugu, B. Mitra, Traffic convergence detection in IoT LLNs: A multilayer perceptron based mechanism, in: 2018 IEEE Symposium Series on Computational Intelligence (SSCI), 2018, pp. 1715–1722.
[92] Meng Wang, Yiqin Lu, Jiancheng Qin, A dynamic MLP-based ddos attack detection method using feature selection and feedback, Comput. Secur. 88 (2020) 101645.
[93] Wei Wang, Ming Zhu, Jinlin Wang, Xuewen Zeng, Zhongzhen Yang, End-to-end encrypted traffic classification with one-dimensional convolution neural networks, in: 2017 IEEE International Conference on Intelligence and Security Informatics (ISI), IEEE, 2017, pp. 43–48.
[94] Zhitang Chen, Ke He, Jian Li, Yanhui Geng, Seq2img: A sequence-to-image based approach towards IP traffic classification using convolutional neural networks, in: 2017 IEEE International Conference on Big Data (Big Data), IEEE, 2017, pp. 1271–1276.
[95] Shahbaz Rezaei, Xin Liu, How to achieve high classification accuracy with just a few labels: A semi-supervised approach using sampled packets, 2018, arXiv preprint arXiv:1812.09761.
[96] Wei Wang, Yiqiang Sheng, Jinlin Wang, Xuewen Zeng, Xiaozhou Ye, Yongzhong Huang, Ming Zhu, HAST-IDS: Learning hierarchical spatial-temporal features using deep neural networks to improve intrusion detection, IEEE Access 6 (2017) 1792–1806.
[97] M Yeo, Y Koo, Y Yoon, T Hwang, J Ryu, J Song, C Park, Flow-based malware detection using convolutional neural network, in: 2018 International Conference on Information Networking (ICOIN), IEEE, 2018, pp. 910–913.
[98] Manuel Lopez-Martin, Belen Carro, Antonio Sanchez-Esguevillas, Jaime Lloret, Network traffic classifier with convolutional and recurrent neural networks for internet of things, IEEE Access 5 (2017) 18042–18050.
[99] Van Tong, Hai Anh Tran, Sami Souihi, Abdelhamid Mellouk, A novel QUIC traffic classifier based on convolutional neural networks, in: 2018 IEEE Global Communications Conference (GLOBECOM), IEEE, 2018, pp. 1–6.
[100] Wei Wang, Ming Zhu, Xuewen Zeng, Xiaozhou Ye, Yiqiang Sheng, Malware traffic classification using convolutional neural network for representation learning, in: 2017 International Conference on Information Networking (ICOIN), IEEE, 2017, pp. 712–717.
[101] Benjamin J Radford, Leonardo M Apolonio, Antonio J Trias, Jim A Simpson, Network traffic anomaly detection using recurrent neural networks, 2018, arXiv preprint arXiv:1803.10769.
[102] Shuyuan Zhao, Yongzheng Zhang, Yafei Sang, Towards unknown traffic identification via embeddings and deep autoencoders, in: 2019 26th International Conference on Telecommunications (ICT), IEEE, 2019, pp. 85–89.
[103] Peng Li, Zhikui Chen, Laurence T Yang, Jing Gao, Qingchen Zhang, M Jamal Deen, An improved stacked auto-encoder for network traffic flow classification, IEEE Netw. 32 (6) (2018) 22–27.
[104] Abebe Abeshu, Naveen Chilamkurti, Deep learning: The frontier for distributed attack detection in fog-to-things computing, IEEE Commun. Mag. 56 (2) (2018) 169–175.
[105] Ly Vu, Cong Thanh Bui, Quang Uy Nguyen, A deep learning based method for handling imbalanced problem in network traffic classification, in: Proceedings of the Eighth International Symposium on Information and Communication Technology, 2017, pp. 333–339.
[106] Md Zahangir Alom, VenkataRamesh Bontupalli, Tarek M. Taha, Intrusion detection using deep belief networks, in: 2015 National Aerospace and Electronics Conference (NAECON), IEEE, 2015, pp. 339–344.

[107] Auwal Sani Iliyasu, Huifang Deng, Semi-supervised encrypted traffic classification with deep convolutional generative adversarial networks, IEEE Access 8 (2019) 118–126.
[108] Rongpeng Li, Zhifeng Zhao, Xuan Zhou, Jacques Palicot, Honggang Zhang, The prediction analysis of cellular radio access network traffic: From entropy theory to networking practice, IEEE Commun. Mag. 52 (6) (2014) 234–240.
[109] Ali Imran, Ahmed Zoha, Adnan Abu-Dayya, Challenges in 5G: how to empower SON with big data for enabling 5G, IEEE Netw. 28 (6) (2014) 27–33.
[110] Fengli Xu, Yuyun Lin, Jiaxin Huang, Di Wu, Hongzhi Shi, Jeungeun Song, Yong Li, Big data driven mobile traffic understanding and forecasting: A time series approach, IEEE Trans. Serv. Comput. 9 (5) (2016) 796–805.
[111] I. Cisco, Cisco visual networking index: Forecast and methodology, 2011–2016, Vol. 518, CISCO White paper, 2012.
[112] Ziaul Hasan, Hamidreza Boostanimehr, Vijay K. Bhargava, Green cellular networks: A survey, some research issues and challenges, IEEE Commun. Surv. Tutor. 13 (4) (2011) 524–540.
[113] Chih-Wei Huang, Chiu-Ti Chiang, Qiuhui Li, A study of deep learning networks on mobile traffic forecasting, in: 2017 IEEE 28th Annual International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC), IEEE, 2017, pp. 1–6.
[114] Haytham Assem, Bora Caglayan, Teodora Sandra Buda, Declan O'Sullivan, St-dennetfus: A new deep learning approach for network demand prediction, in: Joint European Conference on Machine Learning and Knowledge Discovery in Databases, Springer, 2018, pp. 222–237.
[115] H. Zare Moayedi, M.A. Masnadi-Shirazi, Arima model for network traffic prediction and anomaly detection, in: 2008 International Symposium on Information Technology, Vol. 4, 2008, pp. 1–6.
[116] Amin Azari, Panagiotis Papapetrou, Stojan Denic, Gunnar Peters, Cellular traffic prediction and classification: a comparative evaluation of LSTM and ARIMA, 2019.
[117] D. Tikunov, Toshikazu Nishimura, Traffic prediction for mobile network using holt-winter's exponential smoothing, in: 2007 15th International Conference on Software, Telecommunications and Computer Networks, 2007, pp. 1–5.
[118] Yantai Shu, Minfang Yu, Oliver Yang, Jiakun Liu, Huifang Feng, Wireless traffic modeling and prediction using seasonal ARIMA models, IEICE Trans. Commun. 88 (10) (2005) 3992–3999.
[119] Marjan Kuchaki Rafsanjani, Atieh Rezaei, Amin Shahraki, Arsham Borumand Saeid, QARIMA: A new approach to prediction in queue theory, Appl. Math. Comput. 244 (2014) 514–525.
[120] Tiago Prado Oliveira, Jamil Salem Barbar, Alexsandro Santos Soares, Computer network traffic prediction: a comparison between traditional and deep learning neural networks, Int. J. Big Data Intell. 3 (1) (2016) 28–37.
[121] Manish Joshi, Theyazn Hassn Hadi, A review of network traffic analysis and prediction techniques, 2015, arXiv preprint arXiv:1507.05722.
[122] M. Barabas, G. Boanea, A.B. Rus, V. Dobrota, J. Domingo-Pascual, Evaluation of network traffic prediction based on neural networks with multi-task learning and multiresolution decomposition, in: 2011 IEEE 7th International Conference on Intelligent Computer Communication and Processing, 2011, pp. 95–102.
[123] Haytham Assem, Declan O'Sullivan, Discovering new socio-demographic regional patterns in cities, in: Proceedings of the 9th ACM SIGSPATIAL Workshop on Location-Based Social Networks, 2016, pp. 1–9.
[124] Rishabh Madan, Partha SarathiMangipudi, Predicting computer network traffic: a time series forecasting approach using DWT, ARIMA and RNN, in: 2018 Eleventh International Conference on Contemporary Computing (IC3), IEEE, 2018, pp. 1–5.
[125] Davide Andreoletti, Sebastian Troia, Francesco Musumeci, Silvia Giordano, Guido Maier, Massimo Tornatore, Network traffic prediction based on diffusion convolutional recurrent neural networks, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), IEEE, 2019, pp. 246–251.
[126] Jie Feng, Xinlei Chen, Rundong Gao, Ming Zeng, Yong Li, Deeptp: An end-to-end neural network for mobile cellular traffic prediction, IEEE Network 32 (6) (2018) 108–115.
[127] Anestis Dalgkitsis, Malamati Louta, George T. Karetsos, Traffic forecasting in cellular networks using the LSTM RNN, in: Proceedings of the 22nd Pan-Hellenic Conference on Informatics, 2018, pp. 28–33.
[128] Luoyang Fang, Xiang Cheng, Haonan Wang, Liuqing Yang, Mobile demand forecasting via deep graph-sequence spatiotemporal modeling in cellular networks, IEEE Internet Things J. 5 (4) (2018) 3091–3101.
[129] Chaoyun Zhang, Paul Patras, Long-term mobile traffic forecasting using deep spatio-temporal neural networks, in: Proceedings of the Eighteenth ACM International Symposium on Mobile Ad Hoc Networking and Computing, 2018, pp. 231–240.
[130] Tiago Prado Oliveira, Jamil Salem Barbar, Alexsandro Santos Soares, Multilayer perceptron and stacked autoencoder for internet traffic prediction, in: IFIP International Conference on Network and Parallel Computing, Springer, 2014, pp. 61–71.
[131] Dario Bega, Marco Gramaglia, Marco Fiore, Albert Banchs, Xavier Costa-Perez, Deepcog: Cognitive network management in sliced 5g networks with deep learning, in: IEEE INFOCOM 2019-IEEE Conference on Computer Communications, IEEE, 2019, pp. 280–288.
[132] Dario Bega, Marco Gramaglia, Marco Fiore, Albert Banchs, Xavier Costa-Perez, AZTEC: Anticipatory capacity allocation for zero-touch network slicing, in: IEEE INFOCOM 2020-IEEE Conference on Computer Communications, IEEE, 2020, pp. 794–803.
[133] Chuanting Zhang, Haixia Zhang, Dongfeng Yuan, Minggao Zhang, Citywide cellular traffic prediction based on densely connected convolutional neural networks, IEEE Commun. Lett. 22 (8) (2018) 1656–1659.
[134] Xiaofeng Cao, Yuhua Zhong, Yun Zhou, Jiang Wang, Cheng Zhu, Weiming Zhang, Interactive temporal recurrent convolution network for traffic prediction in data centers, IEEE Access 6 (2017) 5276–5289.
[135] Laisen Nie, Xiaojie Wang, Liangtian Wan, Shui Yu, Houbing Song, Dingde Jiang, Network traffic prediction based on deep belief network and spatiotemporal compressive sensing in wireless mesh backbone networks, Wirel. Commun. Mob. Comput. 2018 (2018).
[136] Ming Li, Yuewen Wang, Zhaowen Wang, Huiying Zheng, A deep learning method based on an attention mechanism for wireless network traffic prediction, Ad Hoc Netw. 107 (2020) 102258.
[137] Xu Wang, Zimu Zhou, Fu Xiao, Kai Xing, Zheng Yang, Yunhao Liu, Chunyi Peng, Spatio-temporal analysis and prediction of cellular traffic in metropolis, IEEE Trans. Mob. Comput. 18 (9) (2018) 2190–2202.
[138] Hongyi Zeng, Peyman Kazemian, George Varghese, Nick McKeown, Automatic test packet generation, in: Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, 2012, pp. 241–252.
[139] David Mulvey, Chuan Heng Foh, Muhammad Ali Imran, Rahim Tafazolli, Cell fault management using machine learning techniques, IEEE Access 7 (2019) 124514–124539.
[140] T. Benson, A. Akella, D. Maltz, Unraveling the complexity of network management, in: Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation, ser. NSDI'09, 2009.
[141] Paulo Cesar da Rocha Fonseca, Edjard Souza Mota, A survey on fault management in software-defined networks, IEEE Commun. Surv. Tutor. 19 (4) (2017) 2284–2321.
[142] John S Baras, M Ball, S Gupta, P Viswanathan, P Shah, Automated network fault management, in: MILCOM 97 MILCOM 97 Proceedings, Vol. 3, IEEE, 1997, pp. 1244–1250.
[143] Srinikethan Madapuzi Srinivasan, Tram Truong-Huu, Mohan Gurusamy, Machine learning-based link fault identification and localization in complex networks, IEEE Internet Things J. 6 (4) (2019) 6556–6566.
[144] Huakun Huang, Lingjun Zhao, Huawei Huang, Song Guo, Machine fault detection for intelligent self-driving networks, IEEE Commun. Mag. 58 (1) (2020) 40–46.
[145] David Mulvey, Chuan Heng Foh, Muhammad Ali Imran, Rahim Tafazolli, Cell coverage degradation detection using deep learning techniques, in: 2018 International Conference on Information and Communication Technology Convergence (ICTC), IEEE, 2018, pp. 441–447.
[146] Usama Masood, Ahmad Asghar, Ali Imran, Adnan Noor Mian, Deep learning based detection of sleeping cells in next generation cellular networks, in: 2018 IEEE Global Communications Conference (GLOBECOM), IEEE, 2018, pp. 206–212.
[147] Ayush Dusia, Adarshpal S. Sethi, Recent advances in fault localization in computer networks, IEEE Commun. Surv. Tutor. 18 (4) (2016) 3030–3051.
[148] Lav Gupta, Tara Salman, Ria Das, Aiman Erbad, Raj Jain, Mohammed Samaka, HYPER-VINES: A hybrid learning fault and performance issues eradicator for virtual network services over multi-cloud systems, in: 2019 International Conference on Computing, Networking and Communications (ICNC), IEEE, 2019, pp. 141–147.
[149] Guangyang Qi, Lina Yao, Anton V. Uzunov, Fault detection and localization in distributed systems using recurrent convolutional neural networks, in: International Conference on Advanced Data Mining and Applications, Springer, 2017, pp. 33–48.
[150] Shubham Khunteta, Ashok Kumar Reddy Chavva, Deep learning based link failure mitigation, in: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA), IEEE, 2017, pp. 806–811.
[151] Kai Ding, Sheng Ding, Andrey Morozov, Tagir Fabarisov, Klaus Janschek, Online error detection and mitigation for time-series data of cyber-physical systems using deep learning based methods, in: 2019 15th European Dependable Computing Conference (EDCC), IEEE, 2019, pp. 7–14.
[152] Faris B. Mismar, Brian L. Evans, Deep Q-learning for self-organizing networks fault management and radio performance improvement, in: 2018 52nd Asilomar Conference on Signals, Systems, and Computers, IEEE, 2018, pp. 1457–1461.
[153] Ahmad Asghar, Hasan Farooq, Ali Imran, Self-healing in emerging cellular networks: review, challenges, and research directions, IEEE Commun. Surv. Tutor. 20 (3) (2018) 1682–1709.
[154] Ke Zhang, Jianwu Xu, Martin Renqiang Min, Guofei Jiang, Konstantinos Pelechrinis, Hui Zhang, Automated IT system failure prediction: A deep learning approach, in: 2016 IEEE International Conference on Big Data (Big Data), IEEE, 2016, pp. 1291–1300.
[155] Daniel S Berman, Anna L Buczak, Jeffrey S Chavis, Cherita L Corbett, A survey of deep learning methods for cyber security, Information 10 (4) (2019) 122.

[156] Hesham ElSawy, Ekram Hossain, Martin Haenggi, Stochastic geometry for modeling, analysis, and design of multi-tier and cognitive cellular wireless networks: A survey, IEEE Commun. Surv. Tutor. 15 (3) (2013) 996–1019.
[157] Osianoh Glenn Aliu, Ali Imran, Muhammad Ali Imran, Barry Evans, A survey of self organisation in future cellular networks, IEEE Commun. Surv. Tutor. 15 (1) (2012) 336–361.
[158] ITU, global ICT statistics.
[159] herjavecgroup, Official annual cybercrime report, 2019.
[160] Chirag N. Modi, Kamatchi Acha, Virtualization layer security challenges and intrusion detection/prevention systems in cloud computing: a comprehensive review, J. Supercomput. 73 (3) (2017) 1192–1234.
[161] Dimitrios Papamartzivanos, Félix Gómez Mármol, Georgios Kambourakis, Introducing deep learning self-adaptive misuse network intrusion detection systems, IEEE Access 7 (2019) 13546–13560.
[162] Peng Jiang, Hongyi Wu, Cong Wang, Chunsheng Xin, Virtual MAC spoofing detection through deep learning, in: 2018 IEEE International Conference on Communications (ICC), IEEE, 2018, pp. 1–6.
[163] Sheraz Naseer, Yasir Saleem, Shehzad Khalid, Muhammad Khawar Bashir, Jihun Han, Muhammad Munwar Iqbal, Kijun Han, Enhanced network anomaly detection based on deep neural networks, IEEE Access 6 (2018) 48231–48246.
[164] Mahbod Tavallaee, Ebrahim Bagheri, Wei Lu, Ali A Ghorbani, A detailed analysis of the KDD cup 99 data set, in: 2009 IEEE Symposium on Computational Intelligence for Security and Defense Applications, IEEE, 2009, pp. 1–6.
[165] Ritesh K Malaiya, Donghwoon Kwon, Jinoh Kim, Sang C Suh, Hyunjoo Kim, Ikkyun Kim, An empirical evaluation of deep learning for network anomaly detection, in: 2018 International Conference on Computing, Networking and Communications (ICNC), IEEE, 2018, pp. 893–898.
[166] Sahil Garg, Kuljeet Kaur, Neeraj Kumar, Georges Kaddoum, Albert Y Zomaya, Rajiv Ranjan, A hybrid deep learning-based model for anomaly detection in cloud datacenter networks, IEEE Trans. Netw. Serv. Manag. 16 (3) (2019) 924–935.
[167] Mahmood Yousefi-Azar, Vijay Varadharajan, Len Hamey, Uday Tupakula, Autoencoder-based feature learning for cyber security applications, in: 2017 International Joint Conference on Neural Networks (IJCNN), IEEE, 2017, pp. 3854–3861.
[168] Vrizlynn L.L. Thing, IEEE 802.11 network anomaly detection and attack classification: A deep learning approach, in: 2017 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2017, pp. 1–6.
[169] Yuanfang Chen, Yan Zhang, Sabita Maharjan, Muhammad Alam, Ting Wu, Deep learning for secure mobile edge computing in cyber-physical transportation systems, IEEE Netw. 33 (4) (2019) 36–41.
[170] Khoi Khac Nguyen, Dinh Thai Hoang, Dusit Niyato, Ping Wang, Diep Nguyen, Eryk Dutkiewicz, Cyberattack detection in mobile cloud computing: A deep learning approach, in: 2018 IEEE Wireless Communications and Networking Conference (WCNC), IEEE, 2018, pp. 1–6.
[171] Jiangang Shu, Lei Zhou, Weizhe Zhang, Xiaojiang Du, Mohsen Guizani, Collaborative intrusion detection for VANETs: A deep learning-based distributed SDN approach, IEEE Trans. Intell. Transp. Syst. (2020).
[172] Mahmoud Abbasi, Amin Shahraki, Hamid R Barzegar, Claus Pahl, Synchronization Techniques in ''Device to Device-and Vehicle to Vehicle-Enabled'' Cellular Networks: A survey, Comput. Electr. Eng., 90 106955.
[173] Wei Zhong, Feng Gu, A multi-level deep learning system for malware detection, Expert Syst. Appl. 133 (2019) 151–162.
[174] William Hardy, Lingwei Chen, Shifu Hou, Yanfang Ye, Xin Li, DL4MD: A deep learning framework for intelligent malware detection, in: Proceedings of the International Conference on Data Mining (DMIN), The Steering Committee of The World Congress in Computer Science, Computer..., 2016, p. 61.
[175] Amin Azmoodeh, Ali Dehghantanha, Kim-Kwang Raymond Choo, Robust malware detection for internet of (battlefield) things devices using deep eigenspace learning, IEEE Trans. Sustain. Comput. 4 (1) (2018) 88–95.
[176] Jos van Roosmalen, Harald Vranken, Marko van Eekelen, Applying deep learning on packet flows for botnet detection, in: Proceedings of the 33rd Annual ACM Symposium on Applied Computing, 2018, pp. 1629–1636.
[177] Abdurrahman Pektaş, Tankut Acarman, Deep learning to detect botnet via network flow summaries, Neural Comput. Appl. 31 (11) (2019) 8021–8033.
[178] Yair Meidan, Michael Bohadana, Yael Mathov, Yisroel Mirsky, Asaf Shabtai, Dominik Breitenbacher, Yuval Elovici, N-baiot—network-based detection of iot botnet attacks using deep autoencoders, IEEE Pervasive Comput. 17 (3) (2018) 12–22.
[179] Pablo Torres, Carlos Catania, Sebastian Garcia, Carlos Garcia Garino, An analysis of recurrent neural networks for botnet detection behavior, in: 2016 IEEE Biennial Congress of Argentina (ARGENCON), IEEE, 2016, pp. 1–6.
[180] Yuji Roh, Geon Heo, Steven Euijong Whang, A survey on data collection for machine learning: a big data-ai integration perspective, IEEE Trans. Knowl. Data Eng. (2019).
[181] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mane, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viegas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, Xiaoqiang Zheng, Tensorflow: Large-scale machine learning on heterogeneous distributed systems, 2016.
[182] François Chollet, et al., Keras, 2015, https://github.com/fchollet/keras.
[183] Xiaofei Wang, Xiuhua Li, Victor C.M. Leung, Artificial intelligence-based techniques for emerging heterogeneous network: State of the arts, opportunities, and challenges, IEEE Access 3 (2015) 1379–1391.
[184] Mario Bkassiny, Yang Li, Sudharman K. Jayaweera, A survey on machine-learning techniques in cognitive radios, IEEE Commu. Surv. Tutor. 15 (3) (2012) 1136–1159.
[185] Mohammad Motamedi, Daniel Fong, Soheil Ghiasi, Machine intelligence on resource-constrained IoT devices: The case of thread granularity optimization for CNN inference, ACM Trans. Embedded Comput. Syst. (TECS) 16 (5s) (2017) 1–19.
[186] Ahmed Imteaj, Urmish Thakker, Shiqiang Wang, Jian Li, M Hadi Amini, Federated learning for resource-constrained IoT devices: Panoramas and state-of-the-art, 2020, arXiv preprint arXiv:2002.10610.
[187] Zhuoran Zhao, Kamyar Mirzazad Barijough, Andreas Gerstlauer, Deepthings: Distributed adaptive deep learning inference on resource-constrained IoT edge clusters, IEEE Trans. Comput.-Aided Des. Integr. Circuits Syst. 37 (11) (2018) 2348–2359.
[188] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K Leung, Christian Makaya, Ting He, Kevin Chan, Adaptive federated learning in resource constrained edge computing systems, IEEE J. Sel. Areas Commun. 37 (6) (2019) 1205–1221.
[189] Rui Hu, Yuanxiong Guo, E. Paul Ratazzi, Yanmin Gong, Differentially private federated learning for resource-constrained internet of things, 2020, arXiv preprint arXiv:2003.12705.