IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 15 November 2023, accepted 6 January 2024, date of publication 23 May 2024, date of current version 25 September 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3404866

TUTORIAL

Neural Networks in Selected Aspects of Communications and Networking

PIOTR BORYLO, EDYTA BIERNACKA, JERZY DOMZAL, BARTOSZ KADZIOLKA, MIROSLAW KANTOR, KRZYSZTOF RUSEK, MACIEJ SKALA, KRZYSZTOF WAJDA, (Member, IEEE), ROBERT WOJCIK, AND WOJCIECH ZABEK

AGH University of Krakow, Institute of Telecommunications, 30-059 Krakow, Poland

Corresponding author: Piotr Borylo (piotr.borylo@agh.edu.pl)

This work was supported by the Project "Intelligent Management of Traffic in Multi-Layer Software-Defined Networks" funded by Polish National Science Centre under Project 2017/25/B/ST6/02186.

ABSTRACT As a consequence of increased complexity, technical requirements of services and expectations of end-users, telecommunication networks and systems must still increase their efficiency. Research currently focuses on introducing automation and intelligence into network control and management. Neural Networks form a class of Machine Learning techniques utilizing the supervised learning paradigm. The class is mature and comprises a significant number of algorithms. As a result, Neural Networks-based solutions are widely applicable, also in the context of communications and networking. The aim of this tutorial is twofold. Firstly, to provide fundamentals regarding the Neural Network (NN) method. Secondly, to comprehensively study the examples of applying NN-based solutions to solve problems in different aspects of communications and networking. Studies are supplemented with additional explanations, figures and critical considerations, including pros and cons of using selected methods for particular purposes. This part uniquely complements the tutorial one and facilitates in-depth understanding of NN. Based on the conducted studies, we draw a comparative analysis, summaries and expected future research topics and challenges of using NN in communications and networking.

INDEX TERMS Computer networks, example-based tutorial, neural networks.

I. INTRODUCTION

The number and variety of intelligent end devices is increasing, catalyzed by the concept of Internet of Things (IoT). To ensure efficient and full connectivity, various networking protocols and transmission techniques must be interconnected in both single- and multi-domain scenarios. Furthermore, modern services such as virtual reality, tactile internet, or vehicular networks impose that overall traffic grows exponentially. Not only the amount of data is increasing, but also the speed of transmission as a core network infrastructure is developed to reflect changes at the edge, where high-throughput connectivity is being provided to the end-stations by 5G or Fiber to the Home technologies. To achieve a desired Quality of Experience (QoE) for end-users accessing novel services, both service providers and network operators must cooperate to optimize numerous and adequate Quality of Service Quality of Service (QoS) factors in a dynamic and competitive environment. Finally, additional requirements on an efficient management plane are imposed by managers willing not only to optimize operating and capital expenditures, but also to simultaneously provide services to multiple tenants with reasonable fairness.

Network operators and administrators are not able to manually manage and monitor such complex infrastructures. However, analytical modelling of such an infrastructure requires at least expert's knowledge and in numerous cases is even impossible. Approaches must deal with a tremendous number of variables and a large solution space, impractical for modelling. To overcome these issues, Machine Learning (ML) is being used to solve the advanced problems with the use of computer programs. The ML algorithm instructs the machine how to learn about the environment and how to discover the patterns ruling complex ecosystems. Based

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

on the collected knowledge, the algorithm can, for example, classify samples, predict the future, or take some actions affecting the environment.

Emerging technologies and networking concepts create favorable conditions to meet the technical requirements and overcome the limitations of ML. For example, software-defined networking creates an ability to collect knowledge about the infrastructure in a central point, feed ML algorithms with input data, and distribute expected future actions over the network. A sophisticated control plane is widely ensured in the core network through the existing Network Management System and is being extended toward access thanks to the 5G standard and Virtual Network Functions. End-nodes are sufficiently intelligent to perform the actions defined by the ML solution. Finally, virtualization and cloud-based solutions facilitate a training process offering specialized hardware equipped with Graphical and Tensor Processing Units, distributed data processing frameworks (e.g., Hadoop and Spark), or long-term and virtually unlimited storage.

Although, over the years, several ML techniques reached maturity, some became more popular than the others. Reinforcement Learning (RL) is one of the most attractive and popular solutions applicable to a wide variety of different aspects of communications and networking. Its development is further boosted by the favorable conditions created by both the existing IT infrastructure and evolving network architectures. In [1], we have published a tutorial on RL. Its purpose is: (1) to provide fundamentals regarding the RL method, and (2) to comprehensively study the examples of applying RL-based solutions to solve problems in different aspects of communications and networking.

This paper, however, focuses on NN being, besides RL, one of the most popular ML techniques. We firstly provide a tutorial on NN to clearly explain its fundamentals and properties. Then, we comprehensively analyze examples of using NN in different aspects of communications and networking to illustrate features of NN based on particular examples. We explain how the problem was modeled, how the neural network was designed, and what is the aim of running NN. For each work, we provide the original figures, block diagrams and summarizing tables with the aim to facilitate understanding of papers. This part significantly extends the classical tutorial and further presents crucial issues from a different perspective, fills potential gaps, and sheds light on potential future research avenues. To boost the development of NN in the networking domain, we point out the shortcomings and limitations of each work. These conclusions and critical insights aim to boost further development of NN in the networking domain.

We selected the following network aspects to illustrate the application of NN techniques in a wide range of representative problems and circumstances:

- **traffic prediction** — the ability to forecast future or real-time traffic information based on historical traffic data, such proactive prediction-based approach may strongly improve QoS when compared to reactive methods;
- **traffic engineering** — the ability to optimize utilization of network resources and simultaneously to assure required level of service guarantees (QoS and GoS), expressed by properly tailored routing procedures based on available information from many sources, also historical data;
- **quality of service and quality of experience** — the assessments of quality based on network performance metrics or user experiences;
- **reliability and robustness** — the ability to avoid, counteract, and fix service outages is critical and demanding in complex systems of cooperating unreliable components;
- **security** — the ability to protect the integrity, confidentiality and accessibility of computer network infrastructure by using both software and hardware technologies.

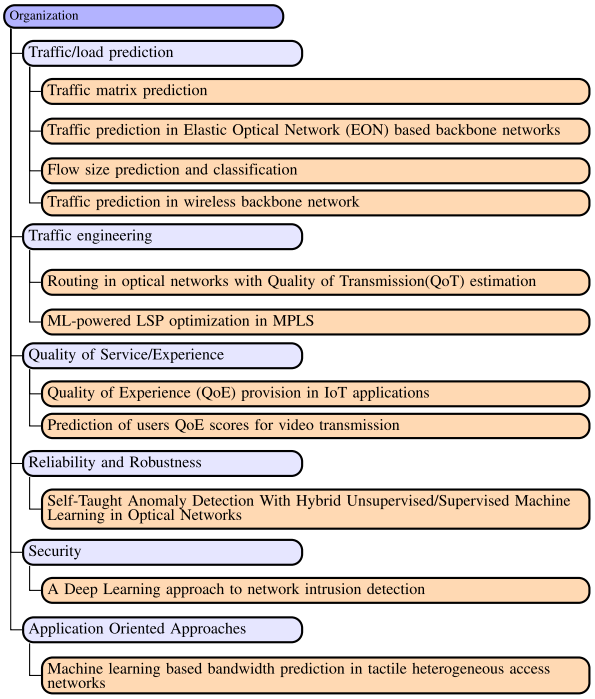


FIGURE 1. Organization of the tutorial.

In addition to network aspects, we also analyze the application-oriented approach, i.e., how ML techniques are used to boost the deployment of emerging network applications. Such an orthogonal approach complements the survey with particular use-cases and practical implementations.

As the aim of our work is to provide an example-based tutorial, we first carefully selected works to be thoroughly analyzed in the context of each aspect. The most important criteria of the selection process was to ensure sufficient tutorial material and cover a wide variety of NN structures, network types (e.g., access, core, metro) and technologies (e.g., wireless, optical, IP). Finally, we selected papers

TABLE 1. Summary of related tutorials and this paper.

Paper	Year	Network type	Network technology	Aspect of communications and networking
[2]	2019	Access	Wireless	Physical layer optimization, management and design
[3]	2019	Access and Edge	Wireless	Usage of ANN for wireless network-based applications
[4]	2016	Access	Wireless	Localization, QoS, load balancing, security
[5]	2010	Access	Wireless	Cognitive radio, dynamic spectrum access, self-organizing networks
[6]	2021	Versatile	Versatile	congestion control, prediction, IDS, TE or QoS
		core, cloud		QoS, resource management, security
[7]	2019	Core	Optical	Failure management
[8]	2019	Access, metro,	SDN	traffic classification, TE, QoE
		core, cloud		QoS, resource management, security
[9]	2017	Core	Wireless mesh	Deep learning based routing
This	2023	Access, metro,	Wireless, wired (e.g. EON),	Traffic prediction, QoE, QoS, resource
paper		core, cloud	SDN, versatile solutions	provisioning, reliability, security, TE

recently published in renowned journals and widely cited in the literature. As a result, we expect that the proposed example-based approach will facilitate a deep and practical understanding of how NN can be leveraged to address problems in networking and communications.

We organize the remainder of this paper as follows. Section II presents the latest and the most valuable tutorials regarding NN applied to networking and communications. In Section III we provide a tutorial on NN. In consecutive subsections of Section IV we analyze research papers applying NN to handle problems in different aspects of communications and networking. In addition to the network aspects, in Section V, we present an application-oriented NN-based approach facilitating the deployment of emerging network applications. The content of Sections IV and V is visualized in Figure 1. Section VI comprises a comprehensive comparative analysis of all works analyzed in this tutorial. Furthermore, it reports lessons learned and formulates summarizing insights and proposes research opportunities for the future. Section VII concludes the paper.

II. RELATED TUTORIALS

The most common approach to organizing tutorials and surveys in the field is to select a particular problem or networking aspect, and then briefly describe the fundamentals of the chosen topic and all the relevant works. This tutorial provides an in-depth technical and quantitative description of various NN methods for communication and networking. As per our knowledge, there is no other tutorial that simultaneously covers a wide variety of networking aspects and comprehensive analysis of NN techniques in an example-based tutorial. In this section, we will focus on the papers presenting an expanded tutorial part on NN.

Work [2] is a tutorial paper that covers NN and Deep Learning (DL) usage in wireless networks. The NN fundamentals together with problems and use cases are introduced deeply. The author surveyed works covering topics of physical layer operations, such as data detection, decoding, channel estimation, and physical layer resource allocation. The paper is somewhat similar; however, it focuses only on algorithms used in wireless networks, which limits its scope. [3] is a tutorial similar to the previous one and covers a variety of NN solutions dedicated to wireless applications such as Unmanned Aerial Vehicle (UAV), Virtual Reality (VR), Self-Organizing Network (SON). SONs are networks that can automatically plan, configure, manage, optimize, and heal themselves without human intervention. They use artificial intelligence, predictive analytics, and pre-optimized software algorithms to achieve various functions such as self-configuration, self-optimization, self-healing, and self-protection. They aim to simplify and improve the performance and security of mobile radio access networks. The paper scope is limited in terms of NN methods, focusing mostly on Artificial Neural Network (ANN). The paper is well organized, and the theoretical aspects are described deeply. Other examples of a wireless network-related papers are [4] and [5].

In addition to the tutorials focusing solely on a selected aspect of wireless communications, there are also some more general works. It is typical to analyze several aspects of computer networks from the perspective of selected neural network methods. Some recent examples are [6], [7], [8], and [9].

Paper [6], firstly, provides a valuable tutorial regarding the ML. Next, the survey part of this work covers a wide range of network types and technologies (e.g. multilayer, optical and 5G networks or IoT architectures) as well as communication and networking aspects (e.g. congestion control, prediction, IDS, TE or QoS). The main difference between [6] and our paper comes from the fact that we extend the general tutorial based on the particular examples that we carefully analyze. Contrary, authors of [6] follow a typical approach and provide a survey comprising numerous works but without in-depth analysis.

Work [7] is a tutorial paper investigating the area of optical networks. The authors provide an in-depth theoretical introduction and an extensive analysis of the selected articles. However, the tutorial's scope is very narrow and covers only failure management for optical networks.

Survey work [8] is written in a tutorial manner and focuses on applying NN techniques to Software-Defined Networking (SDN). The SDN paradigm utilizes central control to open

broader perspectives for optimization thanks to the global knowledge about the network. SDN also raises additional challenges, for example, scalability, reliability, or ability to handle a vast amount of data. Work [8] focuses on aspects similar to our tutorial (routing, QoS, traffic prediction); however, it does not provide guidelines on how to use particular NN methods in the analyzed problems.

Article [9] covers a variety of NN and DL methods applied in numerous areas of computer networks, i.e., wireless networks, network traffic prediction and classification, and SONs. The theoretical fundamentals are carefully explained in the tutorial part. However, the details on application ML methods to the problems are described only superficially.

Related tutorials are summarized in Table 1. To sum up, the most common approach is to select a network type, technology, an aspect of networking or even a particular problem and briefly describe all the relevant works. To sum up, there is no work of a truly tutorial manner that analyses the application of NN methods in a variety of communications and networking aspects based on the selected examples.

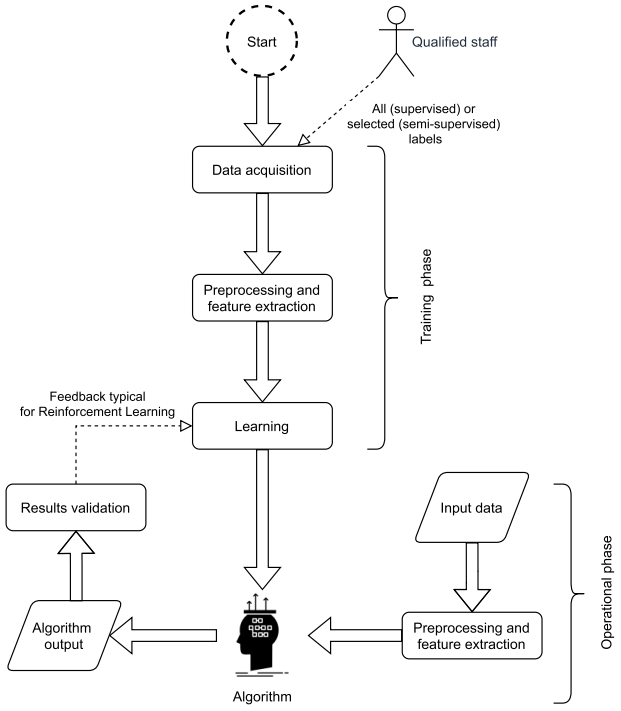


FIGURE 2. General workflow of ML methods.

III. MACHINE LEARNING

Machine Learning was first defined by Arthur Samuel in 1959 as "the field of study that gives computers the ability to learn without being explicitly programmed". Although ML has been significantly developed since then, this basic definition remains valid. The workflow of each ML technique follows the same general pattern presented in Figure 2. Firstly, input data (denoted also as training data) must be gathered, generated or defined to feed the algorithm. Then, during the feature extraction stage, noise from the raw data is removed, its dimensionality is reduced, and critical features are emphasized. The aim of this preprocessing is to reduce computational overhead and increase accuracy. Afterwards, the algorithm learns based on the complex dependencies representing the considered problem. One must note that precise interpretation of data collection, feature engineering and model learning depends on the learning paradigm used by the particular ML method. The above-mentioned steps constitute the process of model creation.

An algorithm prepared in this way can operate on new data to provide the outcome. The interpretation of the outcome and potential preprocessing of new data depends on the considered problem. The outcome may be simple information for a person, but it can also trigger further actions. Finally, based on the outcome validation it is possible, but not obligatory, to adjust and improve three initial phases: data collection, feature engineering and model learning. The presented workflow suggests that ML algorithms are most suitable for problems for which a significant amount of representative data is available [10].

The Fundamental taxonomy of ML algorithms originates from the learning paradigm used by a particular method. The characteristics of possible approaches are briefly presented below as an introduction to the more detailed tutorial on NN.

The first class of ML methods is called supervised learning and includes neural networks. The supervised learning approach requires the training data set to be supplemented with additional labels explaining how to interpret each sample. An example of a typical output of the supervised learning method is discrete information that classifies a sample of new data to one of the predefined categories (e.g., car or bike). However, supervised learning methods may also perform regression to provide a solution in a continuous space.

The second class of ML methods, unsupervised learning (also includes NN), brings a significant advantage as it does not require additional labels in the training set. Unsupervised learning methods analyze the internal structure of the data and perform clustering or association. Clustering means that groups of similar samples are created (e.g., clients of a video service). Association finds out the rules that apply to the data (e.g., people interested in a particular movie, usually also like others from the group).

The semi-supervised learning joins both supervised and unsupervised approaches. Firstly, a small set of training data supplemented with additional labels is used to achieve a partially trained algorithm. Secondly, the initial model processes a large set of unlabeled training data. The aim is to add labels and transform the set into so-called, *pseudo-labelled* data. Finally, labelled and *pseudo-labelled* are together used to train the supervised learning method. Thus, semi-supervised learning limits the need for big sets of labelled training data. This approach is advantageous in cases

when the labelling process is demanding (e.g., advanced classification of videos).

Finally, in the reinforcement learning paradigm, the ML algorithm is represented by an entity called an agent. The agent learns about the uncertain and complex environment (the problem) by interacting with it. The interaction means taking an action for which the algorithm receives rewards or penalties. The goal of the agent is to maximize the total reward by taking the proper sequence of decisions (actions). Thus, there is no supervision, but instead the reward signal. Each action directly affects which information the agent will get. The main advantage of RL is that it can learn from the vast number of parallel sequences of decisions. Powerful computing infrastructure enables this opportunity. Thus, reinforcement learning algorithms are best suited for game-like scenarios where immediate decisions are needed. The flagship example is the control of the autonomous drone.

A. ARTIFICIAL NEURAL NETWORKS

ANN [11], which is usually referred to as NN, is one of the most crucial and most widely used ML method. The importance of ANN is clearly visible in this tutorial, where multiple papers take advantage of various ANN architectures. Therefore, it is more than reasonable to briefly explain this concept. ANN is composed of artificial neurons (derived from the biological neurons existing in the human nervous system) and connections among them. Connections provide the output from one neuron as an input of another neuron with an assigned weight which represents the importance of the connection. It is also worth noting that each neuron may have multiple input and output connections. Neurons are organized into multiple layers. The first layer, denoted as the input layer, receives the input data which is further processed, whereas the output layer is responsible for producing the result. Between the input and the output layers, there are zero or more hidden layers that are responsible for complex data processing. To find the output of a single neuron in the hidden layer, all its inputs and their weights are combined in the form of a weighted sum - equation (1). During the model training, weight values are updated according to the selected method, with one of the most popular methods being backpropagation - error in the output is back-propagated through the network and weights are adjusted to minimize the error rate. The weighted sum is sometimes referred to as the activation. Based on it, the activation function is then able to produce the output of the neuron. It is worth to note that activation functions are mostly non-linear, for example sigmoid or hyperbolic tangent functions. Additionally, it must be stated that activation functions reside only in the hidden and output layers. There are multiple ANN architectures, however here we will focus only on the most crucial and basic ones.

$$z = \sum_{i=1}^{n} w_i \cdot x_i + b \tag{1}$$

where:

z - the weighted sum of the neuron,
n - the number of neuron inputs,
w_i - the weight of the i-th input,
x_i - the value of the i-th input,
b - the bias of the neuron.

1) FEEDFORWARD NEURAL NETWORK

Feedforward Neural Network (FFNN) is the first (in terms of being devised) and the simplest architecture of ANN. In FFNN, the connections between the nodes (neurons) do not form a cycle, i.e., the architecture does not contain feedback loops. In other words, the flow of information is forwarded only in one direction, from the input layer to the output layer through the optional hidden layers. The architecture of simple FFNN is shown in Figure 3. Usually, when authors mention NN or ANN, they mean FFNN.

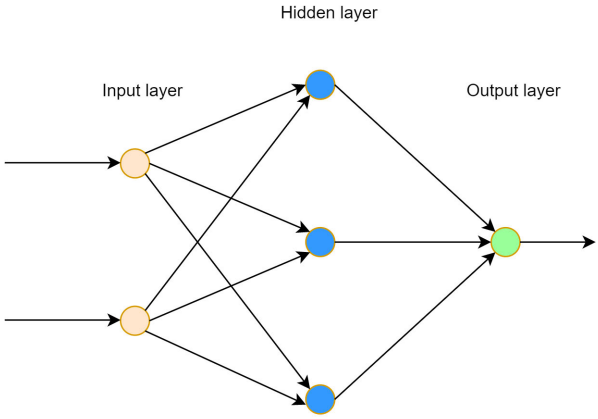


FIGURE 3. Feedforward Neural Network architecture.

2) DEEP NEURAL NETWORK

ANN with multiple hidden layers between the input and the output layers is called as Deep Neural Network (DNN), regardless whether it is FFNN (Deep FFNN) or Recurrent Neural Network (RNN) (Deep RNN). Therefore, the greater the number of hidden layers, the deeper the network. The architecture of DNN is shown in Figure 4. Although some definitions exist, there is no definite answer to the question of how many layers are needed to qualify the network as deep. Additionally, it is very hard to answer the question regarding the number of hidden layers that should be used for the given problem. With the growth of the network size (more hidden layers or more neurons in a hidden layer), its accuracy rises. However, besides the accuracy aspect, the network's performance will be affected negatively, and the network will be more prone to overfitting. Understanding network fit is important for understanding the root cause of poor accuracy. Overfitting means that a network is so complex that it learns the details and noise of the data to the extent that it is unable

to perform well on different datasets. On the other hand, smaller networks are more prone to underfitting. Contrary to overfitting, underfitting happens when a network does not perform well on any dataset. Very often, the number of hidden layers is selected based on either intuition or in an experimental way where neural networks of different sizes are evaluated. Deep neural networks may require a distributed approach, as shown in [12].

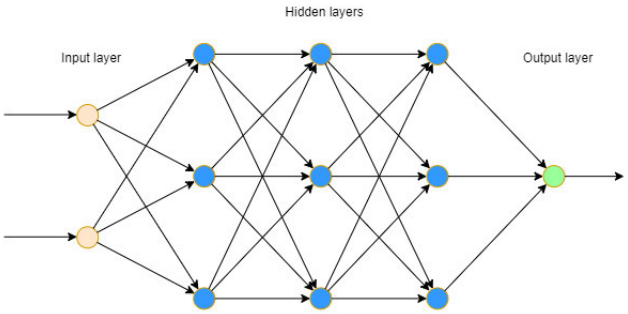


FIGURE 4. Deep neural network architecture.

3) RECURRENT NEURAL NETWORKS

RNN is another example of ANN [13]. As shown in Figure 5, RNN has a recurrent connection to the neurons in the hidden layer. Output from the neuron is both passed to the next layer and fed back to its input (recurrent layer) to ensure that the sequential information is captured in the input data - mathematics behind the neuron state is different for each of the RNN architectures such as Long Short Term Memory (LSTM) or Gated Recurrent Unit (GRU). Equation (2) describes the neuron state at time t for most basic RNN architecture.

$$h_t = \text{activation}(W_{hh} \cdot h_{t-1} + W_{xh} \cdot x_t + b_h) \tag{2}$$

where:

h_t - state (output) at time step t,
h_{t-1} - state at the previous time step $(t-1)$,
x_t - input at time step t,
W_{hh} - weight matrix between states in previous time steps,
W_{xh} - weight matrix between inputs and state,
b_h - bias vector,
$activation(\cdot)$ - activation function.

While deciding, RNN takes into consideration the current input and also what it has learned from the inputs received previously. This type of NN is mainly used when a context is critical to predicting an outcome, or in other words, when decisions from past iterations or samples can influence current ones. There are also many variants of RNNs:

- one-to-one (single, fixed size input & single, fixed size output),
- one-to-many (single, fixed size input & sequence of data outputs),
- many-to-one (multiple inputs or sequence of inputs & single, fixed size output),
- many-to-many (sequence of inputs & sequence of outputs).

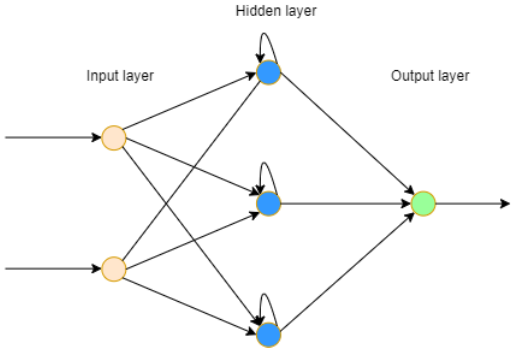


FIGURE 5. Recurrent neural network architecture.

4) CONVOLUTIONAL NEURAL NETWORKS

Convolutional Neural Network (CNN) is a class of DNN. Its name is related to the mathematical linear operation on matrices called convolution. As it is shown in Figure 6, CNN can be split into two parts. The classification part is most often a fully connected layer (regular ANN) that utilizes the output from the feature extraction part and predicts the class of the input based on the features extracted in the previous stage. Feature extraction part introduces new layers: convolutional and pooling. The convolutional layer is the first and core layer of CNN that extracts features from input data. In the convolutional layer, the input data is convolved with a kernel (or filter) - equation (3). A kernel is a small matrix, with its height and width smaller than the input data to be convolved. It is also known as a convolution matrix or convolution mask. Output from the convolutional layer is often referred to as a feature map. The goal of the pooling layer is to reduce complexity for further layers by down-sampling (reducing the size of the input data while preserving the important characteristics). In most cases, the pooling layer is placed after the convolution layer. There are several types of pooling operations, with the most popular being Max Pooling. Max Pooling partitions the input matrix into a set of matrices, and for each of the matrices, the highest number is treated as an output. Max Pooling is described as a mathematical formula (4). Among other types of pooling operations, Average Pooling would calculate the average for each of those matrices, and Sum Pooling would calculate the sum for each of those matrices. The examples of usage of CNN are presented in [14] and [15].

$$Y_i = (X * K)_i = \sum_{j=1}^{m} \sum_{k=1}^{n} X_{i+j-k} \cdot K_{j,k} \tag{3}$$

where:

Y_i - i-th output element of the convolutional layer,
X - input data to the convolutional layer,
K - convolutional kernel (filter),
m, n - the dimensions of the convolutional kernel.

$$Y_{i,j} = \max(X_{2i,2j}, X_{2i,2j+1}, X_{2i+1,2j}, X_{2i+1,2j+1}) \quad (4)$$

where:

$Y_{i,j}$ - i, j-th output element of the pooling layer,
X - input data to the pooling layer,
$X_{2i,2j}$ - input element at position $(2i, 2j)$,
$X_{2i,2j+1}$ - input element at position $(2i, 2j + 1)$,
$X_{2i+1,2j}$ - input element at position $(2i + 1, 2j)$,
$X_{2i+1,2j+1}$ - input element at position $(2i + 1, 2j + 1)$.

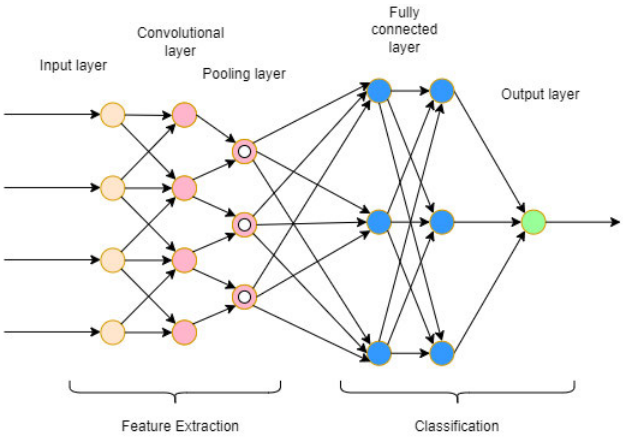


FIGURE 6. Convolutional neural network architecture.

5) DEEP BELIEF NETWORK

A Deep Belief Network (DBN) is a kind of deep learning network formed by stacking several Restricted Boltzmann Machines [16]. Boltzmann Machine is a two-layer network of fully connected units whose representation is shown in Figure 7. What is clearly visible is the fact that there is no output layer. Units in the visible layer are treated as the units in the input layer as they receive the data, while units in the hidden layer are responsible for capturing dependencies between the visible variables. Nodes from the same layer are connected to each other. As the name implies, Restricted Boltzmann Machine (RBM) is a variant of Boltzmann machine with a small difference, there are no connections between nodes within a layer.

In DBN each RBM performs a non-linear transformation on its input vectors and produces output vectors that will serve as input for the next RBM in the sequence. In other words, the hidden layer of the RBM is a visible layer for the next RBM. An example of a three-layer DBN is

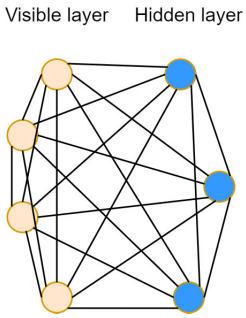


FIGURE 7. Boltzmann machine representation.

depicted in Figure 8. DBN is a hybrid generative model which has undirected connections between the top two layers and directed acyclic connections between all its lower layers. The undirected connections form the associative memory, while the directed connections point toward the layer that is closest to the data (visible layer) and convert associative memory to observed variables. Although utilizing the concept of NN, DBN are classified as an unsupervised algorithm, since they are composed of unsupervised networks, like RBM.

DBN training process consists of two main steps: pre-training and fine-tuning. In the pre-training phase, a greedy learning algorithm is used. RBMs are trained independently and the output of the lower RBM is mapped to the input of the next RBM, until all RBMs are trained. After the pre-training phase, the fine-tuning is performed. Fine-tuning improves the accuracy of the model by finding the optimal values of the weights between layers.

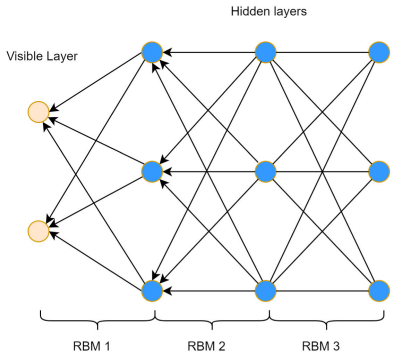


FIGURE 8. A three-layer Deep Belief Network.

One of the first challenges when leveraging NN to solve complex problems in the domain of communications and networking is to select the most prominent NN type for a particular issue. Exact and versatile recommendations for the selection process cannot be formulated due to the complex nature of the problems in this domain. However, we formulated some general insights to facilitate leveraging NN to solve a wide range of problems:

- FFNN: effective for tasks such as classifying network traffic, detecting anomalies, or identifying services and protocols.

- RNN: instrumental in recognizing sequential patterns in network traffic and handling time-related problems, such as: predicting future network loads or latency.
- CNN: excel at pattern recognition, making them valuable for analyzing network graphics and detecting visual patterns.
- DBN: useful for data dimensionality reduction, feature extraction, and data clustering in network analysis.

It's worth noting that these neural network architectures can be combined or configured in various ways to tackle more complex network-related issues. To apply the proper method, one should consider the type of input data, the characteristics of the problem, and the desired output.

B. MACHINE LEARNING LIBRARIES

Machine learning models, including neural networks, can be trained and executed in any language offering decent support for numerical computations. Having said that, the key enablers to make it for a successful ML library are automatic differentiation and support for dedicated accelerators (Graphical Processing Unis (GPU) or ML dedicated hardware (TPU)). Two of the most popular libraries are PyTorch [17] and TensorFlow [18]. Both libraries use python programming language, which is the most popular for machine learning applications.

PyTorch, developed by Facebook, is an open-source ML toolkit. It is designed from its beginning as an eager and imperative library. Because of that, it perfectly fits into the python ecosystem. PyTorch is extremely user-friendly, so it is popular for beginners in the area of machine learning. It also seems more popular among researchers, so it is highly probable to find a new result implemented on top of PyTorch. Just like TensorFlow, PyTorch offers efficient numerical routines on CPU and GPU, automatic differentiation, and in a recent version JIT compilation of a python-based domain-specific language called TorchScript.

TensorFlow, developed by Google, is another open-source machine learning framework. The very early versions contain numerous flaws, however, since 2015 the software matured a lot and the transition to version 2.0 solved most of the issues of the 1.x versions. TensorFlow offers an eager style of execution that is easy to debug and very similar to the very popular library NumPy. Furthermore, recent versions of TensorFlow offer an experimental implementation of NumPy API. When it comes to performance, TensorFlow enables offloading of computation to the runtime and executing on specialized hardware (like GPU or TPU). Further performance gain can be obtained by JIT compiling computations with XLA- a linear algebra compiler developed for TensorFlow.

Both PyTorch and TensorFlow are high-quality machine learning frameworks enriched by specialized libraries, like for example for Graph Neural Networks. Thus, the final decision mostly depends on personal preferences. In our opinion, PyTorach is a better solution to test new research ideas, while TensorFlow is advantageous in the case of deployment of ML-based solutions on a production system.

A new experimental library called JAX [19] is worth mentioning, as it seems to be something between the two above. JAX is a library offering composable function transformation like gradient or jit developed by Google using components of TensorFlow (XLA). It is not an official Google product yet, however, we can observe massive use of JAX within DeepMind. We expect that the new reinforcement learning algorithms will probably be implemented in JAX.

IV. SELECTED ASPECTS OF COMMUNICATIONS AND NETWORKING

In consecutive subsections, we analyze research papers addressing different aspects of communications and networking. We carefully selected limited representative papers that deal with a particular network aspect and provide comprehensive analysis for each of them. The main goal is to ensure a deep understanding of NN and their applications proposed in the consecutive papers. Additionally, for each aspect we provide useful summaries and references to other aspects, providing a reader holistic and comprehensive knowledge.

A. TRAFFIC/LOAD PREDICTION

Accurate traffic prediction may be crucial for energy savings and network resource management. A significant amount of power can be saved if traffic forecasting is used in core network equipment. Accurate forecasting may also improve QoS - accurate prognosis of network traffic enables efficient resource allocation. Furthermore, the detection and prevention of network abuse can be directly linked to traffic prediction accuracy - significant deviation from predicted traffic behavior can be used to detect attacks in their early stages. In this section, we analyze solutions that utilize NN mechanisms for various traffic prediction-related problems. Firstly, we analyze paper [20] which addresses the traffic matrix prediction problem by adapting LSTM RNN. After that, traffic forecasting in cloud data centers based on Elastic Optical Network (EON) technology is presented in [21]. Next, flow size prediction and classification are shown in [22]. At the end of this section, worth noticing network traffic decomposition and estimation are presented in [23]. The structure of this section is presented in Figure 9.

1) TRAFFIC MATRIX PREDICTION

Paper [20] aims to predict network traffic for large networks. More specifically, the authors propose a network Traffic Matrix (TM) prediction framework called NeuTM which is based on DL system. As defined by authors in problem statement - TM presents the traffic volume between all pairs of Origin and Destination (OD) nodes of the network at a certain time. Precise TM can strongly benefit many network operation and management tasks for example traffic scheduling or re-routing. We find it interesting that instead of network traffic prediction in terms of aggregated traffic

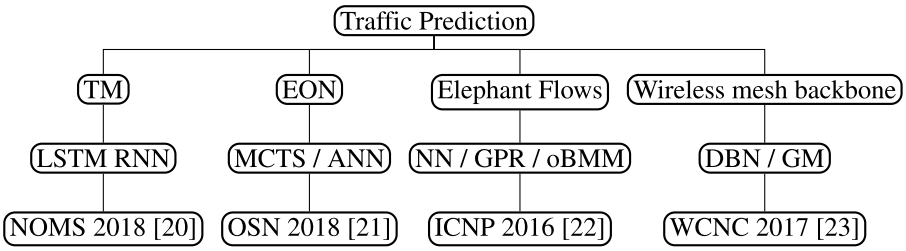


FIGURE 9. Structure of the Section IV-A.

volume for the entire network, authors propose a framework for network TM prediction, and we agree with the authors that it is a more challenging task.

Self-similarity, multi-scalarity, long-range dependence, and highly nonlinear nature characterize the network traffic - its prediction methods can be divided into two categories:

- linear prediction
- nonlinear prediction

The authors provide a summary of the aforementioned categories. Firstly, they describe three time-series linear predictors based on traditional statistical techniques. These predictors include:

- Autoregressive-moving-average model (ARMA)
- Autoregressive integrated moving average (ARIMA)
- Holt-Winters (HW)

Background regarding linear predictors was followed by a brief description of NN for time-series prediction and a summary of important problems and limitations regarding FFNNs. Since linear prediction methods do not meet the accuracy requirement and FFNNs have limitations, authors propose LSTM RNN as a mechanism for TM prediction.

Authors provide good background regarding LSTM RNN where each component of the architecture is described and a set of pre-processing LSTM equations is defined. LSTM network uses the provided equations iteratively to compute the network unit activations. Based on those computations, LSTM maps an input sequence to an output sequence.

TM prediction problem is formulated as the problem of forecasting future network traffic matrix based on the previous and achieved network traffic data. To solve this problem, the widely known gradient-based method back-propagation through time algorithm (BPTT) is used to train the LSTM architecture. To effectively feed the LSTM, authors propose a traffic matrix and its transformation into a traffic vector. Meanwhile, the notion of a learning window is introduced. The learning window represents a fixed number of previous time slots to learn from to predict the current traffic vector. The introduction of this notion is feasible because after some time the total number of time slots becomes very high and thus high computational complexity is necessary. The learning window overcomes this problem as limiting the number of time slots is directly limiting required computation complexity. Having a learning window and traffic vector, the authors construct the final traffic-over-time matrix which is used for training and validation. We present all those steps in Figure 10.

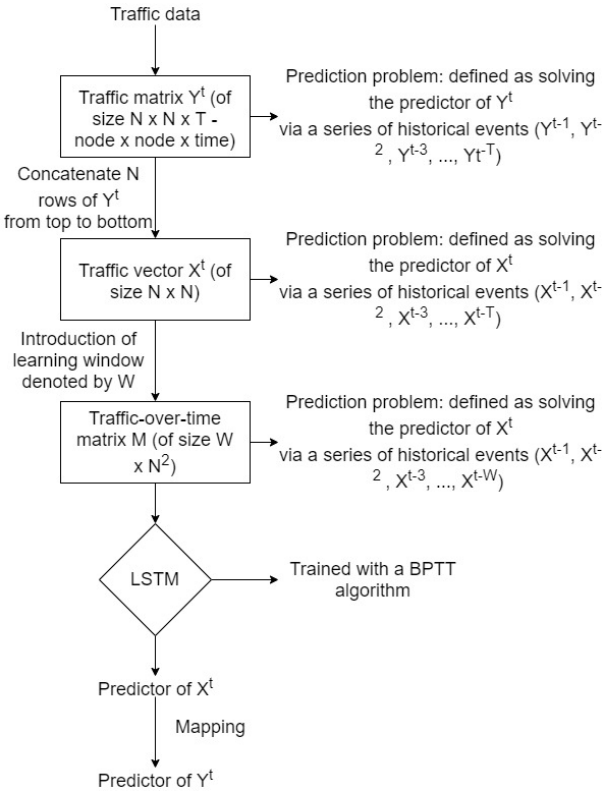


FIGURE 10. Representation of algorithm presented in [20].

Real traffic data from the GÉANT backbone network made up of (as of 2004) 23 nodes interconnected using 38 links was used to evaluate the prediction accuracy of NeuTM - more specifically, 2004-timeslot traffic matrix data sampled from the GÉANT network [24]. Authors evaluated their method on short-term TM prediction as only a set of 309 traffic matrices was considered. Traffic-over-time matrix was split into training and validation matrices, where the training matrix was used to train the LSTM model, and validation matrix was used to evaluate and validate its accuracy. Well-known Mean Square Error (MSE) was used to estimate the prediction accuracy to quantitatively rate the performance of the proposed LSTM model. We group all simulation parameters and provide them in Table 2.

Through simulations, the prediction error of LSTM was compared to other methods mentioned in the paper (ARMA, ARIMA, HW and FFNN). It is worth mentioning that the prediction results of the linear predictors and FFNN were obtained from [25] and they represent the error of predicting only one traffic value and not the whole traffic matrix. We believe that it would be more suitable to compare the prediction error of this framework with other methods of solving traffic matrix prediction problem and not the problem of only one traffic value prediction. Results also show that the proposed model converges quickly - after 3 hidden layers (each containing 500 nodes) prediction error does not fall significantly. Apart from the accuracy measurements authors also measured the training time required for different network depths. It is worth noting that this high prediction accuracy was provided in a short training time, as it takes less than 5 minutes to train a 6-layer network on 20 epochs.

TABLE 2. Simulation parameters in [20].

Parameter	Value
Network topology	23 peer nodes / 38 links
Traffic-over-time matrix	309 x 529 matrix
Training matrix	263 x 529 matrix
Validation matrix	46 x 529 matrix
Number of hidden layers in LSTM	1/2/3/4/5/6
Nodes in each hidden layer	500

This paper shows that LSTM RNN can be successfully used with SDN to very accurately forecast TM. In our opinion, the proposed solution can be successfully used for network operations and management tasks improvement like QoS improvement and anomaly detection, efficient scheduling, or re-routing in other large networks. We also believe that to prove the true superiority of LSTM for traffic matrix prediction additional studies are required. Further, future work might be an extension that would compare NeuTM with other solutions that solve traffic matrix prediction problem.

2) TRAFFIC PREDICTION IN EON-BASED BACKBONE NETWORKS

Paper [21] focuses on traffic prediction in cloud data center (DC) networks using EON as a backbone transport technique. Currently, optical networks use Wavelength Division Multiplexing (WDM) as a backbone transport technique. WDM is used to communicate nodes in the optical layer using all-optical WDM channels. Even though WDM technique has many benefits, they are limited by the fixed-sized optical bandwidth allocation. EON technique copes with this limitation as it allocates appropriate-sized optical bandwidth to an optical path - in other words, the optical path expands according to the traffic volume. In this paper authors present two approaches for traffic prediction that employ ML techniques - Monte Carlo Tree Search (MCTS) and ANN. As described by the authors in the problem statement -ML technique is used to identify the best combination of cloud DCs and candidate path pairs for provisioning the services related to specific requests. We find this specific provider-oriented use case interesting and we agree with the authors that currently, when referring to the cloud DCs, everything is measured by virtual resources and payable per hour of using it.

The problem description starts with the introduction of a detailed network model. We provide a summary of the network model components together with their brief description in Table 3. After the network model introduction, the objective function and set of constraints are defined. The objective function is expressed as the minimization of the average request blocking percentage. Average request blocking percentage is the ratio (expressed in percentage) between the rejected requests for DC resources and all requests offered in a network.

TABLE 3. Network model summary for [21].

Component	Description
Optical network	Modeled as a graph comprising: - set of vertices (nodes) - set of directed edges (fiber links) - maximum number of frequency slices accommodated by each link - link lengths
Cloud DC	Characterized by five parameters: - computation units (CPU units) - memory units (RAM units) - disk space (storage units) - expense of use (USD/hour, provided by Amazon Web Services) - location (provided by Amazon Web Services)
Traffic requests	Multiple request types are considered: - unicast (client to client) - multicast (client to client) - anycast (to and from DCs) - upstream - downstream
EON	Various modulation formats may be used Modulation formats are selected so as to minimize number of regenerators used in network Distance-adaptive transmission rule from [26] is used for spectrum requirement determination Physical model as in paper [27] is used Transmission model as in paper [28] is used 12.5 GHz guard band for neighboring connections Regenerators used whenever necessary

In the first approach, MCTS algorithm is implemented to enable traffic prediction. To minimize the requests blocking percentage, the best pair of DC and candidate path pair must be selected. ANN is selected for the second approach. Training of ANN was executed in three phases:

- data preprocessing (assuring that inputs are in comparable range using normalization),
- selection of significant inputs (link utilization, DC utilization and information on how the network utilization

was affected by the particular routing decisions in previous iterations),
- dimensioning of the hidden layer.

In ANN the backpropagation algorithm was selected to train the dataset with modeled data.

It is also worth noticing that the authors address the runtime complexity for both of the proposed traffic prediction approaches. We summarized the ANN approach in the block diagram presented in Figure 11.

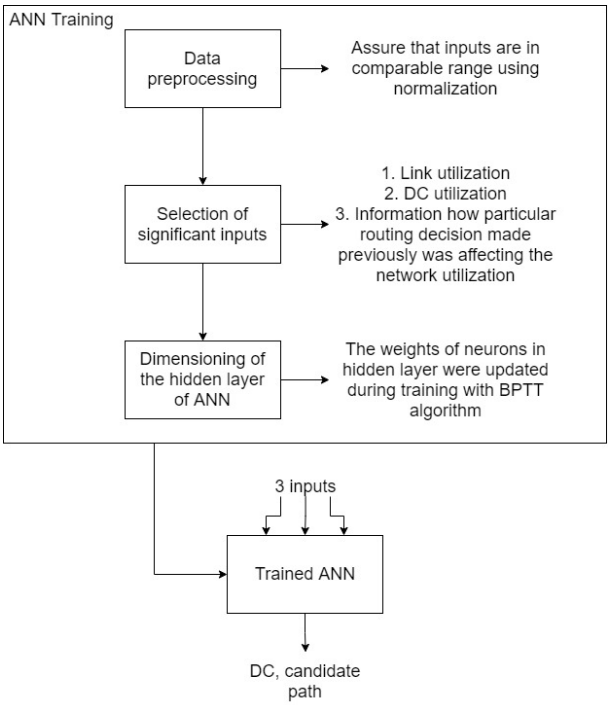


FIGURE 11. Representation of MCTS based algorithm presented in [21].

The experiments carried out by the authors were extensive and then thoroughly analyzed. The superiority of presented mechanisms is proven by simulations where three Routing, Modulation and Spectrum Assignment (RMSA) algorithms from literature are used with enabled and disabled traffic forecasting. Simulations were performed on two different networks with an optical layer using EON with Bandwidth Variable Transponder (BV-T) and PDM-OFDM (polarization division multiplexing of orthogonal frequency division multiplexing) technology with multiple modulation formats selected adaptively. Since authors do not give background knowledge regarding BV-T and PDM-OFDM we provide it here.

- BV-T - A lightpath (optical path) is established from BV-T at the source node to BV-T at the destination node. BV-T can elastically accommodate diverse traffic connections by generating an appropriate number of OFDM sub-carriers. BV-T can also adjust the transmission ratio of each OFDM sub-carrier. Therefore, an optical signal is generated by using just enough sub-carriers with an appropriate modulation level and an appropriately sized end-to-end optical path.
- PDM-OFDM - Using PDM permits multiplying the user capacity and increasing the spectral efficiency. The combination of PDM and OFDM allows maximizing the optical transmission capacity.

Additionally, simulations were performed for different traffic load values. We summarize network parameters that were used in experiments in Table 4. EON technology can be expressed by multiple parameters: total available spectrum, number of slices (optical channels), modulations formats that support it, and many others - those parameters are grouped in Table 5. It is also essential to mention that apart from traffic requests that were shown in Table 3 authors distinguish request types even further - each having different parameters (bit-rate, flow types, DC resources used) - we also provide a summary of the traffic model used in simulations in Table 6.

TABLE 4. Network simulation parameters for [21].

Network	Nodes	Links	DCs location	others
Euro28	28	82 unidirectional	centralized	3 intercontinental links
US26	26	84 unidirectional	east/west coast	3 intercontinental links

TABLE 5. EON simulation parameters for [21].

Spectrum	Slices	Modulations	BV-T bit-rate	Others
4 THz	320, each 12.5 GHz	BPSK QPSK 8-QAM 16-QAM 32-QAM 64-QAM	40 Gbps 100 Gbps 400 Gbps	regenerators used if needed

TABLE 6. Traffic model used for simulations in [21].

Request type, % of all traffic	Bit-rate	Flow Types	DC resources (max)
Processing as a Cloud, 17.8	10-200 Gbps	anycast unicast	16 CPU 64 GB RAM
Storage as a Cloud, 5.6	100-400 Gbps	anycast multicast unicast	1 TB storage
Software as a Service, 58	10-100 Gbps	multicast unicast	8 CPU 32 GB RAM 100 GB storage
Optical as a Service, 18.6	100-400 Gbps	anycast	4 CPU 4 GB RAM 4 TB storage

The simulations show that both proposed ML mechanisms improve the performance of all RMSA algorithms. Generally,

MCTS approach is slightly better when it comes to the allocation of the data center traffic (meaning lower blocking percentage), especially for higher traffic load. However, MCTS is not as cost-efficient as ANN approach. Cost-efficiency of both approaches was comparable to the methods with disabled ML prediction techniques - which shows their superiority even more. In addition to the blocking percentage and cost-efficiency comparison, authors also evaluate decision time (time of performing calculations) with and without proposed mechanisms. Evaluation of decision time shows that the time of calculations was not increased remarkably when using proposed mechanisms.

In our opinion, [21] is a solid work showing that both ANN and MCTS methods can be successfully used for traffic prediction in DC networks using EON technique in the optical layer. We believe that such forecasting approaches may significantly improve efficiency and lower costs for telecommunication operators.

3) FLOW SIZE PREDICTION AND CLASSIFICATION

In [22], the authors address the problem of predicting the size of a flow and detecting very large flows, referred to as elephant flows. Currently, two flow size estimation techniques are considered: those that modify applications or end hosts and those that do not. It is not always possible to modify applications or end hosts, on the other hand, approaches that do not do this are not ideal as they permit re-routing and re-scheduling only after detecting elephant flows which were present for some time. Therefore, too late detection of elephant flows may cause congestion. Being aware of limitations coming from reactive monitoring techniques that only detect elephant flows after they have been flowing for a while authors propose data mining to estimate the size of each flow and to detect elephant flows as they start. Using features of the few initial packets ML based flow size estimators can predict the size of each flow at the start. The accuracy of three online predictors is evaluated in [22], these predictors are based on NN, Gaussian Process Regression (GPR), and online Bayesian Moment Matching (oBMM). Apart from the accuracy of predictors, [22] also presents how routing can be improved when proposed predictors are used with SDN - what in our opinion is an interesting supplement for flow size prediction results.

The goal is to accurately estimate the size of each flow and detect elephant flows as they start. Therefore, a function that takes specific flow features as input and returns an estimated size of each flow is discussed by the authors. The authors consider seven features that can be easily read from the first few packets of each flow. We summarize those features and their description in Table 7.

As mentioned before, three ML techniques are used for online flow size prediction. The first approach was based on NN consisting of:

- input layer with 106 binary nodes (corresponding to the features described in Table 7 transformed into +1/-1 values),
- two hidden layers (60, 40 hyperbolic tangent nodes respectively),
- output layer with 1 node.

TABLE 7. Features used for flow size prediction in [22].

Feature	Description
source IP destination IP source port destination port	extracted from the header of first packet
protocol	TCP/UDP (extracted from the header of the first packet)
server vs client	if the protocol is TCP, feature indicates whether: - the flow is a request from client - the flow is a response from server if the protocol is UDP then this feature is absent
size	first three data packets

The value between -1 (mice) and 1 (elephant) was returned by the hyperbolic tangent activation function which was used for classification. GPR was the second considered ML technique. In this approach, a Gaussian distribution over the size of a flow is computed based on input features. The predicted size of a new flow is equal to the mean which was calculated using the equation provided by authors. oBMM was the third approach proposed in [22]. In this approach, elephant flows are predicted using Gaussian Mixture Model (GMM). The feature of each class is derived from a GMM using provided formulas where weight, mean, and covariance matrix are learned from data using oBMM. In the end, solving simple inequality with Bayes Theorem results in flow assignment to either elephant or mouse class. We summarized described above approaches in Figure 12

Prediction accuracy and efficiency of flow size predictions were evaluated on three different datasets with each dataset having packet traces for more than 3 million flows [29]. Two datasets included both TCP and UDP flows and one dataset included only TCP flows. Elephant and mice flows were classified according to 4 different threshold values. The authors address the problem of mice and elephant flow classes imbalance which comes from low elephant flow percentage. As stated, this problem may harm the training of a classifier. As a solution, the importance of elephant flow misclassification is increased and the importance of mice flow misclassification is decreased. We provide a summary of the mentioned prediction accuracy evaluation parameters in Table 8. During flow size prediction experiments two metrics were analyzed: true positive rate (percentage of correctly identified elephant flows) and true negative rate (percentage of correctly identified mice flows). Results show that oBMM and NN based predictors suffer from class imbalances as the threshold is increased. An exception is an experiment with 1st dataset where oBMM TPR and TNR are the same (100%) for all threshold values - we believe this may not be correct, however, these particular results are not discussed by the

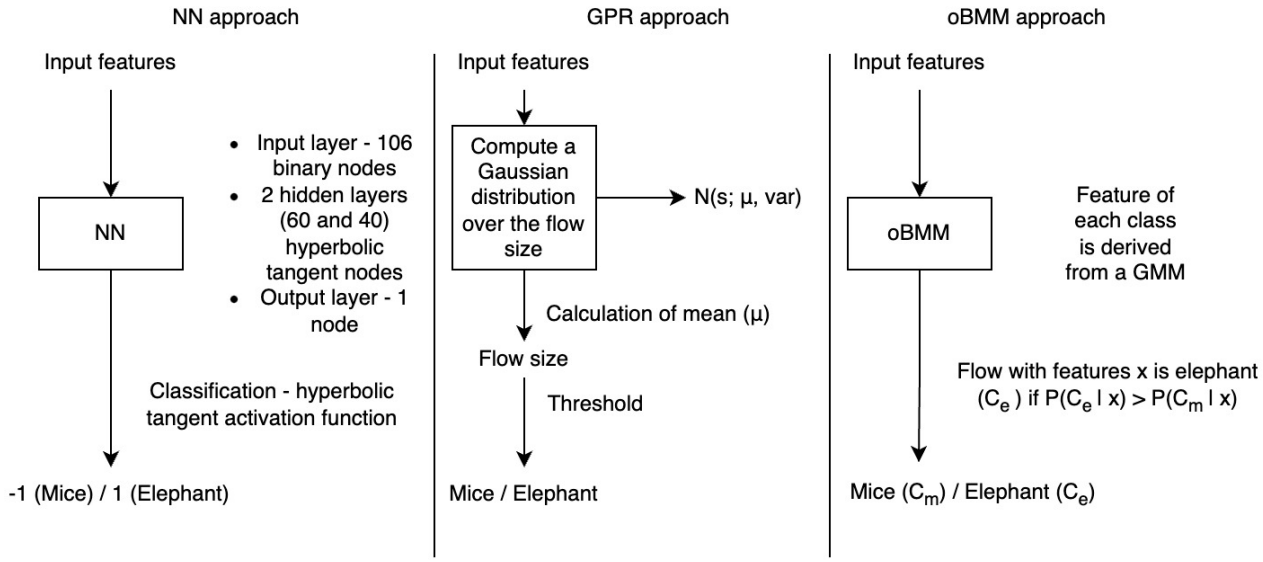


FIGURE 12. Representation of flow size prediction approaches in [22].

TABLE 8. Parameters for flow size prediction experiments in [22].

Dataset	Flows	Protocol	Threshold	Class imbalance mitigation
1	>3 million	TCP & UDP	10 Kb	elephant flows misclassification error is multiplied by a weight (which is inversely proportional to their percentage during training)
2	>3 million	TCP & UDP	31.623 Kb	
3	>3 million	TCP	316.228 Kb	
			1 Mb	

authors. Nevertheless, the GPR technique seems to be the most robust to class imbalances.

In routing experiments, authors propose a simple routing policy that routes elephant flows identified by a previously proposed ML based flow size predictor via the least congested path, and mice flows identified by the same predictor are routed by Equal Cost Multipath (ECMP) - presented in Figure 13. ECMP is a routing strategy where packet forwarding to a single destination can occur over multiple best paths with equal routing priority. ECMP can improve network performance by load-balancing traffic over multiple paths, but it may also face some challenges such as packet reordering, spectrum fragmentation, and physical impairments. Unfortunately, those routing experiments were not conducted with the NN used as a flow size predictor and therefore we will not cover this part in depth.

These experiments show the superiority of ML based flow size estimation mechanisms which take advantage of features carried by a flow's first few packets and can detect elephant flows at its start. Based on that information, such a flow can be almost effortlessly routed through a path selected by any desired policy. We believe that methods used for flow size forecasting shall be discussed more in-depth as that is a crucial part of this paper. In our opinion, this work proves

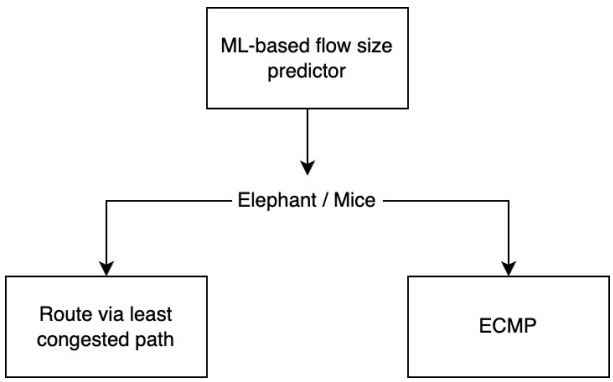


FIGURE 13. Simple routing policy proposed by authors in [22].

that flow size predictors based on ML can be used in other networks to reduce completion times and improve overall traffic engineering.

4) TRAFFIC PREDICTION IN WIRELESS BACKBONE NETWORK

In [23], the main goal is to predict the network traffic in a wireless mesh backbone network. Many statistical and hybrid methods were proposed to forecast the network traffic in

traditional IP backbone networks. However, those methods are not suitable for traffic prediction in wireless mesh backbone networks. The fast evolution of mobile communications and IoT led to a strong increase in traffic demands. This fast evolution challenges the capacity of wireless mesh networks. To cope with this issue authors propose a traffic prediction method for a wireless network mesh backbone network called Deep Belief Network and Gaussian models (DBNG) that predicts two separate network traffic components obtained by the Discrete Wavelet Transform (DWT) - which as stated by the authors is the first paper to address the problem of traffic prediction for wireless mesh backbone networks.

After a short background regarding DWT and DBN is presented, the authors formulate the problem statement and methodology. As mentioned earlier, the proposed solution is partitioning network traffic into two separate components. Using the provided DWT equation, network traffic is decomposed into long-range dependence and irregular fluctuation behavior itself. Based on the independent prediction of those two components a predictor of network traffic is obtained. The long-range dependency (meaning that traffic at any time depends upon multiple historical traffic data) component is estimated by the DBN architecture while irregular, gusty fluctuations are assumed to obey a Gaussian model. Those components are described by the scaling and the discrete wavelet transform coefficients respectively

Firstly, DBN training was performed. To achieve that, the authors used a layer-wise greedy strategy and a specified training set. Trained DBN can then describe the long-range dependence of traffic flows expressed by the scaling coefficients and carry out its predictors. Secondly, network fluctuations expressed by the discrete wavelet transform coefficients are assumed to be obeying a Gaussian distribution. Using Maximum Likelihood Estimation (MLE) expectation and variance of the Gaussian model are calculated. Having both of those values, authors generate a value in terms of this distribution which is a predictor of the discrete wavelet transform coefficient. Therefore, with scaling and discrete wavelet transform predictors, authors forecast the network traffic by the inverse discrete wavelet transform. It is worth noting that this algorithm is presented in a block diagram and a pseudo-code.

In simulations, the authors used a real network traffic dataset containing 2016 time slots. The first 2000 time slots were used as the training set for the DBN architecture and Gaussian model construction.

Results are compared with three methods - Principal Component Analysis (PCA), Tomogravity, and Sparsity Regularized Matrix Factorization (SRMF). Firstly, real traffic versus its predictor is shown on plots and discussed by the authors. These plots show that DBNG proposed by the authors outperforms other methods in terms of prediction accuracy. Next, two metrics are introduced: the spatial relative error and the temporal relative error. DBNG is designed to predict each traffic flow independently, and as a result, the improvement in temporal relative error is higher than that in spatial relative error. This observation is also supported by the figures presented in the results and analysis.

We believe that simulations were not discussed comprehensively and a few things are missing for example network topology and overall network parameters. We also believe that several things raised our doubts in this paper. Firstly, when discussing Figure 5(a) we believe that authors meant SRMF and not SRSVD. Secondly, instead of Figure 7(a), the authors meant Figure 6(a). We also think that network traffic flow identities are sorted in ascending manner and not descending (figure 6(a)), when referring to figure 6, authors meant figure 7, instead of x_1, x_i should be present in line number 4 of pseudo-code.

Nevertheless, this work is interesting in terms of the problem statement, as authors decompose network traffic and predict its components independently. Experiments show that this method achieves satisfactory prediction error and therefore may be used in wireless mesh backbone networks.

5) SUMMARY OF TRAFFIC PREDICTION

As one can see, ML can be used to solve various traffic prediction problems in classical IP networks, optical networks, wireless networks, and SDN. In [20], LSTM RNN is used to predict the traffic matrix in SDN network based on real, historical data. In [21], two different ML (MCTS and ANN) methods were selected and compared to predict traffic and minimize the average request blocking percentage in the backbone network utilizing EON technology. Authors of [22] proposed three ML (NN, GPR, oBMM) methods for flow size prediction and its classification based on information carried by the initial network packets of each flow. In [23] authors propose another approach for traffic prediction. Firstly, they decompose network traffic into two separate components. Based on the independent prediction of those two components (first predicted by DBN and second assumed to obey the Gaussian model) a predictor of network traffic in a wireless mesh backbone network is obtained. ML methods described in this section allow for solving traffic prediction problems in various network types, and their further research may produce better results.

Using NN-based algorithms it is possible to, for example, predict traffic parameters, blocking probability or characteristics of traffic flows (sizes and types). Predicted values can be further used, as an input, for algorithms improving other aspects of communications and networking considered in the next sections.

B. TRAFFIC ENGINEERING

TE is a set of procedures comprising key problems in design and control of the network. As a result, TE algorithms provide both optimization of network resources' utilization and simultaneously required level of service guarantees, expressed e.g., by QoS and Grade of Service (GoS) target values. TE begins with well-designed network and in order to match allocated resources and tuned control algorithms with imposed input traffic and quality requirements.

1) ML-POWERED LSP OPTIMIZATION IN MPLS

Multiprotocol Label Switching (MPLS) technology combines two fundamental networking paradigms: connectionless packet transfer with connection-oriented switching using labels. Additionally, it is designed to support IP flows with established MPLS ''tunnels'' and the process of constructing virtual paths (Label Switched Path (LSP)s) is, in fact, the routing procedure, with possibly introduced constraints, in the MPLS domain. The paper [30] describes the implementation of NN in MPLS in order to facilitate the optimization process. Describing directly implemented optimization process: NN ML method is used to compete with traditional provisioning of LSPs done typically by solving Mixed Integer Linear Programming (MILP) an optimization problem. Thus, ML method is used to compete with the strict, precise method. We are underlining that this is an early implementation of ML method in networking, but it should be considered as an introduction of modern optimization tool at the end of MPLS standardization process [30], thus, it is an illustration of a practically important issue.

To figure out the meaning and advantage to introduce NN method it is necessary to describe how the structure of LSPs is defined in MPLS in a traditional way:

- for off-line LSP design the optimization problem is solved by e.g., MILP method with defined any objective function.
- for on-line LSP establishment the dedicated control plane procedures are activated, such as ReSerVation Protocol with Traffic Engineering (RSVP-TE) and Constraint-Based Label Distribution Protocol (CR-LDP), defining Forwarding Equivalence Class (FEC), finding the path, and allocating labels in a distributed way.

For off-line LSP design MILP optimization method can be used, a linear program with integer constraints and the global optimum selection of paths relative to the chosen objective function. A two-stage solution is employed: the LP-relaxation phase plus searching for a solution near the pure LP optimum. In order to avoid limitations of two-stage approaches-specific search algorithms are implemented using heuristics. Generally, implementing a full formal framework is very time-consuming, and thus it can be simplified using the NN method. An alternative approach is to use heuristics, however this can lead to difficult searching for a properly working method and initial starting point in the input space.

The NN method is used to combine a strict MILP method with an approximate heuristic method. NN is trained using results from a formally correct and precise MILP optimization procedure. The new optimization chain proposed in [30] works as follows:

- NN is trained by MILP results from strict optimization process: the key difficulty in this step is to use sufficiently wide set of scenarios and that NN is trained for specific network topology and traffic matrix;
- using NN with current input data set to find near optimum solution;
- using the heuristic to carry further refinement of flows' allocation - the heuristic starts from the solution closed to the optimum. As a heuristic method, the marginal increase heuristic marginal increase heuristic (MIH) is implemented.

So, the reason to use the NN method is to obtain results in shorter execution time and with reasonable high accuracy compared to the case when formally correct and ''classical'' MILP optimization problem is used.

It comes from the above description that the fundamental task in described approaches is to have a properly trained NN. It is achieved by using a fixed topology consisting of N nodes and bidirectional links but flexibly changing in wide range the traffic demand matrix with $N(N-1)$ flows and using also an overload ratio $[0.1, 0.5445]$ for each ingress node (the reason not to use the overload ratio from 0 to 1 is due to tested workability of carried experiments); further reasonable conditions for credible way of the demand matrix generation are also defined in [30]). After generation of the training set, the NN is trained to produce LSPs structure.

An Important issue is to tailor the complexity and accuracy of such defined and implemented optimization components: MILP, NN and MIH.

During the training period, NN uses the traffic demand vector and correlates it (i.e., learns the optimum mapping) with the structure of paths obtained as a result of MILP optimal solution. The output from NN is then additionally corrected by the MIH algorithm to check and possibly change any broken constraints (this is a ''live'', suboptimal search). When the OD pair has enough allocated capacity, there is no need to subsequently activate the MIH optimizer.

The architecture of the NN-powered optimizer for the working phase using as an input the network topology and actual traffic demand matrix is shown in Fig. 14.

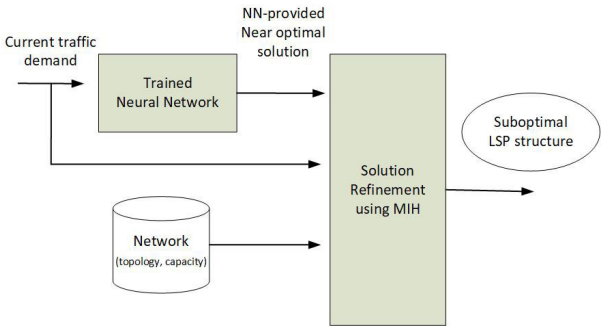


FIGURE 14. Illustration of using NN in MPLS proposed in [30].

It should be underlined that the key point of described proposal is methodologically interested and original usage of NN as an intermediate, additional tool for LSP structure design optimization. The goal (target) of the optimization process is defined by the MILP construction.

Proposed tools are checked by comparing the results for MILP optimization with a novel and complex combination of MILP/NN/MIH and for US IP backbone network topology (23 nodes and 43 links). Additionally, there are included simulation results for IP/MPLS traffic transported using the designed LSP structure. The criterion to compare both methods is to reach the difference in Balance objective function below 1%, however in most cases obtained difference is much lower.

The NN-powered optimization process gives the structure of LSPs to be used for routing of IP packets and this is a different approach from those classically used routes provided by Open Shortest Path First (OSPF). OSPF is a routing protocol that uses link-state technology to find the best path between routers in a network. OSPF routers collect information about the complete network topology (every single link and its cost) and build a shortest-path tree for each destination. Due to this observation further simulation studies were carried out using the ns-2 simulator and reported in [30] to show superiority of explicit routing in MPLS with NN/MIH path searching when comparing with classical IP/OSPF routing. Results are briefly provided in Table 9.

TABLE 9. Simulation studies for ML method in MPLS presented in [30].

Feature/observation	IP with OSPF	IP with MILP/NN/MIH
IP packet losses start from cumulative traffic demand value	70 Gbps	120 Gbps
- average loss for 120 Gbps	18%	5%
- maximum loss for 120 Gbps	70%	25%

It has to be underlined that presented results show not only the correctness and efficiency of the usage of NN method in optimization of LSP structure, but also thanks to the achieved time-efficiency such NN method can be used even for on-line dynamic flow admission control (see Fig. 3 in [30]). Such methodology needs very precise and lengthy processing for provided topology and traffic demand matrix. It should be mentioned that no detailed information about the used MILP optimizer is disclosed which makes presented efficiency comparison less credible. Also, there is only brief information about network topology and traffic demand matrix. The US core IP network used for studies and shown in Fig. 4 of [30] is presented without the demand matrix, however this network has undergone significant quantitative changes concerning installed resources. When searching e.g., in SNDlib - http://sndlib.zib.de/home.action - it does not provide network parameters for the presented case.

Complex description of the presented methodology of NN introduction into route design process and shown practical results either from optimization phase and simulation studies, suggest that such TE approach is significant for network operators. The main message from the presented results is that such NN tool can be introduced with enormous efforts in preliminary, training period. Then the operator gets the efficient optimization and control tool with acceptable accuracy of results with difference below 1% from the optimum value and running time 40 times shorter (as reported in [30]). So, the operator, thanks to additionally implemented NN method can migrate from off-line LSPs design to on-line LSPs admission control in MPLS.

In this section we present the ML method described in [30] as an extra step for LSP optimization and set up in MPLS and after training it simplifies LSP provisioning with the acceptance of loosing of accuracy by approx. 1%.

Since TE mechanisms are widely used and can be based on any available information, they are also tailored to implemented classes of protocols, such as routing, traffic classification and prediction, network topology design, active control in nodes, etc., so ML would help in incorporating a full knowledge database and producing close to optimal solutions.

As we can see, TE mechanisms improved by NN-based solutions can optimize transmission in a network. Usually, these mechanisms operate based on known metrics, parameters, or measurements. However, it is also possible to use results from prediction NN models, like those analyzed in Section IV-A. Such combined solutions are expected to bring even more effective solutions. Furthermore, prediction models as well as TE algorithms can be a part of the advanced quality of service and quality of experience solutions, as shown in Section IV-C.

C. QUALITY OF SERVICE/EXPERIENCE

QoS is a measure of how well a network can deliver its services to meet the expectations of its users. It may involve various aspects of network performance, such as bandwidth, latency, error rate and uptime, while, QoE is a measure of the delight or annoyance of the user of an application or service. QoE focuses on the entire service experience; it is a holistic concept, similar to the field of user experience, but with its roots in telecommunication.

In this section, solutions utilizing NN to provide QoE requirements are analyzed. Particularly, the QoE anlysis in SDN was presented in [14]. We selected this paper because SDN is still promising architecture. Then, we presented the solution [31] to predict QoE metrics in an environment based on DNN. Finally, the presented works are summarized.

1) QOE PROVISION IN IOT APPLICATIONS

In [14], the first proposal to predict QoE of video transmission based on information directly extracted from the network packets using a deep learning model is described. The authors assume that a transmission can be organized in three layers. The first layer is responsible for maintenance of cloud/central services. The middle layer ensures edge processing of network services traffic. The last layer is responsible for traffic service of end-user devices. The middle layer allows for better transmission of e.g., real-time services traffic. For more accurate operation of this layer the authors

propose a classifier which can detect anomalies at the time instant and predict them at the next immediate instant. The QoE classifier is a part of QoE predictor which should be intended to be deployed as a quality monitoring service at the edge processing layer.

In current video transmission, usually used QoE detector can predict QoE directly from transmitted videos, the network packets, or end-user saved on-line meetings (e.g., related network activity). The QoE detector presented in [14] also considers the sources mentioned above. However, it uses deep learning models. For proper operation of this detector, it is necessary to build a training dataset. While data is acquired from end users, it is difficult to collect valuable samples. Such data can be deceitful since people may not be interested in giving appropriate answers. As the authors explain in details, the proposed QoE detector is based on a deep learning classifier. On the other hand, this classifier is based on the combination of a CNN and a RNN with a final Gaussian Process (GP) classifier. A binary classification (good or bad quality) for seven common classes of video anomalies (blur, ghost, columns, chrominance, blockness, color bleeding, and black pixel) that can occur when watching the videos is implemented in the classifier.

Three prediction models had been analyzed by the authors of [14]. In the first one, classical ML models (RF, logistic regression, etc.) were used. This approach resulted in the worst results. The second model was based on deep learning models (CNN and RNN) and the third one was a combination of deep learning models and a GP classifier. The last two out of three models gave similar results, but the last one was assessed by the authors as the best one. The models used in the first approach were: linear Support Vector Classifier (SVC), SVC with an RBF kernel, logistic regression, MultiLayer Perceptron (MLP), RF, Gradient Boosting Method (GBM), and Adaboost. For GBM and Adaboost decision trees as elementary classifiers were used.

In the second group several deep learning architectures were analyzed. The authors explained that the implementation of the RNN network was based on an LSTM, which is a special type of RNN. A CNN was implemented as a neural network which applied several convolutional filters to image-formatted data. Stochastic Gradient Descent (SGD) was used to learn the filter's weights. An RNN was also implemented as a neural network, intended to treat time series data by using sequential input data and an additional internal state.

Finally, as a result of analysis, three canonical models for the second and third groups were selected:

- Model 1, which used only RNN layers,
- Model 2, which applied a sequence of CNN and RNN layers,
- Model 3, which added a GP classifier to a sequence of CNN and RNN layers.

They are briefly summarized in Tab. 10.

TABLE 10. Summary of used models.

ML model	Implementation method	Canonical model no.
classical (RF, logistic regression, etc.)	based on an LSTM	1
deep learning models (CNN and RNN)	based on an LSTM and neural network	2
deep learning models and a GP classifier	based on an LSTM and neural network using sequential input data	3

The GP classifier was based on the so-called Laplace approximation defined in [32]. Finally, for all architectures, the training was conducted with 200 epochs. An epoch was a single pass of all the samples in the training dataset.

The results were obtained using the following performance metrics:

- accuracy,
- precision,
- recall,
- F1-score,
- area under the ROC curve (AUC).

The last two metrics are suitable considering the highly unbalanced distribution of QoE errors.

The results were analyzed in two aspects: one-by-one label or aggregated labels for several models, including MLP, Linear SVC, SVC-RBF kernel, Logistic regression, GBM, Adaboost, Random forest and three models proposed in the paper. They are summarized in Table 11. The best accuracy was observed for model 2 and the best $F1$-score was noticed for model 3. On the other hand, the worst results were observed for model 1, which was formed exclusively by an RNN network and did not include a CNN in its architecture. The one-by-one metrics were presented based on model 2. The authors noticed excellent results for some labels (chrominance and color bleeding), acceptable for others (columns and ghost) and terrible for the rest (blur, blockness, and black pixel). It is also important to note that the performance metrics get worse at the next time step in comparison to the current time step. It was expected by the authors, but the small reduction in performance is good news. This confirms that the prediction of QoE was possible.

The conclusion of the paper is that the classifier allows achieving excellent prediction results for the non-extremely unbalanced labels. The best among proposed models is that one, which includes a combination of CNN and RNN, with the CNN network being the most critical piece.

TABLE 11. Summary of obtained results.

Best accuracy	Best $F1$ score	Best precision	best recall
model 2	model 3	Random forest	SVC-RBF kernel

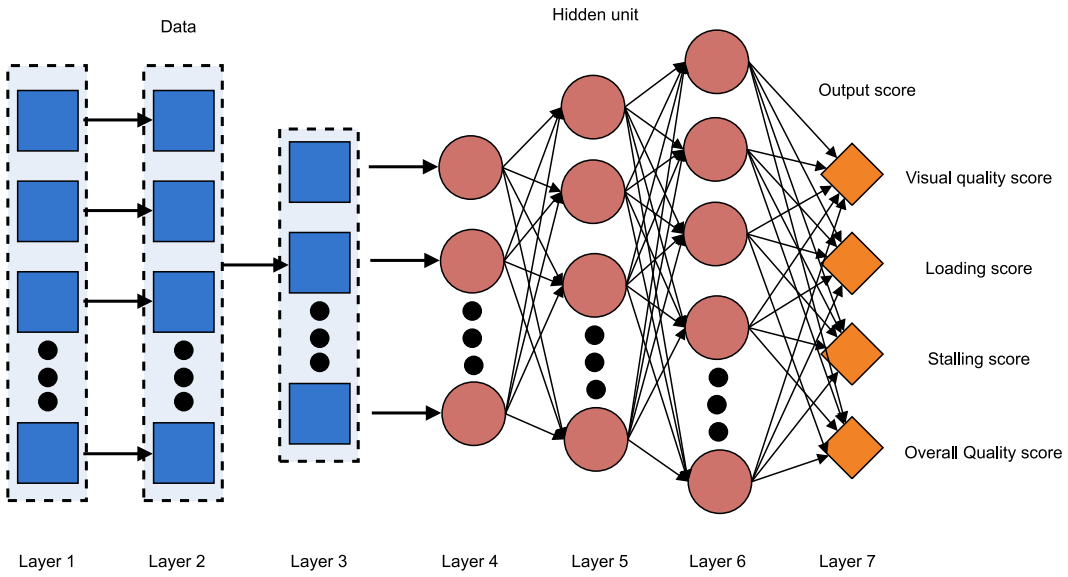


FIGURE 15. DNN architecture.

2) PREDICTION OF USERS QOE SCORES FOR VIDEO TRANSMISSION

In [31], a deep learning-based QoE prediction approach with a large-scale QoE dataset for mobile video transmission is proposed. The authors developed a DNN to learn and assess the relationships between network parameters and the users QoE scores. This solution can also be used to predict users QoE scores for mobile video transmission. The relationships between parameters of mobile video transmission and the subjective scores given by users are complex. It is not easy to assess this relationship correctly. Adaptive learning methods look to be a good solution to parameterize this relationship. The application developed by the authors implements the proposed approach to analyze mobile video transmission. A dataset including the QoE data of eighty thousand samples has been constructed based on collected data. Four metrics:

- visual quality,
- loading,
- stalling,
- overall quality

used in this application allow to collect QoE data for each user. Along with this data 89 network parameters have been recorded. Using the feature selection method and the boxplot method the redundancy of the network parameters for QoE assessment has been reduced and raw data are cleaned by removing outliers. At the end, DNN is used to learn the relationships between the network parameters and the subjective QoE users scores.

A difficulty in machine learning-based process for QoE analysis is usually the lack of sufficient training data. The authors of paper [31] established their own large-scale QoE database. Data was collected by using mobile application. Each user rated four factors listed above, i.e., visual quality, loading, stalling, and overall quality. At the same time, 89 feature parameters of a video file were observed. The authors received eighty thousand QoE sets of data from users. As a result, the total received score set by the authors is $4 * N$, where N is the total data number. The goal was to find the relationships between the feature set and the score set. Using the Feature Subset Selection (FSS) method, which selects a subset of features, allowed to realize that 16 network parameters are critical to QoE prediction. Unnecessary data (the outliers of the data) has been cleaned by using the box-plot, which is sometimes known as the box-whisker plot.

The network model was constructed based on Deep Neural Network architecture. Seven layers have been identified. Their relations are presented on Figure 15.

Relation between the first layer and the second layer is an FSS process. Progressing from the second layer to the third layer is the data cleaning process. The fourth, fifth, and sixth layers are hidden layers with nodes 12, 48, and 48, respectively. The seventh layer is the output layer with four nodes representing the four metrics. All the interactions between feature parameters had been learned as a result of full-connections of the hidden layers, for generating the final QoE scores. The network was implemented in the Tensor-Flow framework. The model was trained with the Adam optimizer, with a rate set to 0.0001. The iteration number was set at 30 000, and the batch size was set to 128.

The experimentation process was conducted by the authors using the collected dataset. The proposed method was compared to some related QoE predicting approaches:

- Support Vector Model (SVMo),
- Adaptive Neuro Fuzzy Inference System (ANFIS) model,
- decision tree and DeepQoE model.

The obtained results have been discussed separately in three parts:

1) Results of Feature Selection

Using the method based on Spearman's correlation coefficient, the authors found that 16 parameters from the 89 recorded feature parameters are strongly correlated with the user's subjective scoring. These parameters are: user buffer time, buffer delay, bitrate, video size, video average rate, video play total time, screen resolution long, video peak rate, video all peak rate, light intensity, phone screen brightness, ping average rate, signal strength, video stalling number, video stalling total time, and video stalling duration proportion. The analysis of results can be presented by some examples:

- strong positive correlation between the overall quality score and the bit rate, video size and video average rate is observed,
- strong negative correlation of the loading quality and the stalling quality with quality score is noticed,
- with the higher the clarity, the larger the video, and the larger the amount of data that needs to be buffered are observed,
- for constant rate the longer buffer time is observed,
- for longer play time, the longer the buffer time and the longer the video start time are observed and lower two scores (loading and stalling) are obtained.

2) Results of Data Cleaning

The results show the correlation between the selected 16 features and the subjective scores through the feature selection. Among them four feature parameters and the corresponding score are bigger than in the case for the rest 12 feature parameters. Table 12 shows the feature parameters for selected score metrics.

TABLE 12. Relation between feature parameters and scores.

Score metric	Parameters
Visual quality	video play total time user buffer time video peak rate video average rate
Loading quality	buffer delay user buffer time video play total time bit rate
Stalling quality	buffer delay user buffer time video play total time bit rate
Overall quality	buffer delay video size video average rate bit rate

Using parameters listed in Tab. 12, the authors were able to remove the outliers in the selected database after FSS. To clean the user data the box-plot method was used.

3) Results of QoE Prediction

The first conclusion is that the number of nodes in the hidden layers should be four times the first level. In such a case, the QoE model shows the best results. The Root-MeanSquare Error (RMSE), which represents the square root of the second sample moment of the differences between predicted values and true values or the quadratic mean of these differences was used to analyze the differences between the predicted QoE values and the true values. The second metric was the Mean Absolute Error (MAE), which is a measure of the difference between two continuous variables.
It was assumed that in the environment for tests the percentage of each fraction was 20%. In this case, the result of the model prediction is the best. The ANFIS, SVMo, Decision Tree (DT) and DeepQoE were selected as the compared methods to the proposed one by the authors. The analysis of obtained results shows that the RMSE of the new proposed method is the smallest in the dataset. Moreover, the value of MAE is close to the method of Decision tree and DeepQoE. It shows that the proposed method works correctly. It also confirms that DNN has an ability in presenting the complex relationship between the network feature parameters and the user scores.

The summarization of the results is presented in Table 13.

TABLE 13. Summary of obtained results.

Feature Selection	Data Cleaning	QoE Prediction
16 parameters from the 89 recorded feature parameters are strongly correlated with user's subjective scoring	4 feature parameters can be selected for each score	RMSE is the smallest in the dataset for the proposed method and MAE is comparable with other methods

The QoE aspects, analyzed in this section, strictly correlate with prediction and TE considered earlier. For example, the prediction models are engaged to improve QoE in [14]. The prediction approach with a large-scale QoE dataset for mobile video transmission is analyzed in [31]. Correlations between overall quality and TE indicators (bit rate, amount of data) have been observed, suggesting that the combination of the analyzed aspects can result in an effective advanced solution. The NN-based solutions are expected to reflect the subjective manner of QoE aspects, while TE mechanisms should improve objective network conditions and transmission parameters.

D. RELIABILITY AND ROBUSTNESS

In this section, the reliability aspect in networks is considered. ML techniques are widely used to improve it in different

ways, e.g., detect failures, predict upcoming deterioration, find an origin of the problem, decrease time needed to repair, or efficiently provision redundant resources. Detection of seamless deterioration of optical network performance is an example of application of NN method considered in this section.

Fault management issues in optical networks are addressed in [33] with special interest to the failures that may seamlessly deteriorate network performance. As there is no immediate and complete service disruption, it is critical to effectively detect and identify anomalies. The most common approach to this problem in the context of ML is to, firstly, prepare a training data set then train a supervised learning model and, finally, apply it - detect and identify anomalies. However, even comprehensive access to the network monitoring results does not ensure sufficient amounts of data to effectively train a supervised model. The problem is even more challenging, as different types of failures usually follow different patterns.

To overcome these shortcomings and propose an efficient and practically justified solution for anomaly detection and identification in optical networks in [33] a hybrid approach was proposed. Namely, monitoring results are clustered with the use of an unsupervised algorithm. Then, learned patterns are automatically transferred to the second module, applying a supervised algorithm to perform data regression and classification. It is critical to note that such an approach enables efficient and accurate detection and identification. Modules are denoted as Data Clustering Module (DCM) and Data Regression and Classification Module (DRCM), respectively.

The workflow of the proposed framework comprising both novel models integrated with the network infrastructure is intuitive. The solution requires a monitoring system to collect data plane measurements like spectrum utilization, signal power, or noise level. As network monitoring is a continuous process, results are stored in the database and periodically retrieved to be preprocessed and further passed to the DCM. The DCM is designed using a density-based clustering algorithm which organizes input data into clusters and outliers.

In the paper [33] pseudocode was provided to explain DCM, but it may be reasonable to explain it in a less formal way. The rationale is to recursively analyze all samples in terms of distance (in the Euclidean sense) between samples. Based on that, new clusters are created (when a group of similar results if distant enough from other clusters) or existing clusters are expanded (the sample or group of samples are close to the existing cluster). The distance threshold was determined empirically. A second important assumption regards to the number of samples that should be considered as a group. A relatively small value should be assigned to that parameter when compared to the sample set. The rationale fully reflects the fact that anomalies are occasional so DCM learns normal behavior and outliers are considered as potential faults. It is important to note that any

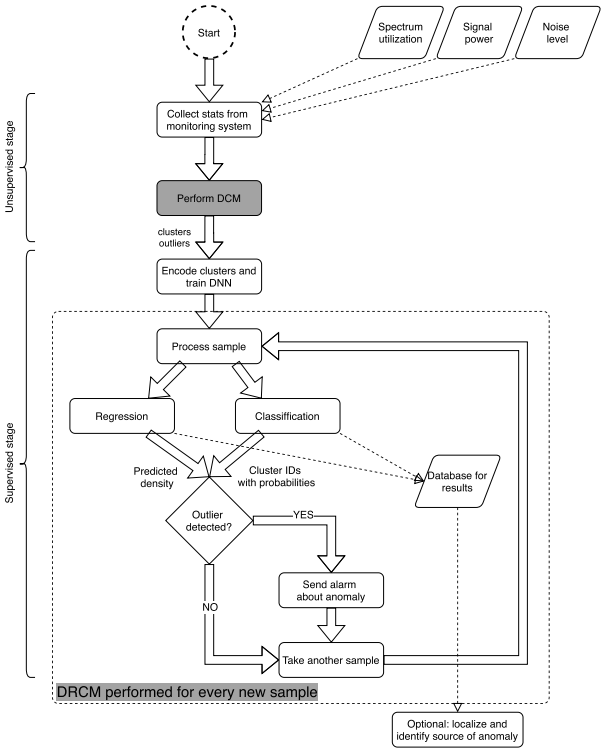


FIGURE 16. Flow chart of the algorithm proposed in [33].

prior knowledge or assumptions regarding the anomalies are needed for DCM.

The output of DCM is further used for training purposes by DRCM which is designed as a DNN. For this purpose, it is properly encoded by adding a cluster ID to each sample. Based on that, DRCM is able to acquire knowledge about sizes, shapes and locations of clusters based on analysis performed by DCM. It is especially important as DCM, even if computationally complex and time-consuming, is run only once while DRCM analyzing every new sample can be significantly simplified. Due to the fact that anomalies are rare events, it will not be efficient to simply predict cluster identifier for incoming samples. To address this issue authors proposed an interesting approach based on the observation that in each cluster samples closer to the border are less dense. The algorithm measures density after both possible decisions regarding adding the sample to the existing cluster or not (this stage is denoted as regressor). If a big change in terms of the cluster's density is observed then the analyzed sample should not be added to that cluster, and further, if it is not added to any cluster, it should be considered as an anomaly (stage denoted as classifier). Both regressor and classifier were implemented as blocks of neurons in the same DNN. A brief visualization of the complete solution proposed in [33] is presented in Figure 16.

Finally, the solution may try to localize the source of the problem in the optical network. The very last step is to provide feedback about identified disruptions to the network control so that it will be able to counteract. By combining

TABLE 14. Summary of pros and cons of supervise and unsupervised components of solution proposed in [33].

Approach	Pros	Cons
Unsupervised	Runs only once Suitable to rare anomalies	Not-scalable as whole sample set is processed
Supervised	Complexity is function solely of DNN Self-taught solution	Run for each incoming sample

both unsupervised and supervised learning, authors aimed at taking advantage of both solutions and mitigating drawbacks as summarized in Table 14.

Authors provided comprehensive and realistic validation of the proposed approach. First to note is that a proper testbed was built to collect input data for experiments. Optical spectrum analyzers gathered samples while abnormal situations were simulated by increasing attenuation introduced by wavelength selective switches. Experiments were performed in two use cases comprising several research scenarios. To conclude considerations in the paper, we briefly summarize results and conclusions in Table 15. Use case I regards the detection of failures in the scope of a single link and comprises separate experiments in regard to the DCM and DRCM modules. In the use case II anomalies are investigated in the scope of end-to-end lightpaths and only DCM was investigated. Figure 17 summarizes several potential improvements that we identified.

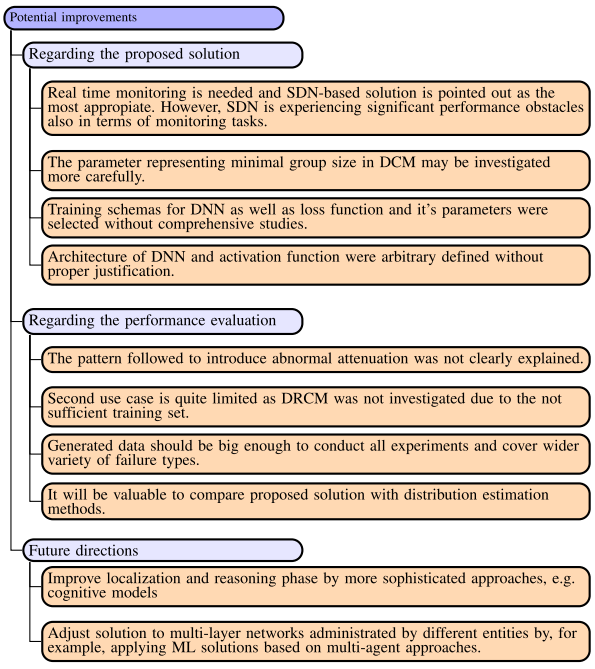


FIGURE 17. Summary of potential improvements to further enhance research presented in [33].

Papers presented in this section analyze failures and NN-based methods to prevent them. These solutions can be successfully enhanced by, for example, traffic prediction models (e.g., to estimate expected behavior and use as a baseline), traffic engineering techniques (to counteract reliability issues), determined impact on quality experienced by the user (to reflect the subjective manner of the problem). As shown in Sections IV-A, IV-B and IV-C, NN can also be successfully applied to these aspects.

E. SECURITY

This section discusses the potential of machine learning (ML) in the field of security. The research on designing and implementing ML algorithms and systems for security has expanded rapidly and encompasses various domains. These domains include detecting spam and phishing, intrusion detection systems (IDS), identifying and classifying malware, managing security policies (SPM), and checking for information leaks. While machine learning cannot fully automate a cyber-security system, it is more effective than other software-based methods in identifying security threats. As a result, it reduces the workload of security analysts.

In this tutorial paper, we decided to limit the broad scope of ML applications to the network security area, presenting two representative papers [34] and [35] to show relevant NN applications and their potential for network security. One must note that intrusion detection systems are one of the most essential applications of NN in the domain of communications and networking. It results from the fact that IDS most rely on classification, for which NN are very accurate. Numerous works have been published in this research area, but our aim is to provide an example-based tutorial and ensure deep and practical understanding on leveraging NN.

The section starts with paper [34] focusing on deep learning (DL) technique for intrusion detection. The authors propose a non-symmetric deep autoencoder (NDAE) for unsupervised feature learning, along with a deep learning classification model constructed using stacked NDAEs. Next, we select work [35] focusing on detection of wireless network intrusion. The proposed model uses Deep Belief Network (DBN) as the feature extraction layer and Support Vector Machine (SVM) as the classification layer.

1) A DEEP LEARNING APPROACH TO NETWORK INTRUSION DETECTION

In recent years, research on Network Intrusion Detection System (NIDS) has emphasized the use of machine learning and shallow learning techniques, such as Naive Bayes, Decision Trees, and Support Vector Machines (SVM), for detecting malicious actors. However, these techniques have limitations that can make their application challenging in a heterogeneous and dynamic environment. One such limitation is the need for expertise to identify useful data and patterns, and another is the requirement for a vast amount of training data.

TABLE 15. Experiments conducted in [33].

Scenario			Conclusions
Use Case I	DCM	Investigation on algorithm's parameters	Distance threshold and minimal group size should be adjusted to balance the tradeoff between false negatives and positives.
		Comparison against K-means	K-means provides too high false negative ratio as only spherical clusters are possible. It further, requires the number of clusters as an input parameter.
	DRCM	Convergence of training stage	Any notable overfitting is observed.
		Accuracy in anomaly detection	Regression is critical to achieve acceptable results. A tradeoff between accuracy and complexity is tunable.
Use Case II	DCM	Algorithm performance	Conclusions are in accordance to those from use case I

TABLE 16. NIDS challenges listed in [36].

Feature	Description
Volume	The volume of data both stored and passing through networks continues to increase.
	The traffic capacity of modern networks has drastically increased to facilitate the volume of traffic observed.
	Many modern backbone links operate at wire speeds of 100 Gbps or more.
Accuracy	Greater levels of granularity, depth and contextual understanding are required to provide a more holistic and accurate view. This comes with various and time financial, computational costs.
Diversity	Increase in the number of new or customized protocols being utilized in modern networks.
	Difficult to differentiate between normal and abnormal traffic and/or behaviours.
Dynamics	Given the diversity and flexibility of modern networks, the behaviour is dynamic and difficult to predict.
	This leads to difficulty in establishing a reliable behavioral norm.
	It also raises concerns as to the lifespan of learning models.
Low-frequency attacks	These attacks have often thwarted anomaly detection techniques, including artificial intelligence approaches.
	Imbalances in the training dataset, meaning that NIDS offer weaker detection precision when faced with these types of low frequency attacks.
Adaptability	Networks adopted new technologies to reduce their reliance on static technologies and management styles.
	More widespread usage of dynamic technologies such as containerization, virtualisation and SDN.
	NIDSs will need to be able to adapt to the usage of such technologies and the side effects they bring about.

To overcome the aforementioned limitations, the research community has turned to deep learning, an advanced subset of machine learning that has gained significant interest in various domains. One of the papers focusing on deep learning technique for intrusion detection is [34]. The authors propose non-symmetric deep autoencoder (NDAE) for unsupervised feature learning, along with a deep learning classification model constructed using stacked NDAEs. This solution combines the strengths of two learning techniques: the power of stacking the proposed NDAE for deep learning and the accuracy and speed of Random Forest (RF) for shallow learning. As a result, the proposed approach significantly reduces training time and improves classification results, outperforming leading methods like Deep Belief Networks (DBNs).

The paper initially outlines a set of challenges related to Network Intrusion Detection System (NIDS). The authors identify various issues, including volume, accuracy, diversity, dynamics, low-frequency attacks, and adaptability, as some of the most significant challenges in this area. We summarize these NIDS related challenges and provide their description in Table 16. To provide a concise summary of these NIDS-related challenges, we have compiled a table (Table 16) that includes a brief description of each challenge.

The authors provided also background information on deep learning necessary to understand the concepts behind the model proposed in the paper. They focused on two popular techniques used within deep learning research, i.e, auto-encoders, applied by the proposed solution, and stacked auto-encoders.

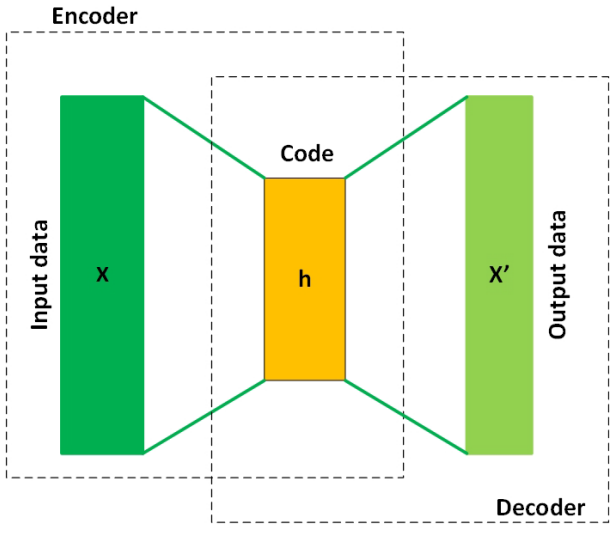


FIGURE 18. Auto-encoder components.

An auto-encoder is an unsupervised neural network-based feature extraction algorithm, which can automatically learn

effective and robust features from a large amount of unlabeled data. Auto-encoders consist of 4 main parts (see Fig 18):

- Encoder: the model learns how to reduce the input dimensions and compress the input data into an encoded representation.
- Bottleneck: the layer that contains the compressed representation of the input data.
- Decoder: the model learns how to reconstruct the data from the encoded representation to be as close to the original input as possible. In other words, it is trying to learn an approximation to the identity function.
- Reconstruction loss: the method that measures measure how well the decoder is performing and how close the output is to the original input.

In the case of stacked auto-encoders, the model includes multiple encoding or decoding layers in the middle (see Fig. 19), unlike the basic auto-encoder framework that only has a single hidden layer. The output from each hidden layer is used as the input for a progressively higher level. Therefore, the first layer of a stacked auto-encoder usually learns first-order features in raw input, while the second layer learns second-order features relating to patterns in the appearance of the first-order features. Subsequent higher layers learn even higher-order features.

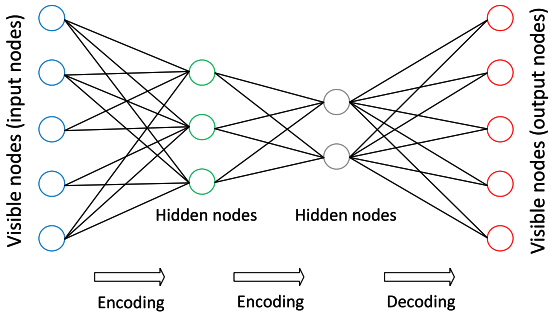


FIGURE 19. Stacked auto-encoder model diagram.

The primary contribution of the paper is the development of a novel deep learning classification model that uses a stacked NDAE, described above. By stacking the proposed NDAEs it is possible to create a deep learning hierarchy, that can learn the complex relationships between different features. This approach, which combines the proposed stacked NDAEs with a Random Forest as a shallow learning classifier, creates a powerful solution that outperforms discriminative models like RF, KNN, and SVM. In the paper, the RF classifier was trained using the encoded representations learned by the stacked NDAEs to classify network traffic into normal data and known attacks. According to the authors, the final structure of the proposed model was determined through experimentation with numerous structural compositions to achieve the best results. The model's definitive structure is illustrated in Fig. 20. It uses two NDAEs arranged in a stack and is combined with the RF algorithm. Each NDAE has three hidden layers, with each hidden layer using the same number of neurons as the number of features (as indicated by the numbering in the diagram).

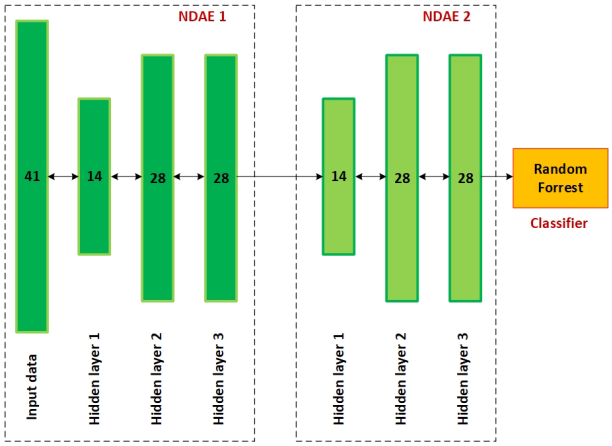


FIGURE 20. Stacked NDAE classification model proposed in [34].

An essential aspect of the paper is the section that provides an overview of the existing research on the applicability of deep learning within the domain of NIDS. The authors selected and briefly discussed several important articles including both surveys and research papers that focus on DL methods specifically for NIDS. For instance, the paper mentions the use of RBM with one hidden layer to perform unsupervised feature reduction, the method referred to as self-taught learning (STL), which combines a sparse auto-encoder with softmax regression, and the proposal of an automatic security auditing tool for short messages (SMS) based on the RNN model, to name a few examples.

The paper [34] also includes a section that provides a comprehensive description of the experiments conducted and the results achieved. The proposed classification model was implemented using TensorFlow, and two benchmarks datasets within NIDS research, KDD Cup '99 and NSL-KDD, were used for evaluations. The paper provides a detailed description of these datasets. The first dataset, the KDD Cup '99, used in DARPA's IDS evaluation program, consists of 4 gigabytes-worth of compressed tcpdump data resulting from 7 weeks of network traffic, and 4 900 000 single connection vectors each of which contains 41 features. Each vector is labelled as either normal or as an attack (specifying 22 specific attack types). The second newer dataset, namely NSL-KDD, is used by most current NIDS research and has the same structure as the KDD Cup '99 dataset.

The authors utilized a set of standard metrics, as presented in Table 17, to evaluate the performance of the proposed solution. These metrics include accuracy, precision, recall, false alarms, and F-score measures.

The authors concluded, based on the conducted experiments, that the proposed stacked NDAE model is highly promising, particularly considering the achieved average accuracy of 97.85%. However, the model requires a greater

TABLE 17. Measures used in [34].

Measure	Description
Accuracy	Measures the proportion of the total number of correct classifications
Precision	Measures the number of correct classifications penalised by the number of incorrect classifications
Recall	Measures the number of correct classifications penalised by the number of missed entries
False Alarm	Measures the proportion of benign events incorrectly classified as malicious
F-score	Measures the harmonic mean of precision and recall, which serves as a derived effectiveness measurement.

amount of data to learn from when compared to other deep learning-based models, making the results less stable for smaller datasets. Nevertheless, for larger classes the model offers improved precision, recall, and F-score values. It is essential to note that these comparable results were achieved with a significant reduction in training time by an average of 97.72%.

To briefly summarized drawn conclusions as well as future research directions we prepared the Table 18.

TABLE 18. Summary of conclusions and research directions presented in [34].

Summary	Novel classification model constructed from stacked NDAEs and the RF classification algorithm.
	Model implemented in TensorFlow and performed extensive evaluations on its capabilities.
	Benchmark KDD Cup '99 and NSL-KDD datasets utilized for evaluation.
Results	Proposed approach offers high levels of accuracy, precision and recall together with reduced training time.
	Stacked NDAE model was compared against the mainstream DBN technique.
	Proposed model offers up to a 5% improvement in accuracy and training time reduction of up to 98.81%.
Research directions	Assessing and extending the capability of the model to handle zero-day attacks
	Expand upon our existing evaluations by utilizing real-world backbone network traffic

2) COMBINED WIRELESS NETWORK INTRUSION DETECTION MODEL BASED ON DEEP LEARNING

The advancement of wireless network and computer technology has led to an increase in wireless network intrusion. Traditional intrusion detection methods have limitations in detecting cyberattacks due to their high concealment, large scale, and easy spread characteristics, resulting in lower detection accuracy rate and precision rate.

The research mentioned above identified the insufficient efficiency and accuracy of wireless network intrusion detection and proposed a combined model based on deep learning in [35]. The authors used the Deep Belief Network (DBN) as the feature extraction layer and Support Vector Machine (SVM) as the classification layer. The DBN layer employed the contrast divergence algorithm and the error back propagation algorithm to reduce data dimensions and extract features. It helped SVM to improve the ability to classify high-dimensional data.

The authors provided a detailed description of the proposed combined detection method. The framework consists of a multi-layer RBM, a BP network layer, and an SVM classification layer. The DBN architecture consists of a multi-layered RBM and a BP network layer. Each RBM has the characteristics of no connection within the layer and a thorough connection between layers. The hidden layer data of the previous layer is transferred to the visible layer of the next layer. After learning, the hidden layer variables are treated as low-dimensional features of the high-dimensional data of the visible layer. The BP network layer serves as an auxiliary layer to fine-tune the weight and improve the SVM classification capability. The adjusted RBM is then connected to the SVM layer. The specific network structure is presented in Fig. 21. It includes the following layers:

1) Multi-layer RBM structure: The feature extraction layer is constructed with a plurality of stacked unsupervised RBM shallow networks.
2) BP network layer: The BP network layer is connected to the end of the RBM structure, as an auxiliary layer to fine-tuning the weight of the network parameters.
3) SVM classification layer: After fine-tuning the weight, the output of the last-layer RBM is the dimensionality reduction feature, and access it to the SVM classification layer. The sequence minimum optimization algorithm is used for supervised learning and classifying

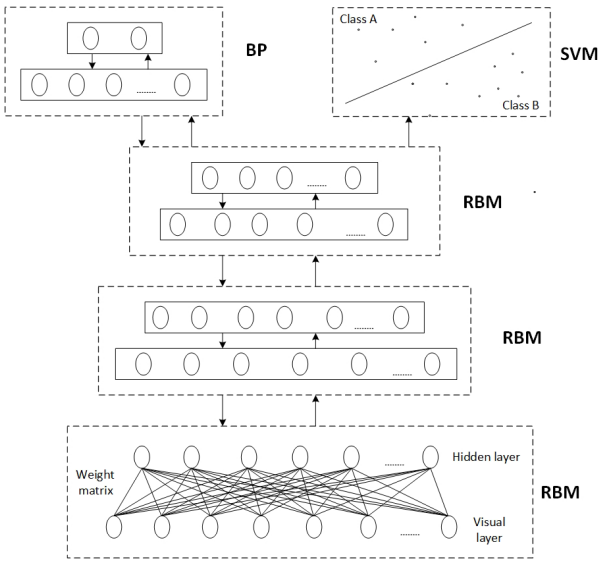


FIGURE 21. Combined deep belief network structure proposed in [35].

The process of the combined detection method was mainly divided into the following four stages (Fig. 22):

1) Pretreatment stage: First, feature mapping maps the training data set to the corresponding feature field. Then, the characteristic features are converted to numerical features by One-Hot Encoding. Finally, unify the dimensions and magnitudes between different features.
2) DBN training stage: First, RBM was pre-trained layer by layer through the contrastive divergence algorithm in an unsupervised way. The weight of the RBM is fine-tuned with the tag of the training data set and the back-propagation algorithm.
3) SVM training stage: First, the SVM parameters were configured. Then, SVM was trained through the sequence minimum optimization algorithm.
4) Testing stage: First, the detection method was subjected to the test data set. Then, performance evaluation indicators were calculated: accuracy rate, precision rate, recall rate, and F1. When the indicator reaches the expected level, the learning process would be ended, if it is not reached, it returns to the stage 2) to restart the training.

The description of components involved in processing flow of the intrusion detection model is given in the Table 19.

The paper brings a comprehensive evaluation of the proposed approach. Experiments were performed with usage of NSL-KDD as the detection data set in the experiment. A set of research scenarios have been tested including the analysis of the impact of network depth on DBN, the influence of the kernel function with the DBN-SVM combined detection method proposed in the 4-layer DBN condition, and verification and analysis of the training time, test time and detection performance of the detection method under different training data volume. The DBN detection performance of different network depths indicates that together with the network depth increase, the classification capability generally shows a growing trend.

According to the experimental results, the proposed detection model has good intrusion detection performance. After parameter optimization, the detection method described in the paper achieves accuracy rate, recall rate, precision rate and F1 over 0.98, what indicates that the combined detection method performs well in detecting and classifying cyberattacks.

The authors conducted a comparison between the proposed detection method and other approaches such as SVM, DBN, and PCA-SVM. After comprehensive analysis, the authors concluded that the proposed method outperforms the other three methods in terms of accuracy rate, recall rate, precision rate and F1.

In summary, the main advantages of the work presented in [35] are related to the proposition of a combined wireless network intrusion detection model based on deep learning. The authors conducted a comprehensive analysis of the achieved results, which included model performance and comparison with other available models. Furthermore, the authors acknowledge the main limitations of their work and indicated potential future research directions that should focus on improving the detection accuracy of the proposed detection model.

TABLE 19. Description of components involved in processing flow of IDS model [35].

Module	Steps	Explanation
Data Acquisition	Data set selection	Appropriate data set was chosen.
	Data set collection	Data set was gathered and read by Python's Scapy library.
	Timestamp tag	Data set was added a timestamp by Python's time module.
Data Pretreatment	Filtering and Sampling	Missing and redundant data filtered and sampled in the source.
	Feature mapping	Feature set defined according to existing wireless network intrusion. Data from the source database mapped to a predefined feature set.
	Character feature digitization	Characteristic features were converted to numerical features with the One-Hot Encoding.
	Normalized	Raw data was dropped into the [0, 1] interval with the min-max normalization method.
Data Analysis	Data set building	Feature database was divided into a training set and a testing set.
	Method selection	Feature learning based on combined DBN and SVM.
	Feature learning	Feature learning method was trained by the training feature set.
		Weight of the multi-restricted Boltzmann machine fine-tuned by sback-propagation algorithm.
	Performance evaluation	Intrusion detection algorithm was trained by SVM.
		Learning process stops when the detection was good enough.
		Intrusion detection algorithm and model were provided to the detection classification.
Detection Classification	Classification detection	Data set was classified and detected with ID model.
	Handle anomalies data	Anomalies data was processed and stored 2in the anomalies database.

Traffic classification and prediction are inevitable parts of the security aspects, also those that are based on the NN. Furthermore, analogously to the reliability algorithms, security solutions can also be enhanced by prediction, TE and QoE verification. All of these can boost malicious traffic detection and other components of IDS, such as: data analysis, data acquisition, and training of data. For example, TE facilitates flow classification and discovering malicious users.

V. EMERGING APPLICATIONS OF NEURAL NETWORKS IN COMMUNICATIONS AND NETWORKING

This section, differently to the previous sections, is not focused on using ML to solve issues in a particular networking aspects. Instead, V-A we analyze papers that

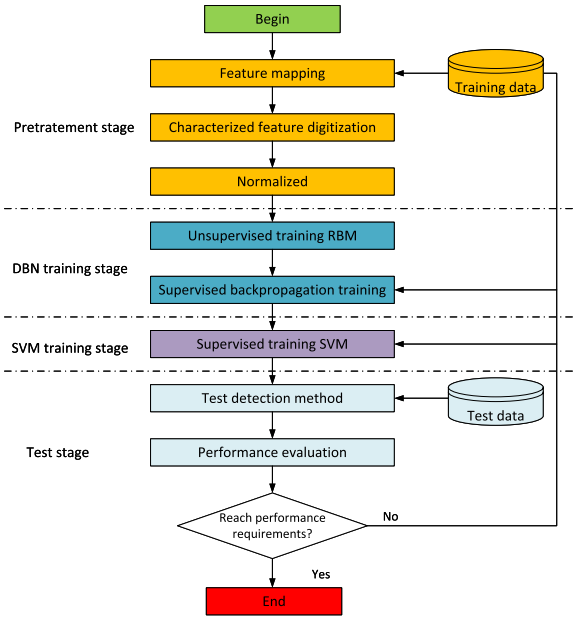


FIGURE 22. The flow chart of the combined detection method proposed in [35].

propose applying ML techniques to boost the deployment of emerging applications that utilize network infrastructure.

A. APPLICATION ORIENTED APPROACHES

Reduction of upstream delay in heterogeneous optical-wireless networks for Tactile Internet is an example of emerging application that can be supported by NN. Tactile Internet paradigm is expected to facilitate our everyday life in several and critical domains. Healthcare, and especially, remote surgery application is usually mentioned as a flagship example, while other popular services are industry, automation, transport, and robotic systems including autonomous driving, education, gaming, and any other applications based on visual and tactile experiences. One of the key characteristics of Tactile Internet is the extremely low and stable end-to-end latency at the level of milliseconds. A recent paper [37] presented during INCFOCOMM Poster session addresses this issue with the use of ML.

The paper is aimed at reducing the upstream delay in heterogeneous optical-wireless networks. ANN is deployed in the centralized controller to allocate bandwidth at the optical network unit where the aggregated upstream traffic is competing for optical resources. A predictive algorithm is proposed to classify the status of optical network segments and flexibly assign resources, and as a result, meet the requirements of emerging tactile applications. The bursty nature and changing packet length are the main impediments to apply any well-known approaches like constant credit, linear credit, Bayesian estimation, and even selected machine learning-based solutions, e.g. k-nearest neighbourhood algorithm (k-NN). Contrary, the proposed algorithm can effectively identify ON/OFF periods characterizing bursty traffic. Comprehensive experiments were conducted and reported to present advantages of the proposed solution comparing to the k-NN and the reference algorithm. The achieved advantages come from the fact that the utilized ANN is efficient in finding non-linear relationships between input and output features.

The only output feature of the model regards to the predicted ON/OFF period, while the input features have more sophisticated structure. It is assumed that traffic features are known by the centralized control entity based on the REPORT messages generated in consecutive polling cycles by the optical network unit integrated with wireless access point (ONU-AP). Reported values regard to the number of received packets and bytes, as well as incoming bytes, ONU-AP buffer's length, and polling cycle time boundaries. A single hidden layer is expected to predict if ON or OFF periods will be experienced by a particular ONU-AP in the upcoming polling cycle. Such a binary value is assigned to the mentioned output feature. This prediction is further used to estimate the predicted ONU-AP buffer's length based on the wireless traffic data rate known thanks to the assumed integration between optical and wireless domains.

TABLE 20. Assignment of ONU-AP to the classes based on the period type indicated by the ANN and expected congestion of the ONU-AP.

Assigned ONU-AP class	Predicted period	Buffer congestion expected
Class-A	ON	NO
Class-B	OFF	NO
Class-C	ON or OFF	YES

Then the proposed solution allocates bandwidth for each ONU-AP and the upcoming polling cycle. For this purpose, ONU-APs are classified to one of three classes based on the following inputs: the predicted ONU-AP's period (ON or OFF, output of the ANN), ONU-AP's current buffer length, estimated ONU-AP's buffer length, and the maximum bandwidth that can be potentially assigned to the ONU-AP.

The rationale is to flexibly assign resources to the ONU-AP depending on the predictions for a particular ONU-AP the next cycle will be ON or OFF as well as the estimation about the buffer's congestion. The ONU-AP will be potentially congested if the sum of ONU-AP's current and estimated buffer lengths is greater than the maximum bandwidth that can be potentially assigned to the ONU-AP. The assignment is summarized in Table 20. Then, the central controller, for ONU-APs in classes A and B resources are assigned to handle the overall buffer length (the only difference between A and B classes regards to the needed signalling), while for ONU-APs in class C an additional time slot for transmission is granted by an additional GATE message. The rationale is fully intuitive to immediately react and assign resources to the ONU-APs that are currently congested due to the bursty nature of Tactile Internet traffic. The full workflow of the proposed algorithm is briefly presented in Figure 23.

The effectiveness of the proposed solution is verified based on event-driven packet-level simulations assuming realistic network architecture and system load. The prediction accuracy was compared with the k-NN-based solution configured with three different k values. ANN provides stable accuracy at the level above 90% proving it can effectively identify ON/OFF periods, while k-NN accuracy never exceeds 80% and drops down even below 50% for traffic load with frequent changes between ON and OFF periods. The second set of experiments regards to the QoS parameters: average upstream delay and packet drop ratio. The results achieved by the proposed solution are compared to the classical limited-service Dynamic Bandwidth Algorithm (DBA) algorithm. In this case, both indicators are significantly improved, especially for heavy traffic, when elasticity in terms of resource allocation may bring additional advantages.

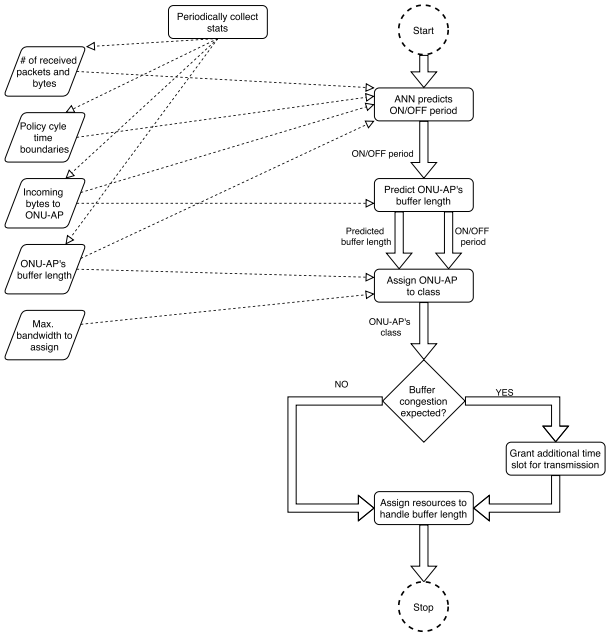


FIGURE 23. Workflow of the algorithm proposed in [37].

To sum up, the work [37] addresses an actual issue enabling the deployment of the emerging Tactile Internet in a heterogeneous optical-wireless networks with the use of ANN. Namely, the upstream delay of ONU-AP is reduced by the algorithm aimed at identifying ON/OFF periods of bursty traffic typical for tactile applications. Due to the short paper format (two pages presented during the poster session), the authors could not explain all the matters useful to fully understand the paper. Thus, we believe that the supplement provided below will facilitate perception of the paper for further readers.

One of the fundamentals that should be explained to readers not being experts in passive optical networks regards to the DBA problem and the corresponding algorithms. DBA is a process of assigning time-slots to Optical Line Terminal (OLT) to enable the access of multiple OLTs to a single Optical Network Unit (ONU) [38], where ONU is a device that connects the end users to the optical fiber network. It converts optical signals to electrical signals and provides the end user services, such as video, voice, and broadband. Moreover, the limited-service DBA algorithm considered as a reference is a well-known approach which assumes that ONU grants transmission time to OLT to satisfy it's demand until the defined maximum value is not exceeded. The REPORT and GATE messages are both Ethernet messages that are part of the Multipoint Control Protocol (MPCP) developed by the IEEE 802.3ah task force [39]. MPCP is used as a supporting protocol for bandwidth allocation schema between ONU and OLT in gigabit passive optical networks. Another issue that may rise reader's concerns regards to the maximum bandwidth that can be granted to the particular ONU-AP. This value results directly from the size of the transmission window assigned to the ONU-AP. The authors also had not the chance to explain the issue of optical-wireless integration in terms of ONU-AP node architecture [40].

In terms of results analysis, it may be concerned that the proposed solution is simultaneously compared with k-NN and the limited-service DBA algorithm using different performance indicators. It results from the fact that k-NN can be used solely to predict ON/OFF periods, and thus, it is reasonable to analyze the prediction accuracy. Simultaneously, the limited-service DBA algorithm is a complete solution aimed at dynamic resource assignment, so end-to-end transmission indicators are suitable to validate the performance of the solution proposed in [37]. One should also note that the limited-service DBA algorithms does not use k-NN. The dependencies between k-NN, DBA algorithm and the proposed solution in terms of the aims of each solution are presented in Table 21. Furthermore, Table 21 indicates performance metrics suitable to assess a particular solution.

Furthermore, thanks to the [37] we are able to identify the following research avenues that may contribute to the future development of ML supporting Tactile Internet applications. It will be interesting to verify why k-NN algorithm prediction accuracy of ON/OFF period is approximately constant in function of k. Analogously, the authors assumed a basic model of ANN with only a single hidden layer and binary output. Thus, it remains an open question if other ANN can achieve better accuracy in terms of ON/OFF period prediction. Finally, the evaluation performed in the [37] is based on the event-driven packet-level simulations. Some valuable insights regarding the real life deployments may be formulated by extending experiments to the testbed environment. For example, some additional delays while exchanging MPCP messages may have impact on the performance indicators measured in the experiments.

VI. COMPARATIVE ANALYSIS, INSIGHTS, LESSONS LEARNED AND RESEARCH OPPORTUNITIES

The class of NN methods comprises a wide variety of approaches and solutions. Thus, NN are applicable to

TABLE 21. Summarizing comparison of k-NN, DBA algorithm and proposed solution comprising indicators suitable for performance assessment.

Algorithm		Is performance indicator applicable			
Name	Aim	Prediction Accuracy Error	Prediction Error	Average Upstream Delay	Packet Drop Ratio
k-NN	Predict ON/OFF periods	YES	YES	NO	NO
DBA	Dynamically assign resources	NO	NO	YES	YES
Proposed solution	Predict ON/OFF periods	YES	YES	NO	NO
	Dynamically assign resources	NO	NO	YES	YES

numerous problems in communications. In Tables 22, 23 and 24 we summarize the works analyzed in this tutorial. The main aim is to provide a comparative analysis and some general conclusions. The tables group solutions based on the NN algorithm being used. We focus on the implementation of NN and properties of the proposed algorithm. These are the main themes common to all the analyses presented in our work. Different solutions have been used by the authors of the analyzed papers. Table 22 summarizes papers using basic NN methods, Table 23 regards to the works that use DNN subclass of NN. Finally, solutions based on the advance NN methods, such as DBN, RNN or CNN, are summarized in Table 24.

The comparative tables are divided into sections using horizontal lines. In the first section of each table, the comparison focuses on the general properties, taxonomy, and quality of the articles and writing. Then, the second section of each table compares the general properties of the NN method being used. Finally, in the last section, some more detailed properties of NN are the subject of comparative analysis.

The typical aims of using NN methods are prediction and detection. It is a consequence of the nature and typical applications of the supervised learning class, which comprises NN methods. However, NN-based solutions can also improve transmission characteristics as well as the efficiency of security and reliability mechanisms.

Regarding the strong aspects of various works, it is common to conduct valuable and interesting experiments that regard emerging topics. Papers are also often well-written and organized, while the content is supplemented with meaningful figures. Results are usually deeply analyzed. It is also worth noting, that NN methods are often combined with other techniques to provide a complete solution of the problem.

However, some typical drawbacks are also present in numerous works addressing NN in networking. The algorithms proposed in the papers are often described too briefly to be understandable by engineers who are not ML experts but are interested in applying the proposed solution. Simultaneously, authors are often focused on the issues related to the NN method that they neglect providing critical information about the problem and networking aspects. For example, the network topology and fundamental parameters are missing. Furthermore, performance parameters and anomaly patterns are sometimes not explained in a clear and unambiguous way.

Proposed NN methods are usually accurate, precise, and efficient when solving problems in different aspects of communications and networking. However, it should be noted that introducing NN method usually brings also additional computation and latency overhead. Additional computing resources are needed to train algorithms and make them operate. An additional threat regards also to potential failures, especially, as ML methods further depend on the input data that may be corrupted or even malicious. A separate issue regards the fact that all supervised learning solutions comprise a type of learning stage. Thus, it is critical to ensure proper training data as input for the designed solution.

A. LESSONS LEARNED FOR THE AUTHORS OF FUTURE PAPERS

Based on the comparative analysis, it seems that engaging a network specialist without the NN background in the process of writing the paper may be advantageous. Firstly, the specialist may verify if the text is easy to follow for readers and if the solution can be easily applied. Secondly, a person with a strong networking background will help with formulating practical validation scenarios and describing them in detail. Finally, the authors should carefully explain the results achieved by the proposed algorithms using metrics and criteria typical for telecommunications. Furthermore, some intuition on how these advantages have been achieved is also a valuable part of the work.

In writing this tutorial, we aimed to provide supplementary materials for existing papers to fill some gaps (e.g., proper explanation of models or handy summary of results) and give examples for the authors of future papers. Furthermore, based on the performed studies, we formulated the following theoretical and hypothetical recommendations for technologies that can advance the *state-of-the-art*. The former type of recommendation includes: developing new structures on NN to reflect the complexity of networking aspects, establishing theoretical frameworks to integrate NN principles with communication network theories, standardization

TABLE 22. Comparison of papers using basic NN.

	Paper [21]	Paper [22]	[30]	[37]
ML method	MCTS / ANN	NN / GPR / oBMM	NN	ANN
Aspect	Prediction	Prediction	TE	App-oriented
Strengths	EON instead of WDM, comprehensive analysis	prediction experiments complemented with routing experiments	improved speed	Emerging topic, well justified and stated
Weaknesses	-	insufficient description of used methods	NN as the additional step	Ambiguous performance indicators
Comment	deep analysis, well defined methods and simulation environment	ML predictors comparison and analysis of flow features impacting prediction accuracy	needs MILP optimizer time-consuming optimization for training	Limited description of background
Aim of using ML	selection of best DC and candidate path pair	elephant flows detection	improved speed	Reduce latency
Network type	core	core	core	access
Network technology	EON	SDN	IP/MPLS	Heterogeneous optical-wireless
Pros of using ML	effective handling of the client's request	improved routing	faster solution lower packet loss	Recognition of ON and OFF
Cons of using ML			additional computing	Limited practical usability
Result of using ML			faster LSP design lower IP traffic loss	
Performance indicator	average request blocking percentage	true positive rate true negative rate normalized flow completion time	time, packet loss	Prediction accuracy, delay, packet loss
Input for ML	DC utilization optical resource utilization	first few flows features	traffic demands	MPCP traces
Features used			traffic pattern	
Reward			packet loss reduction	
Output of ML	combination of cloud data centers and candidate path pairs for provisioning services related to specific requests	flow size	LSP structure	Indicator about ON/OFF period
Training data	private dataset	public dataset and one academic building	output from MILP optimizer	Self-generated
Training process	backpropagation	backpropagation	comparison with MILP results	Preliminary stage
Complexity	medium	low	related to MILP complexity	low
Scalability	-	-	MILP-based	linear

TABLE 23. Comparison of papers using basic DNN.

	Paper [31]	[33]	[34]
ML method	DNN	Density-based clustering and DNN	DNN
Aspect	QoE	Reliability	Security
Strengths	Analysis of the results	Well organized, written, and visualized	Well justified, good organization
Weaknesses	Not relevant	Ambiguity regarding anomalies patterns	Results analysis and validation
Comment		Novel combining supervised and unsupervised learning	Novel combining SL and UL
Aim of using NN	Validate the effectiveness of QoE prediction	Detect anomalies	Intrusion detection
Network type	Mobile	Metro/Core	Versatile
Network technology	Wireless	Optical	Versatile
Pros of using NN	Analyzing large scale datasets	Efficient for rare anomalies	Reduction of non-symmetric data dimensionality, training time, and analytical overheads
Cons of using NN	Additional computing	Need for real-time monitoring, parametrized	Not able to handle zero-day attacks
Performance indicator	Visual quality, loading, stalling and overall quality	Anomaly detection	Attack detection
Input for NN	dataset including the QoE data of eighty thousand samples	network measurements	Network parameters
Features used	many, e.g. video rate, buffer delay		
Output of NN	Predicted quality	Anomaly and its location	Attack detection
Training data	Prepared data	Self-generated anomalies in testbed	KDD Cup '99, NSL-KDD datasets
Training process	Preliminary stage	Preliminary stage	Error minimization function
Complexity	Medium/low	Low	low
Scalability	Good scalability	Linear	
Robustness	Users perception	Prone to the failures of measurement system	No single point of failure
Versatility		Yes, detection of anomalies causing partial service deterioration	

TABLE 24. Comparison of papers using advanced NN methods.

	Paper [20]	Paper [23]	Paper [14]	[35]
ML method	LSTM RNN	DBN / GM	CNN, RNN, GP	Multi-layer RBM
Aspect	Prediction	Prediction	QoE	Security
Strengths	consideration of traffic matrix	network traffic decomposition	First proposal to predict QoE	Well organized and visualized
Weaknesses	results analysis and validation	not defined network topology and overall network parameters	Not relevant	Unclear anomaly patterns
Comment	well defined prediction problem and algorithm	prediction of two separate traffic components after decomposition		Novel combining DBN and SVM
Aim of using ML	traffic matrix prediction	classical traffic prediction	QoE detector	Intrusion detection
Network type	core	core	Cloud	Access
Network technology	SDN	wireless	Wireless	Wireless
Pros of using ML	potential to avoid network congestion	potential to use proactive methods for network control	Excellent prediction results	Efficient for detecting wireless network intrusion behavior
Cons of using NN			Additional computing	Small size of training dataset
Performance indicator	prediction error	prediction accuracy temporal relative error spatial relative error	QoE accuracy precision, recall	Intrusion detection performance
Input for ML	traffic volume	traffic volume	video data specifically prepared	Network parameters
Output of ML	traffic vector predictor	traffic value predictor	Predicted QoE of video transmission	Intrusion detection
Training data	pubic dataset	private dataset	Prepared data	NSL-KDD dataset
Training process	BPTT	layer-wise greedy strategy	Preliminary stage	Preliminary stage (SVM)
Complexity	medium	medium	Medium	low

of metrics to effectively reflect the applicability of NN in communication aspect, and finally, improving interpretability and explainability of NN-based algorithms.

On the other hand, hypothetical recommendations can be summarized as follows: utilize proper simulation and modeling techniques to explore NN performance in hypothetical networking scenarios, conduct experiments in environments reflecting real-world network infrastructures, develop techniques enabling generation and augmentation of communication network datasets for NN training and validation purposes, and finally, investigate adaptive learning methods to improve NN performance in the dynamic environment of communication networks.

B. RESEARCH OPPORTUNITIES

The NN-based solutions should now be developed towards practical applications. Thus, insights and considerations explaining how to deploy the proposed mechanism are expected. Furthermore, the performance and complexity overhead should be carefully studied as well as the technical requirements of the solution.

It is also practically justified to conduct research towards building a NN-based comprehensive system that will be able to address several aspects of communications and networking in an integrated manner. Such a system should comprise a prediction module providing an input for TE mechanisms that will utilize NN to effectively route the network traffic. On the other hand, NN-based mechanisms aimed at ensuring QoE should also introduce application-level input for traffic management. Accurate predictions should also facilitate the process of securing the network and making it more reliable. It can be achieved, as predictions of expected network conditions, can be used as a reference while detecting any security or reliability issues. More precisely, finding the difference between the current and expected network state can improve the accuracy of NN-based classification while detecting intrusions or anomalies detection. Finally, decisions and countermeasures provided by security and reliability modules will be introduced into the network with the use of TE the module. To sum up, it should be considered as an attractive and challenging research avenue to ensure effective integration between numerous NN-based solutions working in different modules of such a complex system.

Knowledge about confidence and uncertainty of a particular NN-based algorithm is another factor important from the business point of view. This information may be an input to the process related to risk management. Simultaneously, the susceptibility to corrupted and malicious training data should be carefully considered. From that perspective, explainable NN-based solutions should be one of the research directions. It will be valuable to propose a framework ensuring transparency of decisions suggested by the NN algorithms. With the use of such a framework, organizations should be able to build the trust required to adopt NN-powered solutions. The ideal framework should also be mature enough to ensure model accountability and mitigate compliance, legal and reputational risks.

Another research avenue that may facilitate practical applications of NN algorithms is to develop versatile solutions that ensure sufficient accuracy and performance in numerous network topologies and technologies. One of the examples from the communications field is to use Graph Neural Netowrks (GNN) to optimize computer network parameters independently of their topology. A more general example of efforts is to combine NN techniques with other ML algorithms. The first step aims to identify a particular problem and infrastructure and properly prepare the second stage, in which the actual problem is solved. Development of sophisticated ML models will be facilitated by compilers optimized for ML purposes. It also opens other interesting research directions. Firstly, to investigate, assess and compare such compilers. Secondly, to prepare precise and comprehensively justified guidelines for the usage of these compilers.

Finally, for NN-based solutions that belong to the class of supervised learning, the quality of training data is critical. The most reliable approach is to use real-life data collected in the existing networks. However, it requires cooperation between data owners, network operators, industry researchers and scientists. In some cases, involved parties may even have opposite requirements. The most attractive and valuable information is, usually, simultaneously critical for the business owner and may comprise personal data. Thus, it is an open issue to find cooperation schemas that will be beneficial for everyone.

VII. CONCLUSION

Conventional programming instructs a computer program how to act. In contrast, ML-based solutions inform how to learn about the system. In supervised learning approaches, including also NN, the learning phase must be directly addressed. This process is critical for accuracy and often demanding. Training data must ensure sufficient volume, variety, and quality. Furthermore, a proper preprocessing phase is typically required. Usually, supervised learning approaches are dedicated to the prediction and detection processes.

Despite those obstacles and limitations, NN-based solutions are simultaneously popular and effective in a wide range of aspects of communications and networking. Valuable predictions and detections are performed solely by NN algorithms in terms of, for example, security or QoE. Thus, comprehensive and complete solutions can be composed with the use of NN to address issues in different aspects of communication and networking. For example, NN-based predictions can be used by TE algorithms to, on the one hand, improve the performance and meet QoE requirements, while on the other hand, to ensure proper tools to improve reliability and security of the network. Moreover, predictions can also improve the efficiency and accuracy of the

NN-based reliability- and security-oriented solutions by providing baselines and references.

Furthermore, NN are often combined with other approaches to design complex solutions. Available IT infrastructure as well as modern network architectures create favorable conditions to deploy the proposed solutions and take full advantage of them. Virtualized cloud environments offer powerful computing platforms comprising specialized Graphical and Tensor Processing Units to enable parallel operations and facilitate the training process. Software Defined Networking and Network Function Virtualization paradigms provide global control over the network infrastructure, which facilitates the implementation process.

In this paper, we supplement the theoretical tutorial with a practical analysis of selected works applying NN in various aspects of networking, network types and technologies. The main aim is to provide an example-based tutorial of works that apply NN in the field of communications and networking. We provide original content and in-depth analysis in the form of additional figures, block diagrams and summarizing tables. At the end of each section that regards a particular networking aspect, we provide some insight into how these areas can interact with each other in the context of NN. We also ensure comparative analysis of papers among all the communication and networking aspects. Comparative tables and conclusions along with provided studies may be helpful in future research on NN-based solutions for communications.

REFERENCES

[1] P. Borylo, E. Biernacka, J. Domżał, B. Kądziołka, M. Kantor, K. Rusek, M. Skała, K. Wajda, R. Wójcik, and W. Zabek, "A tutorial on reinforcement learning in selected aspects of communications and networking," *Comput. Commun.*, vol. 208, pp. 89–110, Aug. 2023.

[2] A. Zappone, M. Di Renzo, and M. Debbah, "Wireless networks design in the era of deep learning: Model-based, AI-based, or both?" *IEEE Trans. Commun.*, vol. 67, no. 10, pp. 7331–7376, Oct. 2019.

[3] M. Chen, U. Challita, W. Saad, C. Yin, and M. Debbah, "Artificial neural networks-based machine learning for wireless networks: A tutorial," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3039–3071, 4th Quart., 2019.

[4] N. Ahad, J. Qadir, and N. Ahsan, "Neural networks in wireless networks: Techniques, applications and guidelines," *J. Netw. Comput. Appl.*, vol. 68, pp. 1–27, Jun. 2016. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804516300492

[5] A. He, K. Kyoon Bae, T. R. Newman, J. Gaeddert, K. Kim, R. Menon, L. Morales-Tirado, J. J. Neel, Y. Zhao, J. H. Reed, and W. H. Tranter, "A survey of artificial intelligence for cognitive radios," *IEEE Trans. Veh. Technol.*, vol. 59, no. 4, pp. 1578–1592, May 2010.

[6] M. A. Ridwan, N. A. M. Radzi, F. Abdullah, and Y. E. Jalil, "Applications of machine learning in networking: A survey of current issues and future challenges," *IEEE Access*, vol. 9, pp. 52523–52556, 2021.

[7] F. Musumeci, C. Rottondi, G. Corani, S. Shahkarami, F. Cugini, and M. Tornatore, "A tutorial on machine learning for failure management in optical networks," *J. Lightw. Technol.*, vol. 37, no. 16, pp. 4125–4139, Aug. 2019.

[8] J. Xie, F. R. Yu, T. Huang, R. Xie, J. Liu, C. Wang, and Y. Liu, "A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 393–430, 1st Quart., 2019.

[9] Z. M. Fadlullah, F. Tang, B. Mao, N. Kato, O. Akashi, T. Inoue, and K. Mizutani, "State-of-the-art deep learning: Evolving machine intelligence toward tomorrow's intelligent network traffic control systems," *IEEE Commun. Surveys Tuts.*, vol. 19, no. 4, pp. 2432–2455, 4th Quart., 2017.

[10] R. Boutaba, M. A. Salahuddin, N. Limam, S. Ayoubi, N. Shahriar, F. Estrada-Solano, and O. M. Caicedo, "A comprehensive survey on machine learning for networking: Evolution, applications and research opportunities," *J. Internet Services Appl.*, vol. 9, no. 1, p. 16, Jun. 2018, doi: 10.1186/s13174-018-0087-2.

[11] M. Hassoun and A. Hassoun, *Fundamentals of Artificial Neural Networks* (A Bradford Book). Cambridge, MA, USA: MIT Press, 1995. [Online]. Available: https://books.google.pl/books?id=Otk32Y3QkxQC

[12] O. Gupta and R. Raskar, "Distributed learning of deep neural network over multiple agents," *J. Netw. Comput. Appl.*, vol. 116, pp. 1–8, Aug. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1084804518301590

[13] N. M. Rezk, M. Purnaprajna, T. Nordström, and Z. Ul-Abdin, "Recurrent neural networks: An embedded computing perspective," *IEEE Access*, vol. 8, pp. 57967–57996, 2020.

[14] M. Lopez-Martin, B. Carro, J. Lloret, S. Egea, and A. Sanchez-Esguevillas, "Deep learning model for multimedia quality of experience prediction based on network flow packets," *IEEE Commun. Mag.*, vol. 56, no. 9, pp. 110–117, Sep. 2018.

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

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

[17] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "Pytorch: An imperative style, high-performance deep learning library," in *Proc. Adv. Neural Inf. Process. Syst.*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32. Red Hook, NY, USA: Curran Associates, Inc., 2019, pp. 1–7. [Online]. Available: https://proceedings.neurips.cc/paper/2019/file/bdbca288fee7f92f2bfa9f7012727740-Paper.pdf

[18] M. Abadi et al. (2015). *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. [Online]. Available: https://www.tensorflow.org/

[19] I. Babuschkin et al. (2020). *The DeepMind JAX Ecosystem*. [Online]. Available: http://github.com/deepmind

[20] A. Azzouni and G. Pujolle, "NeuTM: A neural network-based framework for traffic matrix prediction in SDN," in *Proc. NOMS IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2018, pp. 1–5.

[21] M. Aibin, "Traffic prediction based on machine learning for elastic optical networks," *Opt. Switching Netw.*, vol. 30, pp. 33–39, Nov. 2018, doi: 10.1016/j.osn.2018.06.001.

[22] P. Poupart, Z. Chen, P. Jaini, F. Fung, H. Susanto, Y. Geng, L. Chen, K. Chen, and H. Jin, "Online flow size prediction for improved network routing," in *Proc. IEEE 24th Int. Conf. Netw. Protocols (ICNP)*, Nov. 2016, pp. 1–6.

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

[24] S. Uhlig, B. Quoitin, J. Lepropre, and S. Balon, "Providing public intradomain traffic matrices to the research community," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 36, no. 1, pp. 83–86, Jan. 2006, doi: 10.1145/1111322.1111341.

[25] M. Barabas, G. Boanea, A. B. Rus, V. Dobrota, and J. Domingo-Pascual, "Evaluation of network traffic prediction based on neural networks with multi-task learning and multiresolution decomposition," in *Proc. IEEE 7th Int. Conf. Intell. Comput. Commun. Process.*, Aug. 2011, pp. 95–102.

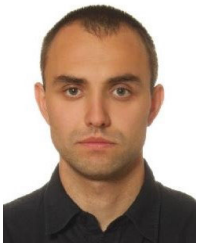
[26] M. Jinno, B. Kozicki, H. Takara, A. Watanabe, Y. Sone, T. Tanaka, and A. Hirano, "Distance-adaptive spectrum resource allocation in spectrum-sliced elastic optical path network," *IEEE Commun. Mag.*, vol. 48, pp. 138–145, 2010. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/5534599

[27] M. Klinkowski and K. Walkowiak, "On the advantages of elastic optical networks for provisioning of cloud computing traffic," *IEEE Netw.*, vol. 27, no. 6, pp. 44–51, Nov. 2013. [Online]. Available: https://ieeexplore.ieee.org/abstract/document/6678926

[28] C. Politi, V. Anagnostopoulos, C. Matrakidis, A. Stavdas, A. Lord, V. López, and J. P. Fernández-Palacios, "Dynamic operation of flexi-grid OFDM-based networks," in *Proc. OFC/NFOEC*, Mar. 2012, pp. 1–3.

[29] T. Benson, A. Akella, and D. A. Maltz, "Network traffic characteristics of data centers in the wild," in *Proc. 10th ACM SIGCOMM Conf. Internet Meas.*, Nov. 2010, pp. 267–280, doi: 10.1145/1879141.1879175.

[30] M. Dale, H. Ferra, and R. Palmer, "Fast MPLS network optimisation using machine learning," in *Proc. TENCON IEEE Region 10 Conf.*, Nov. 2005, pp. 1–6.
[31] X. Tao, Y. Duan, M. Xu, Z. Meng, and J. Lu, "Learning QoE of mobile video transmission with deep neural network: A data-driven approach," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 6, pp. 1337–1348, Jun. 2019.
[32] C. K. I. Williams and D. Barber, "Bayesian classification with Gaussian processes," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 12, pp. 1342–1351, Dec. 1998.
[33] X. Chen, B. Li, R. Proietti, Z. Zhu, and S. J. B. Yoo, "Self-taught anomaly detection with hybrid unsupervised/supervised machine learning in optical networks," *J. Lightw. Technol.*, vol. 37, no. 7, pp. 1742–1749, Apr. 2019.
[34] N. Shone, T. N. Ngoc, V. D. Phai, and Q. Shi, "A deep learning approach to network intrusion detection," *IEEE Trans. Emerg. Topics Comput. Intell.*, vol. 2, no. 1, pp. 41–50, Feb. 2018.
[35] H. Yang, G. Qin, and L. Ye, "Combined wireless network intrusion detection model based on deep learning," *IEEE Access*, vol. 7, pp. 82624–82632, 2019.
[36] K. A. Simpson, S. Rogers, and D. P. Pezaros, "Per-host DDoS mitigation by direct-control reinforcement learning," *IEEE Trans. Netw. Service Manage.*, vol. 17, no. 1, pp. 103–117, Mar. 2020.
[37] L. Ruan, S. Mondal, and E. Wong, "Machine learning based bandwidth prediction in tactile heterogeneous access networks," in *Proc. IEEE INFOCOM Conf. Comput. Commun. Workshops (INFOCOM WKSHPS)*, Apr. 2018, pp. 1–2.
[38] P. P. Sahu, Ed., *Fundamentals of Optical Networks and Components*. Boca Raton, FL, USA: CRC Press, Jul. 2020.
[39] D. Sudhir, Ed., *IP Over WDM: Building the Next-Generation Optical Internet*. Hoboken, NJ, USA: Wiley, Jun. 2004.
[40] A. M. Abdalla, J. Rodriguez, I. Elfergani, and A. Teixeira, Ed., *Optical and Wireless Convergence for 5G Networks*. Hoboken, NJ, USA: Wiley, Sep. 2019.



PIOTR BORYLO received the Doctorate and D.Sc. degrees in telecommunications from the AGH University of Science and Technology, Krakow, Poland, in 2016 and 2023, respectively. Since 2015, he has been a Faculty Member of the Institute of Telecommunications, where he is currently an Associate Professor. He is the coauthor of numerous research articles, including works prepared in international teams from Italy, France, and Canada. His research interests include software defined networks, network function virtualization, cloud computing, 5G core networks, emerging network services, and network optimization.

EDYTA BIERNACKA received the M.S. and Ph.D. degrees in telecommunications from the AGH University of Science and Technology, Krakow, Poland, in 2012 and 2021, respectively. Currently, she is an Assistant Professor with the Institute of Telecommunications, AGH University of Science and Technology. Her research interests include aspects of multi-layer networks, elastic optical networks, and software defined networks.

JERZY DOMZAL received the M.S., Ph.D., and D.Sc. degrees in telecommunications from the AGH University of Science and Technology, Krakow, Poland, in 2003, 2009, and 2016, respectively. He is currently an Associate Professor with the Institute of Telecommunications, AGH University of Science and Technology. He is the author or coauthor of many technical articles, two patents, and two books. His research interests include optical networks and services for future internet. International trainings: Universitat Politecnica de Catalunya, Barcelona, Spain, in April 2005; Universidad Autnoma de Madrid, Madrid, Spain, in March 2009; and Stanford University, USA, in May 2012 and June 2012.

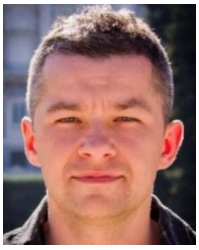
BARTOSZ KADZIOLKA received the B.S. and M.S. degrees in electronic and telecommunication engineering from the AGH University of Science and Technology, Krakow, Poland, in 2017 and 2019, respectively, where he is currently pursuing the Ph.D. degree with the Institute of Telecommunications. His research interests include software-defined networking, traffic engineering, and network optimization.

MIROSLAW KANTOR is currently an Assistant Professor with the Department of Telecommunications, AGH University of Science and Technology, Krakow. He participated in several European Projects and in projects funded by Polish funding institutions. He is the author or coauthor of over 60 science publications, including one academic script, two book chapters, and one internet draft. His research interest includes cybersecurity aspects of telecommunication networks.

KRZYSZTOF RUSEK is currently an Assistant Professor with AGH and a Data Scientist with Barcelona Neural Networking Center, where he defended his Ph.D. thesis on queuing theory, in 2016. Prior to that, he has was a System Administrator and a Machine Learning Engineer in the research group focused on processing and protection of multimedia content. His main research interests include performance evaluation of telecommunications systems, machine learning, and data mining. Currently, he is working on the applications of graph neural networks and probabilistic modeling for performance evaluation of communications systems and data mining in astronomy.

MACIEJ SKALA received the B.S. and M.S. degrees in electronic and telecommunication engineering from the AGH University of Science and Technology, Krakow, Poland, in 2015 and 2017, respectively, and the M.S. degree in mobile communications from Telecom Paris, Paris, France, in 2017. He is currently pursuing the Ph.D. degree in information and communication technology with the AGH University of Science and Technology. His research interests include network optimization, cloud computing, and software-defined networking.

KRZYSZTOF WAJDA (Member, IEEE) received the M.Sc. degree in telecommunications and the Ph.D. degree from the AGH University of Science and Technology (AGH-UST), Krakow, Poland, in 1982 and 1990, respectively. He spent a year with Kyoto University and half a year in CNET, France. He is currently an Assistant Professor with the AGH University of Science and Technology. He actively participated in a few international projects, including: Leonardo da Vinci (JOINT and ET-NET), COST 242, ACTS 038 BBL, Copernicus ISMAN, IST LION, IP NOBEL, NoE e-photon/ONe(+), BONE, SmoothIT, and SmartenIT. He has also served as a Reviewer of journals, including, e.g., *IEEE Communications Magazine*, *Telecommunications Systems*, and international conferences. He has been a Consultant to major Polish telecommunication companies. He is the author/coauthor of six books and above 150 technical articles. His research interests include traffic management, performance evaluation, network reliability, control plane, management systems, and network services.



ROBERT WOJCIK received the Ph.D. and D.Sc. (Hons.) degrees in telecommunications from the AGH University of Science and Technology, Krakow, Poland, in 2011 and 2019, respectively. Currently, he is a Professor with the Institute of Telecommunications, AGH. He is the coauthor of more than 70 research publications, including 21 research articles in JCR journals and several patents. He was involved in several international EU-funded scientific projects, including: SmoothIT, NoE BONE and Euro-NF and smaller, and national projects. He was the leader of three national science projects. His current research interests include multipath routing, flow-aware networking, quality of service, and network neutrality.

WOJCIECH ZABEK received the M.S. degree in IP traffic management from Rzeszw University of Technology, Rzeszw, Poland, in 2004. He is currently pursuing the Ph.D. degree with the Institute of Telecommunications, AGH University of Science and Technology, Krakow, Poland. Since 2010, he has been a Cisco Network Academy Trainer within the CNAP Program (ID:7001212). Since 2012, he has also been an Academic Teacher with the University of Computer Engineering and Telecommunication, Kielce, Poland. His research interests include the area of multi-layer routing in IP networks, congestion prediction mechanisms, and quality of service assurance.