electronics

MDPI

Review

A Survey of Machine Learning Techniques for Video Quality Prediction from Quality of Delivery Metrics

Obinna Izima *,†, Ruairí de Fréin †and Ali Malik †

School of Electrical and Electronic Engineering, Technological University Dublin, D07 XT95 Dublin, Ireland; ruairí.defrein@tudublin.ie (R.d.F.); ali.malik@tudublin.ie (A.M.)

* Correspondence: izimaobinna@gmail.com

† These authors contributed equally to this work.

Abstract: A growing number of video streaming networks are incorporating machine learning (ML) applications. The growth of video streaming services places enormous pressure on network and video content providers who need to proactively maintain high levels of video quality. ML has been applied to predict the quality of video streams. Quality of delivery (QoD) measurements, which capture the end-to-end performances of network services, have been leveraged in video quality prediction. The drive for end-to-end encryption, for privacy and digital rights management, has brought about a lack of visibility for operators who desire insights from video quality metrics. In response, numerous solutions have been proposed to tackle the challenge of video quality prediction from QoD-derived metrics. This survey provides a review of studies that focus on ML techniques for predicting the QoD metrics in video streaming services. In the context of video quality measurements, we focus on QoD metrics, which are not tied to a particular type of video streaming service. Unlike previous reviews in the area, this contribution considers papers published between 2016 and 2021. Approaches for predicting QoD for video are grouped under the following headings: (1) video quality prediction under QoD impairments, (2) prediction of video quality from encrypted video streaming traffic, (3) predicting the video quality in HAS applications, (4) predicting the video quality in SDN applications, (5) predicting the video quality in wireless settings, and (6) predicting the video quality in WebRTC applications. Throughout the survey, some research challenges and directions in this area are discussed, including (1) machine learning over deep learning; (2) adaptive deep learning for improved video delivery; (3) computational cost and interpretability; (4) self-healing networks and failure recovery. The survey findings reveal that traditional ML algorithms are the most widely adopted models for solving video quality prediction problems. This family of algorithms has a lot of potential because they are well understood, easy to deploy, and have lower computational requirements than deep learning techniques.

Keywords: machine learning; quality of delivery; video streaming

Citation: Izima, O.; de Fréin, R.; Malik, A. A Survey of Machine Learning Techniques for Video Quality Prediction from Quality of Delivery Metrics. *Electronics* **2021**, *10*, 2851. https://doi.org/10.3390/electronics10222851

Academic Editor: DanielGutiérrez Reina

Received: 18 October 2021
Accepted: 15 November 2021
Published: 19 November 2021

1. Introduction

Machine learning (ML) is a subset of artificial intelligence (AI) that enables computers to learn on their own to deliver predictions or solutions based on previous experiences. Deep learning (DL) is a subset of ML, which basically entails a neural network with multiple layers that attempts to mimic the human brain, enabling it to learn from massive amounts of data [1,2]. DL automates most of the steps involved in the feature extraction process. ML delivers insights to a wide range of disciplines, including computer vision [3], speech recognition [4,5], software defined networks (SDNs) [6], communication networks [7], and Internet of Things (IoT) [8]. In recent years, there has been a rapid adoption of ML techniques to solve practical problems in computer networks [9]. In this paper, we provide a review of recent applications of ML techniques that use quality of delivery (QoD) measurements for the prediction of video quality. QoD measurements capture the end-to-end performances of network services [10].

Video traffic makes up the largest portion of all IP traffic [11], which puts pressure on network operators to manage their resources efficiently while meeting customer expectations. Traditionally, video quality assessments (VQA) and predictions have been conducted through subjective [12] and objective [13] means. The subjective approach relies on quantifying the experience of the end users. This quality of experience (QoE)-driven network traffic management relies on monitoring and predicting application-level or QoD performance in terms of video key performance indicators (KPIs), as they affect the end users' experience. The International Telecommunication Union (ITU) defines QoE as the overall acceptability of an application or service, as perceived subjectively by the end user [14]. A major challenge with video streaming is that there is no unified way to measure the QoE [15]. This has given rise to research efforts that investigate QoE models based on network statistics. In fact, QoE modeling can assist in identifying different KPIs for different categories of user. For instance, a network provider may be interested in rebuffering, quality switches, and how these artifacts impact the video stream. This will help quantify the effect on the user's QoE, and how these in turn may be related to network parameters such as delay or jitter. According to the authors of [16], subjective user studies provide reliable evaluations, but are costly, time consuming, and are not suitable for real-world applications. Objective VQA approaches place an emphasis on mathematical modeling aimed at providing a quality score that closely resembles the perceived image/video quality. Although objective VQA metrics such as peak signal to noise ratio (PSNR) [17] and structural similarity index (SSIM) [18] are fast and relatively easy to implement, these measurements do not always reflect the end user's experience [18,19]. Unlike those methods, our focus in this paper is on network performance, as opposed to user experience.

Due to the inherent drawbacks in subjective and objective VQA approaches, researchers have turned their focus to ML-based video quality prediction. This shift has led to numerous studies that leveraged ML models for mapping network measurements to the streamed video quality. Some studies considered quality of service (QoS) metrics in video quality prediction. An application or service's QoS is a set of technology employed on a network to make sure it can operate reliably, even when the network's resources are limited [20]. According to the ITU-T Rec. E.800 [21], QoS can be defined as the "totality of characteristics of a telecommunications service that bear on its ability to satisfy stated and implied needs of the user of the service." From the network perspective, the European Telecommunications Standards Institute (ETSI) in its technical report, defines QoS as, "The ability to segment traffic or differentiate between traffic types in order for the network to treat certain traffic differently from others" [22]. Essentially, QoS refers to a group of technology that operates together on a network to ensure that it can reliably run high-priority applications and traffic even when network capacity is constrained. This technology achieves this by providing differentiated handling and capacity allocation to certain network traffic flows. These traffic flows correspond to service types, as QoS measurements are biased by the service under consideration. This offers the network administrator the capability to manage the order in which packets are processed and an ability to throttle the bandwidth available to the service. These QoS mechanisms in IP routers and monitoring protocols take a service view. They are affected by the services considered. QoS measurements for one service may not be relevant for another one. It is important to note that network monitoring and measurements are not always aimed at determining the quality of a particular service or application. In some cases, the aim is to determine the network's overall health. These QoS measurements are normally taken far away from the end users. As a result, they do not provide information that can be easily translated to another service.

QoD is concerned with the quality of the data delivery process [23]. It relates to the ability of the network and transport stacks to ensure quality data delivery. The authors of [24] make the case that QoD is correlated with the network stack's inability to deliver data in a reliable manner. Unlike QoS, QoD is service agnostic. It captures the end-to-

end performance of the data delivery process of a network. For video applications, QoD measurements capture the capability of ensuring reliable delivery of the video frames. Some common measures of QoD include packet delivery delays from the source to the destination, i.e., transport delays and queuing delays, packet loss, jitter, and throughput. For video applications, limited QoD may result in frames being delayed, which could lead to non-smooth playback for the client. Some examples of video artifacts introduced as a result of limited QoD include jitter, stalls or freezing, and jerkiness. Figure 1 illustrates the scope and differences between QoD, QoS, and QoE.

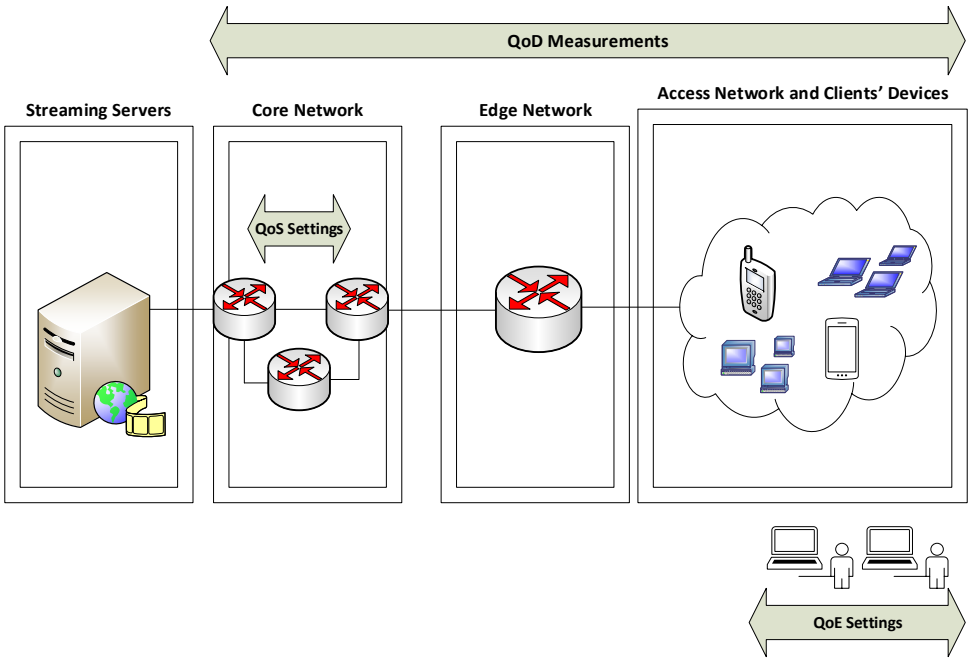

Figure 1. QoD measurements capture the end-to-end performance of the data delivery process from the core network to access networks and client devices. QoS settings are applied at the core IP routers to provide differentiated handling and capacity allocation to certain network traffic flows. QoE evaluations are conducted on a subjective basis. The mean opinion score (MOS) or perceptual evaluation of speech quality (PESQ), for example, is then calculated for these evaluations.

With the widespread adoption of end-to-end encryption for privacy and digital rights management reasons, operators lack insights into video quality metrics, such as startup delays, resolution, and stalling events, which are needed to accurately predict the video quality and drive optimum resource management. HTTP Adaptive Streaming (HAS) is becoming the de facto method for video streaming. This introduces some extra dynamics that pose a challenge for network operators attempting to predict the performances of their applications. The situation is further exacerbated by the different design approaches used in adaptive bitrate (ABR) algorithms employed in HAS platforms. A range of solutions have been proposed and validated that can predict a video application's stream quality in terms of KPIs or overall QoE, from the QoD measurements. Based on the data collected, these solutions propose models that map QoD measurements to QoE and/or video quality KPIs using ML techniques. In this paper, we provide a review of research efforts that utilized QoD measurements in ML-enabled video quality prediction.

1.1. Survey Methodology

We reviewed the relevant literature in the field published between 2016 and 2021. We focused mainly on papers from databases and publishers such as IEEE Xplore, Elsevier, MDPI, Nature, ACM, Springer, and ArXiv. We have reviewed more than 200 papers on various ML applications in QoD prediction for video streaming services. This highlights that this is a very active area of research given the growth of video-related IP traffic. We

selected papers for analysis and review based on (1) recent contributions in the area, (2) ML applications for QoD predictions, (3) the actively researched family of ML used for video services, (4) challenges in applying ML and DL techniques in video streaming and suggested resolutions, and (5) the computational requirements and feasibility of the approaches.

1.2. Scope

The rapid advances in network-level operations, such as SDNs, adaptive codec development, resource allocation, and slicing of networks, highlight the need for a new assessment on how to measure QoD. A review of the most promising future directions in this area is therefore needed. In response, this survey paper provides a systematic review on recent breakthroughs in video quality prediction that leverage QoD measurements—from task formulation, QoD measurements utilized, ML algorithms to datasets, to future research directions. The review includes aspects such as video quality KPIs, model assumptions, research key findings, and the ML learning paradigms employed.

Figure 2 shows the scope of this review paper. We provide a review of the recent trends in the application of ML algorithms in the prediction of video streaming quality using QoD-derived metrics from 2016 to 2021. In recent years, there has been an emphasis on end-to-end encryption, HAS streaming applications, video streaming with SDNs, and technological advancements in broadband and wireless networks. This survey focuses on contributions in these areas that deployed ML models using QoD statistics for video quality prediction. This survey also reviews contributions that predicted video quality under QoD impairments or in limited QoD settings and in WebRTC applications. A large number of publications on video quality prediction using ML exist. We focus on the research objectives, ML learning algorithms used, and datasets. We also provide crucial insights into the methodologies used, their benefits, limitations, and applicability to real-world networking scenarios. The contributions of this paper are outlined as follows:

- An overview of ML applications in networking. This includes a review of related surveys, the learning paradigms used, and the applications considered.
- A background overview on video streaming. This includes the evolution of video streaming, modeling video quality, and common issues encountered in video streaming protocols.
- A review of the applications of ML techniques for predicting QoD metrics with the aim of improving video quality. We also provide a review of works that leveraged ML algorithms for video quality predictions from QoD measurements. We discuss the ML techniques used in these studies, and analyze their benefits and limitations. Figure 2 highlights the scope and areas considered in this survey.
- A discussion of the future challenges and opportunities in the use of ML techniques in video streaming applications.

The rest of this paper is organized as follows. Section 2 provides an overview of ML and DL algorithms. Section 2 also reviews some related surveys on the applications of ML in networking. Section 3 provides an overview on the background of video streaming. This section also discusses the structure of videos, an overview on the evolution of video streaming, and common protocols deployed over the years in the field of video streaming. Finally, we discuss some common challenges faced in video streaming and highlight some common video quality metrics. Section 4 surveys ML-based techniques for predicting video quality from QoD measurements in the research literature. Section 5 provides some discussion on the findings of the survey. Section 5 concludes the review by outlining several open research questions in the field.

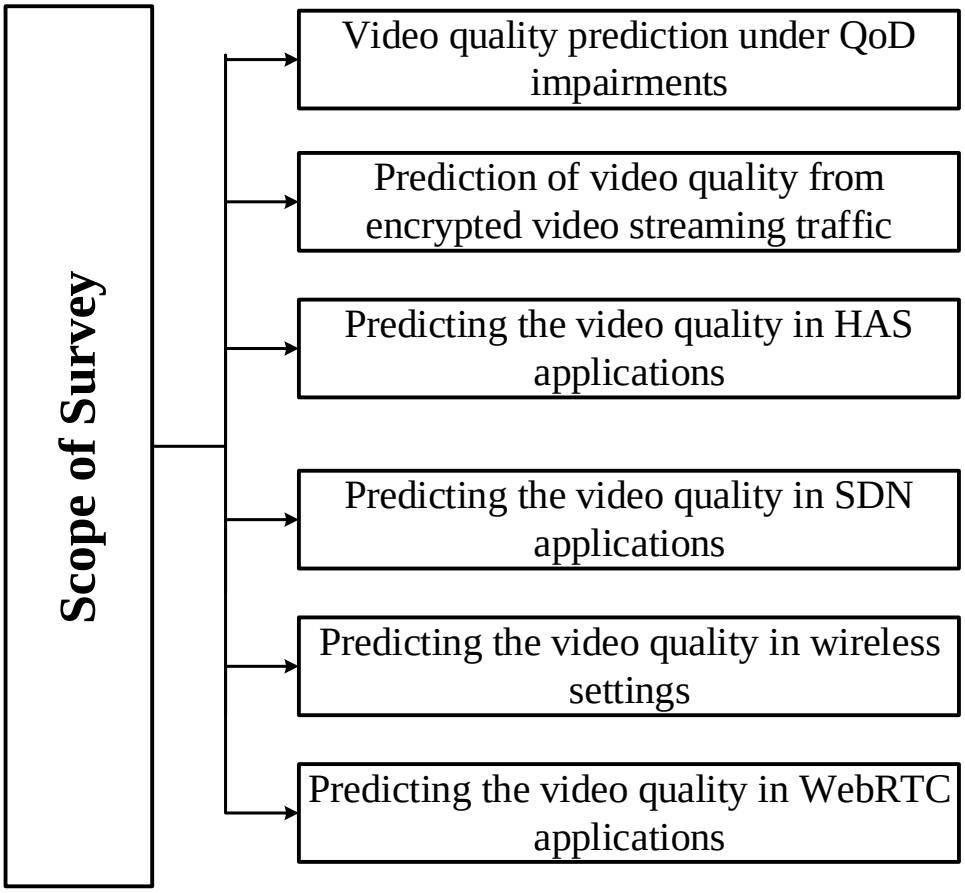

Figure 2. This survey reviews ML video quality prediction using QoD metrics under QoD impairments, encrypted video traffic, HAS video applications, SDN, wireless settings, and WebRTC applications.

2. Overview of ML Applications in Networks

In this section, we provide an overview on how ML is used in networks. We review previous surveys on the applications of ML and DL based on the areas of interest and learning paradigms discussed.

ML is a subset of AI that enables computers to learn and improve from past experiences. DL is a subset of ML that relies on deep neural networks (DNN) to train a model. In comparison to traditional ML, DL's main advantage is its automatic feature extraction, eliminating the need for time-consuming feature engineering [25]. This comes at the cost of requiring more computing power, and in most cases, an increase in computational complexity. ML models are easily adaptable to most problems and easy to understand.

The nature of the training data is one of the most important considerations in ML problems. The ML framework is trained to achieve a certain objective during the training process, such as making a decision, predicting a value, or performing a classification task. Without any human intervention, the ML framework learns the relationship between input and output data through training [26]. The online ML algorithm is another type of ML in which the model is updated for each new input feature after each prediction [27].

Figure 3 depicts the basic workflow of machine learning algorithms. To train the algorithms, the dataset is loaded into the ML platform. The ML platform then generates the model, by learning the correct parameters and features for the prediction task. This model is tested for accuracy (against ground truth). If the accuracy is not satisfactory, then more optimization may be needed. This process may include modifying or discarding variables, and adjusting hyperparameters [28] (model-specific settings) to achieve acceptable levels of accuracy [29]. The trained ML system is then evaluated on additional data to guarantee

that the model generalizes well to new data. There are generally three main categories of ML models: supervised learning, unsupervised learning, and reinforcement learning (RL).

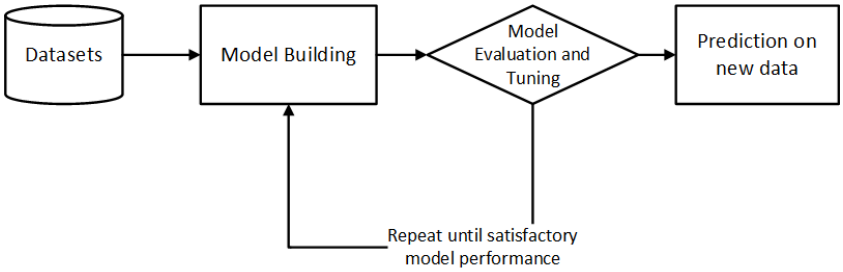

Figure 3. The basic workflow of ML systems is shown. The datasets are loaded into the ML algorithm, which learns the appropriate parameters for the task. The model evaluation and tuning process may involve modifying the learned model parameters and hyperparameter tweaking until an acceptable accuracy level is achieved. The model is then tested on new data to verify its accuracy.

Supervised learning relies on labeled datasets to create ML models. The goal in this technique is to predict the value of one or more output variable from a vector of input variables. The training dataset consists of samples of the input variables and the corresponding output values. In the learning method, e.g., regression, there is a function that provides a prediction of the value of the output variable in relation to a change of input variable. Linear regression (LR) [30], decision trees (DT) [31], random forests (RF) [32], and support vector machines (SVM) [33] are some examples of supervised learning algorithms. An important application of LR in the type of studies we survey is given in [34].

Unsupervised learning aims to explore the input data space to infer structure directly from unlabeled data. This sort of learning is essential in cases where the applications lack labeled data. While unsupervised learning can be used to solve a variety of problems, clustering algorithms [35] are the most popular. Examples of other unsupervised learning algorithms are k-means [36], self-organizing maps (SOM) [37], expectation-maximization (EM) [38], and generative adversarial networks (GAN) [39]. In this context, the method in [40] is a form of blind deconvolution approach, which can be classed as an unsupervised learning algorithm.

RL is used to handle applications in which the goal is to learn a policy, i.e., a mapping between environmental conditions and actions to be done, while interacting directly with the environment. RL is an agent-based iterative technique where the agent learns by interacting with the environment and exploiting the knowledge. Unlike supervised machine learning, RL does not learn from a specific dataset. Instead, the RL agent learns from the relevance of its actions and selects an action based on previous information and a new choice. RL is fundamentally a trial-and-error learning strategy [41]. The reward from a specific action is learned via RL, which provides a feedback loop to the algorithm. The agent then alters its behavior in response to the preceding reward. Until the reward saturates or reaches a pre-defined threshold, the agent continues to interact with the environment by learning both the action and the reward [42]. Based on the rewards obtained from the environment, the agent learns how good or bad its action was. Examples of RL include the Q-learning [43] and deep reinforcement learning (DRL) [44].

The applications of ML to networks have been reviewed in numerous studies. Imran et al. in [45] provided a review with a focus on recent research studies and future trends in IoT, SDN, and ML hybrid applications. A range of perspectives were covered in this study, including wide-area networks, edge networks, and access networks. The authors reviewed how ML applications in SDN-enabled IoT environments bring about intelligent network decisions in the areas of traffic classification, routing optimization, QoE/QoS prediction, and resource management. Related surveys such as that conducted by Imran et al., which focused on IoT, include [46–48]. Al-Garadi et al. in [46] surveyed ML and DL applications in developing security mechanisms for IoT frameworks. Mahdavinejad et al. in [47]

provided a survey on how various ML methods may provide solutions for the challenges presented by IoT data, focusing on smart cities as the use case. The survey in [48] listed traffic profiling, IoT device identification, security, edge computing infrastructure, and network management as the major applications of ML in IoT. Security is a common area covered in these studies. As IoT systems are complex and integrate multiple components, maintaining security within the IoT wide-scale attack surface is challenging. The works described in [45,48] considered the role of ML with SDNs in addressing the issue of security. The authors presented cases of ML integration with SDN for anomaly detection, traffic profiling, and classification.

Miller et al., in [49], presented a review of DNNs for defending networks against attacks. The review in [50] by Tang et al. surveyed the literature on applications of ML techniques for preventing phishing attacks in real-time. Meshram et al. in [51] provided a review of ML for anomaly detection in industrial networks. Hodo et al. in [52] conducted a survey which reviewed ML applications in network intrusion detection systems (NIDS) and their performances in detecting anomalies. The focus in this work was on the applications of DL in NIDS. The survey in [53] by Sultana et al. reviewed recent works on ML techniques that leverage SDN to implement NIDS. In their paper, the authors argued that the SDN framework can be used to detect vulnerabilities in networks and to monitor the overall network health. Buczak et al. in [54] presented a review of ML applications for cyber security intrusion detection with a focus on wired systems. Otoum et al. in [55] provided a study of DL-based intrusion detection for monitoring critical infrastructures through wireless sensor networks (WSN). The authors of [54,55] both examined DL-based intrusion detection systems (IDS) as an alternative to ML-based IDS. The DL techniques offered slightly better performances than the traditional ML approaches considered. The authors of [55] proposed ML-IDS for WSN-based infrastructure monitoring due to the shorter training time. In addition to the shorter training times of ML-IDS, the authors of [54] listed model interpretability, accuracy, and computational complexity as other factors to consider when choosing between ML-IDS and DL-IDS. These factors were common to both studies. Buczak et al. argued that the dataset needed to be more representative of the problem task. According to the authors, in order to detect anomalies and misuse, it is desirable for an IDS to reach network and kernel-level data. This granularity in data was not considered in the works described in the survey.

Sharma et al. in [56] reviewed the applications of ML in WSNs in the context of smart cities. The survey in [57] by Zhang et al. focused on mobile and wireless research based on DL. The review by Klaine et al. in [58] examined self-organizing networks (SON) from the point of view of learning and provided an overview of the most common ML techniques used in cellular networks. Musumeci et al. in [59] provided an overview of ML applications in optical networks. These studies focused on specific ML algorithms or applications.

A survey on unsupervised learning applications in networking was provided by the Usama et al. in [60], who pointed out the dominance of supervised learning in networking. The authors surveyed the literature on unsupervised learning applications in the areas of internet traffic classification, anomaly/intrusion detection, network operations, optimizations, and analytics. Other areas covered in the survey include QoS/QoE optimization and TCP optimization. Fadlullah et al. in [61] provided a survey of ML algorithms relevant to network traffic controls systems. They reviewed and compared the performances of ML and DL techniques proposed in the literature. The authors of Boutaba et al. in [62] conducted a review of the various network applications of ML, which included traffic predictions, traffic classifications, routing, congestion control, resource allocation, and Qos and QoE management. Although Boutaba et al. provided one of the most comprehensive surveys on the use of ML, their review focused on papers before 2018. We also note that the survey missed out on some novel ML applications in networking. For instance, the load-adjusted learning first proposed in [34], and used in [9,40,63], was not covered in the survey. Our background in algorithms as opposed to networks gives us a different perspective on the survey. A recent survey was published by Ridwan et al. in [64]. The

authors presented a survey of recent trends in the application of ML in networks. The authors considered the applications of ML in congestion control, predictive models, IDS, routing, QoS improvements, and resource management. We note some reoccurring themes in these surveys in the following areas: ML for resource management, congestion control, routing, traffic classification, and network operations. For instance, given the deteriorating effect network congestion has on network performance, numerous studies have considered ML methods to ensure network stability and efficient resource utilization. Figure 4 shows the summary of existing surveys and the branches of AI considered in the review. In Table 1, we provide a cluster of related surveys illustrating the different areas of focus and applications covered in these surveys.

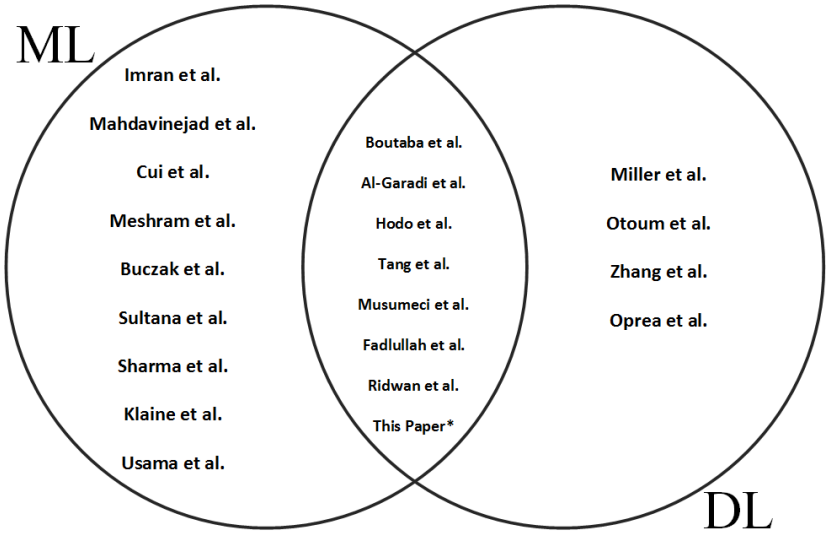

Figure 4. A summary of existing surveys and the branches of AI considered. Traditional ML algorithms dominate the literature, possibly owing to the lower computational requirements. A good number of studies combine DL and ML techniques.

Table 1. Areas of focus and applications considered in related surveys. The majority of the surveys we see are on ML applications in the network security domain. The growth of video traffic necessitates a review of existing works; this paper fills this gap.

Area of Focus and Application	Survey Paper
Network Security/IDS	Al-Garadi et al. [46], Cui et al. [48], Miller et al. [49], Tang and Mahmoud [50], Meshram and Haas [51], Hodo et al. [52], Sultana et al. [53], Buczak and Guven [54], Otoum et al. [55], Usama et al. [60]
IoT	Imran et al. [45], Al-Garadi et al. [46], Mahdavinejad et al. [47], Cui et al. [48]
Smart Cities	Mahdavinejad et al. [47], Sharma et al. [56]
WSN / Mobile / Cellular Networks	Otoum et al. [55], Sharma et al. [56], Zhang et al. [57], Klaine et al. [58]
Networking, Network Operations & Optical Networks	Usama et al. [60], Fadlullah et al. [61], R. Boutaba et al. [62], Ridwan et al. [64]
SDN	Imran et al. [45], Cui et al. [48], Sultana et al. [53], This Paper
Video Frame Prediction	Oprea et al. [65]
Video Prediction from QoD measurements	This Paper

Given the dominance of video traffic, a good number of research efforts from both academia and industry have investigated the applications of ML in the field of video streaming. Oprea et al. in [65] surveyed applications of DL techniques to intelligent video frame prediction from a sequence of context frames. Aroussi et al. in [66] reviewed the literature on ML-based mappings of video quality with QoS-derived metrics. That study

was similar to our paper. However, their review considered papers published before 2014. In this paper, we review the literature on studies that leverage the QoD measurements in the prediction of the video quality.

Previous studies on the prediction of the video quality show that QoD measurements can be used to predict the overall quality of the video stream [67]. Recent evidence suggests that for QoD optimization, ML techniques can be leveraged to support more advanced and flexible models than traditional algorithms to achieve higher performing network services [68] We give an example of how ML is used for QoD prediction in video networks. Figure 5 depicts a time-varying client population accessing video streams being streamed by a cloud-hosted video server. Using this setup, the authors of [9,63] demonstrated how audio buffer rate (ABR), video frame rate (VFR), and packet count, QoD measurements, could be predicted using system activity reports (SAR). The SAR function is used to extract device statistics from the server. These device statistics, feature set x, refer to the operating system's metrics (the server). Some features captured with SAR include: the number of TCP active connections and number of running processes. Video stream requests are served by a VLC player at the clients, which extract QoD measurements: packet counts, VFR, and ABR.

The authors posit that by modeling the effect of the time-varying load [34] from the client population, they can predict the QoD measurements, y. They called this the load-adjusted (LA) technique. Let θ_n represent a video resource currently being accessed by a client. Equation (1) defines the server response to one video request at a time i as the sum of the resources held by a user and some deviation signal specific to a feature, $\epsilon_i[n]$:

$$x_i[n] = \theta_n + \epsilon_i[n], \quad \text{where} \quad i \in \mathbb{Z}, x_i[n], \theta_n \in \mathbb{R}. \tag{1}$$

The load signal $K(i)$ is multiplied by a typical usage weighting, θ_i, and produces the level of usage of resources θ_n.

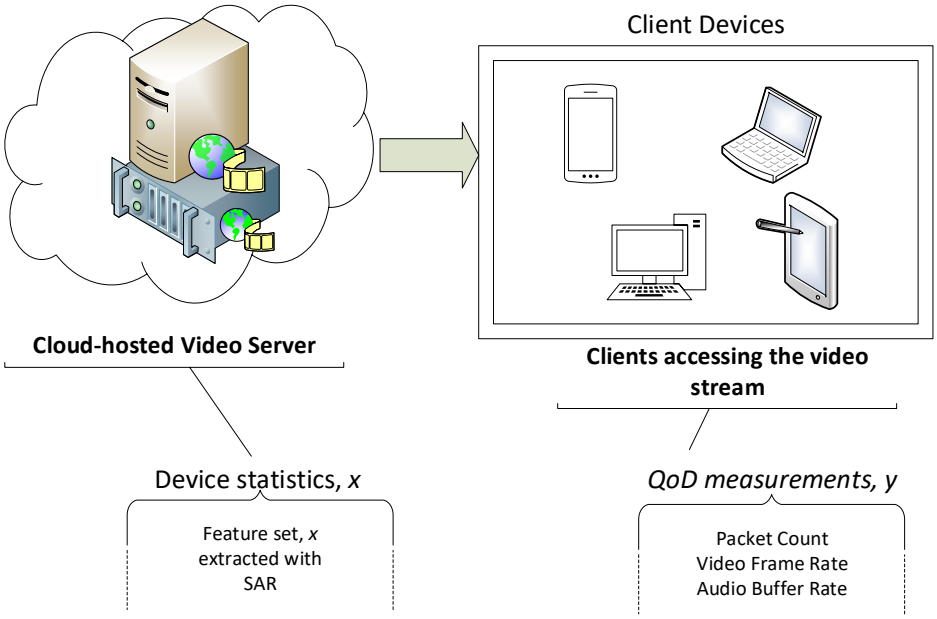

Figure 5. Client devices access video streams served by cloud-hosted servers. Device statistics, feature set x, are collated from the servers using SAR; ML models are used for the prediction of QoD measurements, y.

The objective is to predict the QoD metrics, packet count, VFR, and ABR at the client device with the device metrics. We use ML models to learn parameters from the input features in a way that our model estimates closely approximates the actual.

The authors of [34] first considered the LR model. The LR models the relationship between the target QoD metric y and the independent variable x as a linear function of the form:

$$\hat{y}_i = \sum_{n=1}^{N} x_i[n]\beta[n] \tag{2}$$

where the intercept is represented by $x_i[n]$, and the remainder of the features are the predictor features. The model coefficients are represented by $\beta[n]$ where $n = 1, \ldots N$.

Using the LA method, the LR models are load-adjusted by training weights for each value of the load signal.

$$\hat{y}_i \Big|_{K(i)=k} = \sum_{n=1}^{N} x_i[n]\beta[n] \Big|_{K(i)=k} \tag{3}$$

In a more general sense, $y = f(x)$, where $f()$ is a ML algorithm such as RF or DT.

Figure 6R1 illustrates the packet count and the system load, recorded during 8000 s. The load $K(i)$ on the server side plays a key role in the QoD of the streamed video at the client device. As the load on the system increases, the TCP active connections' signal increases and may decrease the number of packets delivered to the client because of the limited system resources. The authors employed a range of ML models for the prediction of the QoD measurements. The models used in the study were as follows: LR [34], ridge regression (RR) [69], least absolute and selection operator (LASSO) [70], Elastic Net (EN) [71], and RF. The EN and RF models achieved the best performances for all QoD targets. Figure 6, R2 compares the prediction accuracy of the EN model with the actual packet count statistics [9]. By utilizing a suitable adaptive regularization function, the load-adjusted learning algorithms outperformed the techniques that did not model the load effect by 10%, and reduced computation.

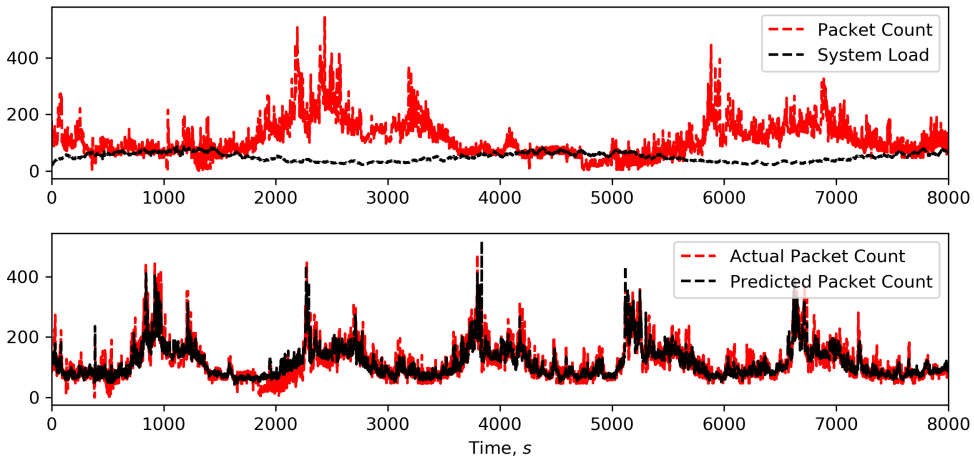

Figure 6. Row 1 (R1): The packet count, y, is illustrated for 8000 s along with the system load. The load signal is a feature of x. There is a periodic change in request patterns. The load signal increases as the packet count decreases, which illustrates the dependencies between the two statistics. Row 2 (R2): The actual packet count statistics are compared to the predicted packet count.

We give an example of where a feedback loop is required to improve a service and the overall health of the network. The authors of [72] proposed a codec-aware network adaptation agent (CNAA), a light-weight and responsive system for predicting jitter, a QoD metric, suitable for a video delivery system that uses adaptive video codecs. The authors,

by modeling the adaptive behavior of the codec in response to time-varying levels of network congestion, achieve accurate predictions of jitter. The demonstration in [73] shows how predictions realized from the CNAA learning agent can be integrated with an SDN controller to enable QoD improvements for an SDN-based video delivery system. Figure 7 illustrates the ML-SDN framework demonstrated, which consists of three components: (i) the Topology Discovery and Statistics Gathering Component: creates a topological view of the underlying infrastructure and collects information periodically about the network traffic flow; (ii) Learning Phase Component: the CNAA learning agent achieves accurate predictions of jitter by estimating the adaptive behavior of the code; (iii) Action Phase Component: the reconfiguration component which exploits the information learned from the learning phase to achieve improvements in the QoD of the video.

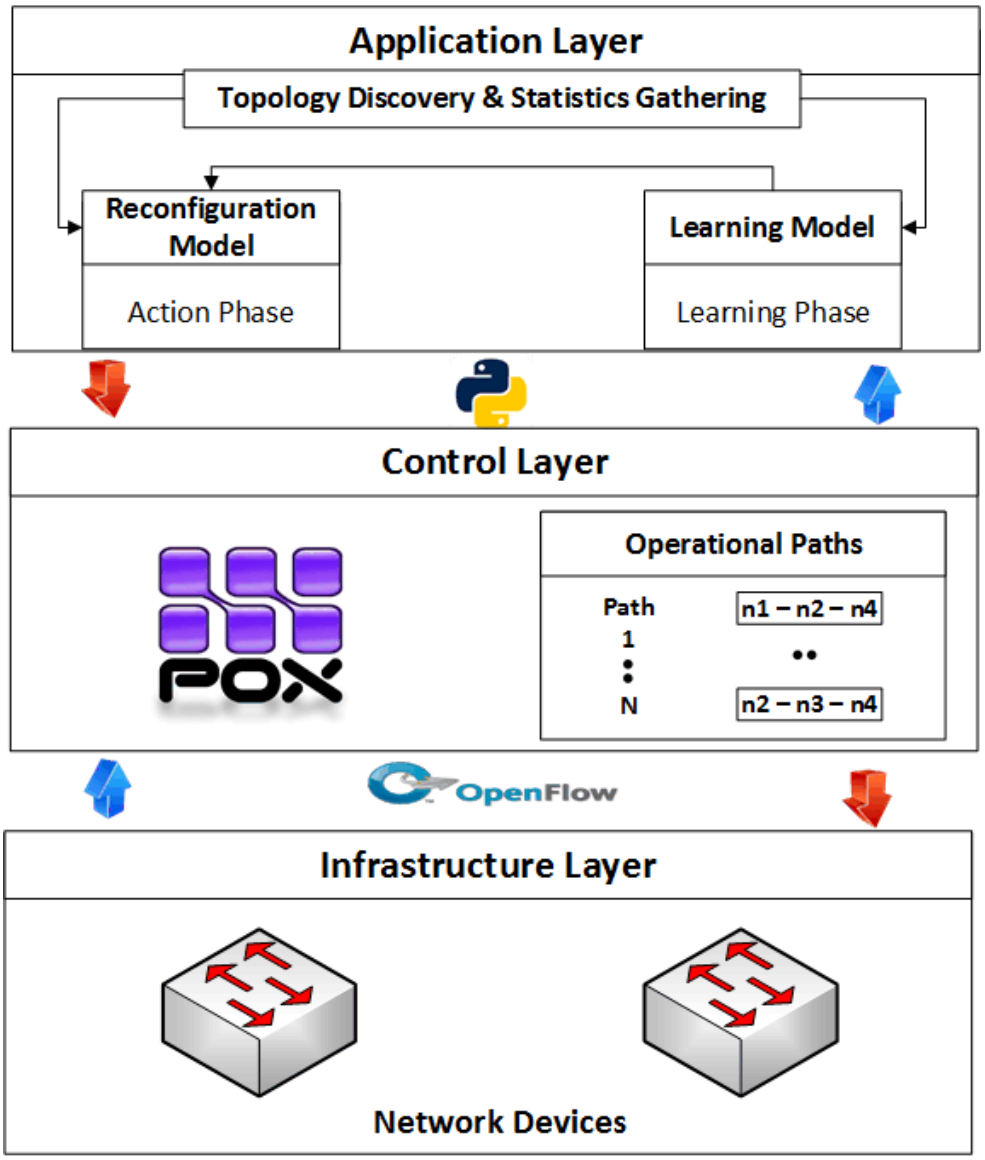

Figure 7. An SDN controller has the ability to adaptively reroute the operational paths for a video delivery system by integrating QoD measurements from an ML model. Integration of CNAA predictions with an SDN controller in a feedback loop is able to preserve the video quality by avoiding congested paths.

3. Background on Video Streaming

Video streaming over IP networks enables viewers of a video to view the video stream without having to fully download the whole video first. Playback at client or end devices

occurs as the media files are being downloaded. This in many ways differs from traditional data transfer over the Internet. The packet sizes in video streaming are much larger and generally require more bandwidth for transmission in comparison to data traffic. Another fundamental difference between video transmission compared with traditional data traffic is the fact that there are real-time constraints for video delivery.

Video streaming can be broadly divided into two flavors, video on demand (VoD) and live (real-time) video streaming. For VoD streaming, the content is stored on a server, and upon user request, the content is transmitted to the client. In live streaming, the server generates video content in real-time as it is being injected by the source. Relay of live events via traditional broadcast TV is an example of this type of application [74].

3.1. The Anatomy of a Video Stream

A video stream comprises of a sequence of multiple segments. Individual segments are composed of several group of pictures (GOP), with each GOP having a header at the beginning. The number of GOPs in that segment, and the types of GOPs, are listed in the segment header. A GOP is made up of a series of frames. An I (Intra) frame comes initially, followed by multiple P (predicted) and B (bi-directional predicted) frames. A GOP frame is broken into numerous slices, each of which contains several macroblocks. Video encoding and decoding activities are performed at the macroblock level. There are two types of GOP: closed GOP and open GOP. Regarding closed GOPs, they are independent and can be processed independently. On the contrary, each open GOP is dependent on another GOP; it is impossible to process the GOPs independently [75].

Video streams can be split at various levels to be processed, including segment, GOP, frame, slice, and macroblock levels. Several GOPs can be processed independently at the sequence level. The time needed for the transmission and processing of a sequence, on the other hand, is a bottleneck due to its large size. Processing at the frame, slice, and macroblock levels necessitates dealing with spatio-temporal relationships, which makes the process difficult and time-consuming [76].

3.2. Video Streaming over IP Networks

Typically, for a video streaming session, the video streams or frames have to be played at between 24 and 30 frames per second (fps) to realize the illusion of motion. Prior to the delivery of video streams over IP networks, video compression algorithms are deployed to achieve intra and inter-frame compression. These have temporal dependencies of producing I, B, and P frames. Of the three, I frames are largest due to the fact that they only utilize intra-frame compression, whereas B and P frames are smaller because they use previous I frames for further size reduction [77]. This gives rise to a situation whereby the encoded bits per frame is a variable quantity, which results in variable bit rate (VBR) video.

The VBR encoded video stream is then transmitted on the Internet. As the Internet does not provide a consistent guaranteed bandwidth for the video data, the network can only support the video streams on a best-effort basis. As a result, if the network bandwidth is not enough to accommodate the video bit rate, then the decoder at the client end starts consuming the video content at a greater rate than the rate at which the new data stream is being received from the network. Subsequently, at some point the decoder finally runs out of video data to decode. This situation may result in frozen screens (rebuffering events and video stalls). This is clearly a problem. This is one of the motivations for the ML approaches outlined in this paper. To address these issues without requiring some costly and complex guaranteed bandwidth mechanisms, the following solutions attempt to balance the video bit rate with the available network bandwidth [77]:

- Use of large playout buffer: The use of a large receive buffer can help overcome temporary variations in the network throughput. The video player can decode the pre-fetched data stored in the playout buffer.
- Transcoding-based solutions: These solutions modify one or more parameter of the raw video data algorithm to vary the resultant bit rate. Examples include varying the

compression ratio, video resolution, or frame rate. However, transcoding-based solutions require complex hardware support and are computationally intensive processes.

- Scalable encoding solutions: These solutions are achieved by processing the encoded video data rather than the actual raw data. Hence, the raw video content can be adapted by utilizing the scalability features of the encoder. Some examples of these solutions involve adaptation of the picture resolution or frame rate by exploiting the spatial and temporal scalability in the data. However, these solutions require specialized servers to implement this enhanced processing.
- Stream switching solutions: This technique is the simplest to implement and is also used for content delivery networks (CDN). This approach involves preprocessing the raw video data to produce multiple encoded streams, at varying bitrates, resulting in multiple versions of the same content. Thereafter, a client-side adaptive algorithm is used to select the most appropriate rate based for the network conditions during transmission. Stream switching algorithms do not need specialized servers and use the least processing power. However, these solutions require more storage and finer granularity of encoded bitrates.

Industry favors using a large playout buffer and stream switching as the preferred solution for video transmission owing to the ease of deployment. To avoid a situation that may bring about a buffer under-run, the video server has to decide on the appropriate sending rate [78].

3.3. Evolution of Video Streaming

Video streaming has attracted much attention and research for quite some time. Its popularity has grown tremendously, particularly with the deployment of popular video streaming services such as Netflix and YouTube. In this section, we attempt to go back in time and present an overview of the evolution of video streaming over the Internet. We adopt a chronological order as we overview three stages of video streaming via the Internet.

3.3.1. Client–Server Video Streaming

The introduction of video streaming in the 1990s started with the design and implementation of the client–server (C/S) streaming architecture. In C/S video streaming, the client device receives a video data stream pushed by a media server. With respect to the C/S architecture, the existing transport protocols, the TCP and User Datagram Protocol (UDP) were not entirely suitable for video relay over the best-effort Internet infrastructure. Owing to the real-time nature of most video streaming applications, the TCP's features of congestion control and reduction of data transfer windows sizes following packet losses were found to be detrimental to the application setup. The UDP, on the other hand, is a simple connectionless protocol. UDP offers no handshaking dialog with no retransmission, and no guarantee of delivery or ordering of packets [79]. To enable UDP real-time video streaming, the Real-time Transport Protocol (RTP) [80], Real Time Streaming Protocol (RTSP) [81], Session Description Protocol (SDP) [82], and Real-Time Control Protocol (RTCP) [83] were proposed to detect packet loss, compensate for jitter, and control the video streaming clients.

The RTP/RTCP/RTSP protocol suite was standardized by the Internet Engineering Task Force (IETF) [81] specifically for Internet video streaming. However, the characteristics of these protocols result in complex and expensive servers. When Network Address Translation (NAT) devices are present, they may not provide access to control traffic or media traffic [84]. Interoperability among media servers from different vendors was hard to achieve, despite implementing the same baseline protocols. This was largely due to some optional features or differences in design. Failovers often cause presentation snags due to a server fault and are rarely seamless unless certain redundancy schemes are in place. These scalability, vendor lock-in, and vendor dependency issues, along with the high maintenance costs, all present deployment challenges for protocols such as RTSP [15].

3.3.2. Peer-to-Peer (P2P) Streaming

P2P networks [85] allow users to share content without the need for centralized servers, making them an appealing option for video delivery over the Internet. In P2P video streaming [86], peers consume resources and supply resources to other peers. The peers contribute their uplink bandwidths and do not depend on the underlying network infrastructure for any support [87]. There are two types of P2P networks: tree-based [88,89] and mesh-based [90,91]. In tree-based video streaming framework, peers are organized in a tree structure such that the video content is pushed from the root to subsequent levels of the tree until it gets to the leaves. This enables lower latency in data dissemination. Such setups are easy and simple to control but can be negatively impacted by peer churn [92]. In mesh-based video streaming systems, each participating peer can connect to a random set of neighbors. Each peer can then download and upload data to different peers at the same time. This is possible as there are no dependencies in this setup, whereas there are in the tree-based overlay. Data distribution here is done in an unstructured manner, and this setup is more suited for applications that can tolerate start-up delays. Since each peer maintains a set of neighbors at any given point in time, this overlay is much less susceptible to peer churn than the tree overlay. P2P offers some level of scalability, but there exist some drawbacks for users. There may be requirements to download and install specific software. In most cases, users have to keep some ports open to allow access beyond firewalls and NAT which could pose a security risk [93].

3.3.3. Hyper Text Transfer Protocol (HTTP) Video Streaming

The HTTP protocol is used to deliver video content in today's video streaming services. Using HTTP for video streaming is simple in terms of setup because most firewalls support HTTP/HTTPS traffic, which eliminates the need for additional network configurations to handle video traffic [94]. HTTP Adaptive Streaming (HAS) [95] enables content providers to meet the needs of a wide range of devices and scenarios. The use of HTTP over TCP adds to the benefits of using this technology. HAS can be developed on top of existing content delivery systems for everyday Web use [96]. HAS divides a video file into a number of chunks. Each chunk is encoded at different video rates and stored with a file called Media Presentation Description (MPD) as a description [97]. Unlike the traditional C/S architecture, which was based on a media server push-based video delivery method, with HAS, the clients pull media streams from the server. HAS treats media files just like regular Web content and delivers them in chunks over the HTTP. A streaming client continuously assesses and evaluates its own capabilities. The chunk with the greatest video rate that is sustainable given the estimated capacity is then requested. Adaptive bitrate selection (ABR) [94] is the mechanism by which a client determines the profile and schedule of a chunk to download.

The studies in [15,98] provided extensive classifications of ABR algorithms. Most of these previous studies agreed on dividing ABR algorithms into three categories based on their needed inputs: buffer-based (BBA [99], BOLA [100]), throughput-based (PANDA and CONVENTIONAL [101], FESTIVE [102]), and hybrid-buffer-throughput-based. ML and control techniques have shown much promise in ABR, leading to the design of ML-based [103] and control-based class [104] ABR algorithms.

Most HTTP-based video streaming solutions are based on HAS. Commercial solutions exist, such as Microsoft's Smooth Streaming [105], Adobe's HTTP Dynamic Streaming (HDS) [106], and Apple's HTTP Live Streaming (HLS) [107]. In a move toward ensuring an open standard for video streaming, the Moving Picture Experts Group (MPEG) in conjunction with 3rd Generation Partnership Project (3GPP), released MPEG Dynamic Adaptive Streaming over HTTP (MPEG-DASH) [97].

3.4. Common Challenges in Video Streaming

Even though switching from a server-push to a client-pull model has its advantages, HAS faces a few challenges, nonetheless. Some issues encountered in HAS systems include:

- HAS Multiplayer Competition and Stability: It is important for HAS clients not to switch bitrate frequently, as it leads to video stalls, which can negatively affect the video quality. In a multiplayer HAS environment, it is crucial to achieve fairness. Clients competing for available bandwidth should equally share network resources based on their viewers, content, and device characteristics.
- Consistent Streaming Quality: The correlation between video bitrate and its perceptual quality has been shown to be non-linear by studies conducted on video quality analysis [108]. In general, it is preferable to stream videos at a consistent quality rather than at a consistent bitrate, which results in fewer oscillations in perceptual quality [109].
- Frequent Switches: Depending on the network condition and/or buffer status, the rate adaption algorithm switches video quality. While quality switching is a useful feature of HAS that helps to reduce the frequency of stalling occurrences, frequent quality switching may cause user frustration.
- Throughput: TCP throughput of nearly twice the video bitrate is necessary for effective streaming performance in general, which highlights a fundamental shortcoming of HAS applications [110].

HAS relies on ABR algorithms to dynamically select appropriate video bitrates in response to changing network conditions. To achieve this, it employs two control loops: (1) a TCP congestion control loop that reacts to network congestion by attempting to match the sending rate to the network's capacity; and (2) the ABR selection loop, which adjusts the video bitrate to match the average TCP throughput [111]. In general, the goal of the ABR algorithm is to avoid playback interruptions arising as a result of buffer depletion. The importance of a de-jitter buffer was investigated in [112] to overcome this challenge. This study provided some evidence showing the effect of video codec type and bitrate on video quality. The algorithm also attempts to maximize the quality of the video stream whilst minimizing the number of video quality fluctuations. It is imperative that the algorithm minimizes the startup delay time of the video stream. The main challenge for the ABR designer and the research community is to strike a balance between these objectives to provide excellent video quality to end users. Although the HAS media display module has been standardized, the ABR adaptation logic is left to the developer's discretion.

The study in [113] lists the initial playback delay, the played out quality, the playback quality changes, and stalling as the key quality degradation factors that affect the video streaming session and impact the video quality. We introduce here the descriptions of these QoD-derived KPIs and some other quality influencing factors.

- Media Session: Media session refers to the start of video playback until the end. It includes the effects of initial loading times, rebuffering events, and switching quality, if applicable. This means that any of these events will cause the media session to be longer than the video/audiovisual playback time.
- Initial Delay: Initial delay is the duration between the video request by the client and the actual time when video playback commences. It is also referred to as initial buffering.
- Playback Quality Changes: It refers to the change in quality throughout the course of the video playback. It is also known as rate or quality adaptation.
- Quality Switching Frequency: The rate at which the quality changes during media playback is referred to as quality switching frequency.
- Stalling: This occurs when video playback is interrupted. In cases where the network throughput is insufficient for the content to be downloaded faster than it is consumed, the buffer depletes, and playback is forced to pause until more data are downloaded and the buffer is refilled.
- Rebuffering: This refers to cases when the data in the buffer are depleted, thereby leading to a video playback stalling. These events in a streaming session are normally represented by a spinning wheel, loading sign, or sometimes a frozen frame.
- Rebuffering Frequency or Ratio: The amount of rebuffering incidents per unit of time is referred to as the rebuffering frequency.

- Rebuffering duration: This is the total duration of all rebuffering incidents in a single media session.

4. Review of Video QoD Prediction via ML

Quantifying the effect of network performance on the video quality is critical, as this determines the success, degradation, or failure of a service. The following subsections discuss the applications of ML techniques for the prediction of the video quality from QoD measurements.

4.1. Video Quality Prediction under QoD Impairments

Vega et al. in [114] investigated the prediction of video quality under QoD impairments. In their work on real-time quality assessment of video streaming applications, the authors proposed an unsupervised DL model based on restricted Boltzmann machines (RBMs) [115]. Using only a subset of features extracted by a client involved in a streaming session, the proposed models were used to infer the no-reference features [116] of the received video stream. The features utilized in the study were mainly from the video-related features (such as the bit stream, frame, inter-frame, and content). In the study, the authors considered the effects of network QoD metrics on the streaming session. Two synthetic datasets (one for generic network conditions and another for extremely lossy networks) were used in the study. Although the authors presented results demonstrating the feasibility of their approach, there were a few issues in this approach. First, the QoD impairments studied (network delay, jitter, throughput) were treated as independent variables. This assumption is not a realistic one, given the way that events in networks cause these variables to have dependencies. If a client device involved in a streaming session has to assess the video stream quality, it will require information on what impairments may exist already in order to select the appropriate predictor. Secondly, given that this technique is based on a multi-RBM solution, the computational costs of this method would need to be evaluated. This would help ascertain the practicality of using this method in real-time as proposed.

Accurate throughput prediction and bandwidth prediction have the potential to boost the performances of a wide range of applications. Adaptive multimedia streaming services, for example, can enhance a video stream session by making timely quality modification decisions based on these information. In [117], Raca et al. described how ML and DL techniques can be leveraged for accurate throughput prediction in video streaming applications running over cellular networks. The authors trained a throughput predictor using RF and SVM models. In addition, they compared the performance, training, and data requirements of these ML models with a DL model, the long short-term memory (LSTM) model. The predictor was trained to estimate the average throughput over a future time range by processing the historical data over a set period of time. In their setup, the collector module collated device and network specific information which was used during the predictor training stage for future throughput estimation. QoD data used included the average throughput of devices connected to a given cell and the load, i.e., the number of devices in the same cell. To evaluate the performance of the throughput predictor in a video streaming application, the authors considered a HAS video client connected to a server using a controlled link whose bandwidth was driven by a 4G trace. They computed the following metrics: bitrate, number of switches, mean count of bitrate switches, number of stalls, and the mean bitrate of selected video chunks. Their technique was compared with a base case scenario; the streaming performance parameters were computed without using the prediction module. In comparison to the traditional player's reactive actions (i.e., the base case), the throughput predictor-assisted player made excellent decisions that were future-aware. As a result, the throughput predictor-assisted player eliminated stall situations by switching to a lower bit rate as soon as possible, but the traditional player failed to forecast abrupt dips in available bandwidth and chose higher bit rates. The

authors also reported that the throughput predictor module reduced quality switching. Results showed improved performance when network-level data were used.

In a study related to that of Raca et al., the authors of [118] proposed ABR for chunked transfer encoding (ACTE). They considered the problem of ensuring low-latency in live streaming applications. The proposed method relied on predicting the bandwidth of the applications and using the information in the selection of the appropriate bitrate setting for the client. In their investigation, they assumed that the client's choices would differ under different bandwidth conditions or impairments. The proposed design employed HTTP chunked transfer encoding instead of short video segments. This approach ensured that long segments can be created and sent in small bits, which are referred to as chunks [119]. The authors reported that ACTE realized accurate bandwidth measurement and prediction in low-latency streaming applications. To make ABR judgments, ACTE relies on three primary components: (1) a chunk-based sliding window moving average bandwidth measurement—this is similar to the baseline prediction approach used in [72]; (2) an adaptive recursive least squares (RLS)-based [120] bandwidth prediction module; and (3) a throughput-based bitrate selection logic. To evaluate the performance of ACTE, the authors utilized bandwidth measurements and packet-level data collated from the Twitch API. The authors measured the performance of their design by computing the average buffer occupancy, average number of switches, the average number of stalls and their average duration, and the average startup delay. When compared to ABR schemes that used segment-based bandwidth measurements, the reported results showed that ACTE achieved higher performance by reducing the number of stalls by 65% and their durations by 83%, maintaining a low latency of 2.3 to 3 s (36% reduction), providing a 28% higher bitrate. These results in [117,118] are quite promising, but given that the techniques' successes relied heavily on accurate and timely throughput prediction and bandwidth prediction, the ability to identify abrupt changes in throughput or bandwidth measurements is concerning. According to Cisco [121], the recommended network latency for video should be $\leq$150–300 ms. The latencies recorded here are much higher. The performances of these techniques across different channels and use cases warrants further investigation.

In [122], Mao et al. proposed Pensieve, a DRL model that makes future bitrate selections based on prior observations collected by the client video players. Pensieve is decribed as a neural network (NN) engine that learns a control policy for bitrate adaption entirely through experience without relying on any preset fixed rules. The model learns how to improve on past ABR decisions through reinforcements, in the form of reward signals that are measures of the video quality for previous decisions. To train the NN, the learning agent is fed the state inputs of the following: client playback buffer occupancy, previous bitrate decisions and network throughput measurements, and download time of the video chucks or the throughput interval times. These measurements are fed into the NN which then outputs an action, i.e., the bitrate selection for the next chunk. The authors reported that the ABR RL agent was able to identify policies that outperformed algorithms baseline methods that relied on fixed heuristics or employed erroneous system models. According to the authors, their method recorded this performance gain by optimizing its control policy based on the actual performance of previous selections.

Some related studies that investigated the use of RL as a data-driven technique to automatically optimize ABR algorithms include [123,124]. The authors of [124] adopted a similar approach as that described in the seminal work reported in [122]. According to the authors, their approach was able to account for latency issues, which were some limitations they observed in the Pensieve RL agent. They also claimed to have improved the performance of the streaming session by employing an ensemble model of two NN models to improve ABR decisions and enforce latency control. QoD metrics used in the evaluation included the average throughput for about 50 minutes, at a 0.5 second granularity. Similarly, the authors of [123] proposed ABRL, an RL-based ABR module in Facebook's production Web-based video platform. A limitation of their approach is that the authors reported experimental results based on Web-based videos, well-connected traffic patterns. This

evaluation may need to consider more generic conditions, network variability issues, and cellular networks to evaluate the RL agent's performance in those real network conditions.

In a related effort, Hiba et al. in [103] demonstrated how ML approaches, namely, supervised classification, can predict the ABR algorithm's class by estimating the most important features (feature importance based on mean decrease in impurity). The authors posit that their technique can forecast any ABR algorithm's bitrate decisions, providing a set of input features that may be seen at the application level. The authors formulated the ABR bitrate prediction problem as a multiclass classification problem. In keeping with their goal of designing an ABR agnostic technique for bitrate prediction, the authors used a set of generic features which did not require knowledge of the ABR design logic beforehand. Some QoD input features used include the buffer level, bandwidth, previous bandwidth level during the last download, download time, and previous bitrate. They tested their approach on some well-known ABR algorithms and reported experimental results on commercial closed-source players, using realistic VoD and live datasets. A range of ML algorithms were evaluated—namely: logistic regression [125], SVM, random forest (RF), decision tree (DT), adaptive boosting (AdaBoost) [126], gradient boosting (GB) [127], naive Bayes (NB) [128], and k-nearest neighbors (kNN) [129]. The results reported showed that the RF and GB models attained the highest prediction accuracies.

In [130], Yusuf et al. proposed SMASH: a supervised machine learning method for adaptive streaming over HTTP. This work, like Hiba et al's work, was an attempt to develop a more generalized method to ABR selection. The authors presented a scenario where the streaming client would adapt to a range ABR algorithms in response to changes in network conditions irrespective of the transmission context. The authors generated their dataset by streaming a mix of popular ABR algorithms across a range of different network settings (3G, 4G, and WiFi). During the streaming session, they logged in 22 features which included the buffer level, codec type, arrival and download time of video segments, segment duration, VFR, and average bitrates of the previous two or five chunks. Features used for model training were selected for the evaluation by considering the correlation matrix and some statistical information of the features set. Several ML models were used to evaluate the performance of SMASH. The models used include LR, quadratic discriminant analysis (QDA) [131], kNN, DTs, NB, AdaBoost classifier, RFs, and multi-layered perceptron (MLP) [132]. The model with the best performance in terms of classification accuracy, RF, was then selected for comparison with some state-of-the-art techniques. The authors reported some promising results, with SMASH outperforming all other approaches in all network conditions evaluated.

Most of the studies surveyed in this section adopted the RF algorithm for their experimental evaluations. The authors based this decision on the model's ability to reduce overfitting (via variance reduction) by averaging over several trees compared to other ML models used. In some cases, we found that the GB model improved the RF model's accuracy. The GB algorithm improves the RF algorithm by modeling the performance of the intermediate trees. GB's performance is boosted by sequentially and repeatedly fitting trees to residuals of the model, correcting errors resulting from previously trained trees with each successive new tree. We found that RL models are the current best ML candidates for ABR algorithm optimization. The RL seems well suited to the settings for ABR corrective actions owing to the ability for the RL agent to adaptively refine their behavior in response to the environment. For most of these studies, the RL learner interacted with the learning framework in the presence of degrading QoD, such as throughput fluctuations. This seems to be an obvious choice owing to the need to take corrective or remedial actions at the client device when the network conditions change. Table 2 provides a summary of the key studies surveyed in this section.

Table 2. Summary of video quality prediction under QoD impairments.

Author, Year	Objective, ML Techniques	Features	Key Results
Vega et al. [114]	• Real-time video quality assessment • Multi-RBMs	• Delay, jitter, throughput	• >85% accuracy
Raca et al. [117]	• Throughput Prediction in Video Streaming Services • RF, SVM and LSTM	• Throughput, number of devices in same cell	• Througput Predictor module achieved a reduction in quality switches and eliminated stalls. • Improved performance was achieved with network-level data
Bentaleb et al. [118]	• Low-latency in live streaming applications (ACTE) • RLS	• Bandwidth and packet-level data	• Achieved higher bitrates and reduced stall events and latency
Mao et al. [122]	• Pensieve, bitrate adaptation agent • RL	• Buffer occupancy, bitrate decisions, throughput, chunk download times, throughput interval times	• Pensieve recorded ≈12–25% improved performance gains over the best state-of-the-art schemes
Zhao et al. [124]	• Improved ABR decision making and latency control • Ensemble of two NNs	• Throughput measurements	• Improved video quality by controlling network latency limits
Yousef et al. [103]	• Prediction of ABR Algorithm's class • Logistic Regression, SVM, RF, DT, AdaBoost, GB, NB, kNN	• Buffer level, bandwidth, download times, bitrates	• ABR agnostic model for bitrate prediction • Best performance by GB and RF
Sani et al. [130]	• SMASH, a generic ABR selector • LR, QDA, kNN, DTs, NB, AdaBoost Classifier, RFs, MLP	• Codec type, video segment arrival and download times, segment duration, VFR, bitrate	• Best performing model, RF was chosen for comparison with baseline methods • The SMASH algorithm is able to make the best decision from a range of ABR algorithms

4.2. Prediction of Video Quality from Encrypted Video Streaming Traffic

The quality of the video stream playback at client devices can be correlated with the application QoD features, such as initial loading time (also known as startup delay or join time in the literature), frequency of rebuffering/stalling events, and playout quality (spatial resolution) [133]. However, since most video data are encrypted, network operators rarely have such information on the video traffic generated in their networks. As a result, one way to assess video streaming performance is to use network-level QoD features extracted from encrypted video traffic traces or independent network measurement tools run outside the video application data plane [134]. This is due to the fact that deep packet inspection (DPI)-based technology for analyzing the video data is no longer a viable option [135]. Using this approach contradicts the end-to-end security policies of SSL, and therefore causes an unfortunate set of issues, as outlined in [136,137]. Internet service providers (ISPs) must infer the video quality from network level measurements. This motivates the application of ML techniques to link the network-level QoD measurements to the video quality. In this section, we review the use of ML techniques in predicting video quality with QoD measurements obtained from encrypted streaming video traffic. We first consider

studies that focused on real-time prediction of the video quality. Thereafter, we present studies predicted the video quality by utilizing the video session data.

4.2.1. Real-Time Video Quality Prediction:

The authors of [138] proposed ML-based frameworks for real-time prediction of startup delay and resolution for encrypted streaming video services. The authors focused on the YouTube service. In their work on video quality classification, the authors defined three classes: *high, medium, low.* To generate data, the authors ran 39 experiments with varying levels of bandwidth. Video metadata, such as video duration, view, like, and dislike count, were included in the dataset. Network-level QoD features captured in the dataset include throughput, packet count, inter-arrivals, and byte counts. They started with a dataset of 54 features, which they divided into six categories: packet length statistics, size of transferred data in 5-second intervals, packet count statistics, inter-arrival time statistics, throughput statistics, and TCP flags count. The authors reduced the feature set number to 33, mostly due to redundancy. They defined two functions which analyzed an instance of a video streaming session, quantifying measured degradation and classifying the session as high or low.

If it fell into neither of these categories, it was assigned to the medium category. The authors evaluated different ML models in the classification task; models used included one rule (OneR) [139], sequential minimal optimization (SMO) [140], J48 decision tree [141], RF, and NB. The RF and NB models were reported as having recorded the highest accuracies on the full dataset and the reduced dataset.

Dimopoulos et al. adopted a similar approach in [142], relying on real cellular network measurements and ML models to predict typical system performance metrics for streaming services (e.g., playing resolutions and stalling occurrences) using round trip times (RTT), packet loss, and chunk sizes. The authors used a Web proxy in the network to collate data and infer the video quality of the data stream of YouTube sessions. Similarly to the study in [138], the authors solved a classification problem in which they leveraged the RF model in predicting the video quality. They reported high classification accuracies of ≈94%. One of the paper's primary results was that changes in video segment size and inter-arrival periods were among the most relevant markers of quality degradation. This is a reasonable assumption because as bitrate varies, so does the size and resolution.

YouTube video quality-relevant KPIs such as stalls, quality changes, and delays can be challenging to measure passively, especially on smartphones and cellular networks [143]. This is because of the difficulty in accessing application-level metrics directly on the YouTube application if the monitoring is done at the device level without root privileges. Considering this problem and the prevalence of end-to-end encryption in video streaming services, the study in [144] used ML and network-layer information to evaluate video-quality parameters (initial delay, stalling ratio or rebufferring rate, number of stalls, total stalling time) and user engagement for YouTube videos watched on smartphones. The authors extracted these network-layer features through the Android API. Simple metrics easily accessible through the Android APIs, such as the numbers of incoming and outgoing bytes, the signal strength, and the number of network switches, were some features extracted from end user Android handsets. The authors collated a dataset of 275 features in total. In general, these features included information about the received signal strength, the number of handovers, the number of network switches, and several statistics about incoming and outgoing traffic, aggregated over time windows of 1, 5, 10, 30, and 60 s intervals. They employed 10-fold cross-validation to evaluate a 10-tree RF model for each statistic [145]. To balance classes for learning purposes, the authors used simple bootstrapping approaches [146]. ML models were used to design predictors, which considered each problem as a classification task with discretized goals. They reported perfect performance in detecting the stalling or rebuffering ratio. Results reported showed that detecting the frequency of stalling events was the most challenging. However, the performance of the

RF model was not compared to that of any other ML model. For this type of task, it would be beneficial to evaluate a range of ML models.

Wassermann et al. in [147] presented a ML-driven architecture used to predict YouTube video resolution in real-time using network packet-level data only. The features used in the study included the numbers uplink and downlink packets, and transmitted bytes. Others included time-based data such as the time from the start of the slot to the first packet, and the time between the first and last packets of the time slot. To achieve this prediction, the framework analyzed ongoing streaming sessions using 1-second time frames. The system employed a stream-based approach to calculate several lightweight, snapshot-like statistical characteristics from video traffic. It considered three windows (current, trend, and session). The system computed the trend features using a first sliding window which aggregated the latest three time slots. The session-progression features were computed using a sliding window averaging all past slots since the start of the session. At the end of each 1-second time slot, the properties of that slot, as well as those of the adjacent windows, were fed into ML models that predicted the video resolution (144p, 240p, 480p, 720p, or 1080p).

The authors employed a diverse dataset of over 15,000 different YouTube videos recorded under varying network settings. The videos used in the work were streamed over an LTE mobile network, or a home or corporate WiFi network. The authors employed a 5-fold cross-validation to test nine ML algorithms: (1) DT, (2) RF with 10 trees (RF10), (3) AdaBoost using 50 trees, (4) an ensemble with 10 extremely randomized trees (ERT10) [148], (5) bagging with 10 trees (BAGGING) [149], (6) NB, (7) kNN with $k = 5$, (8) a feedforward neural network (FFNs) with three hidden layers, and (9) an SVM. The results showed that all models correctly detected the 480p-class, with SVM being the least accurate with an accuracy of less than 70%. For the video resolution task, DT, RF10, ERT10, and BAGGING earned near-perfect scores. The authors claimed that this was due to the fact that about 50% of the time slots belonged to this video resolution. In terms of computational complexity and achieving real-time prediction of the video resolution, the authors reported the RF10 as the best ML model. This result is promising, because a model for real-time prediction tasks should be highly efficient in terms of computational time. However, due to the obvious bias towards the majority class (i.e., the 480p-class), ML algorithms are likely to produce good accuracy scores on these datasets. As such, accuracy is not a very clear measure of performance for algorithms that work on imbalanced data, since the classifier will record poor results over the minority class [150].

In a related effort, Wassermann et al. in [151], considered the prediction of the average bitrate and the video resolution in real-time. Using the same dataset in [147], the authors evaluated the performances of two ML models, namely, RF10 and kNN with $k = 1$ and $k = 3$. The video resolution prediction task was posed as a classification task, and the bitrate prediction was considered as a regression problem. The results reported showed that the RF10 model outperformed the kNN model with an accuracy of 71%, against 66% accuracy recorded by the kNN with $k = 3$. For the bitrate prediction, the RF10 model outperformed the kNN. It was also reported that the RF10 model actually overestimated the actual bitrate for a large proportion of the time slots, ≈59%. The authors hypothesized that this may be advantageous for ISPs regarding applying the appropriate QoS policies that can help ensure continuous playback of the video stream.

Gutterman et al. [152,153] proposed Requet—a framework for real-time quality of experience metric detection for encrypted traffic. The authors focused on three metrics: buffer warning (low buffer, high buffer), video state (buffer increase, buffer decay, steady, stall), and video quality. The video state metric can be determined when the video level of the user is in a steady state. The metric also captures occurrences of buffer depletion and stall conditions. The system comprises a chunk detection algorithm, chunk feature extraction, and ML video quality prediction models. The chunk detection algorithm identifies video and audio chunks in encrypted traffic through IP headers. The audio or video chunk metrics recorded were: the protocol used to send the *get* request, start time,

time to first byte (TTFB), download time, slack time, chunk duration, and chunk size. The features extracted from the chunks were then fed into ML algorithms for prediction. The authors gathered two datasets: (i) using a laptop running YouTube from a browser over WiFi, Browser-WiFi; (ii) using the YouTube application on an Android smartphone over an LTE cellular network, App-LTE. In this work, these video quality metrics were predicted with a RF model. The results reported listed Requet's accuracies for predicting buffer warning, video state, and video resolution in the Browser-WiFi setting as 92.0%, 84.2%, and 66.9%, respectively. In the App-LTE setting, the accuracies for predicting buffer warning, video state, and video resolution were listed as 97.8%, 88.2%, and 80.6%, respectively. While these results are promising, given they considered real-time constraints of video services, there are a few limitations: (i) the evaluations considered only one service, the YouTube service; (ii) the technique was not tested on other services. The differences in ABR algorithms may introduce some dynamics that were not captured in the current design.

Seufert et al. [154,155] proposed ViCrypt, a system able to realize video quality predictions in real-time. The system was described as a ML-based approach for monitoring YouTube's video quality-relevant metrics in real-time. It was reported to be capable of predicting rebuffering events from such basic features. The system implemented with YouTube achieved these predictions by relying on constant memory stream-like inputs extracted from the encrypted stream of packets for an ongoing YouTube session. Firstly, ViCrypt utilized a sliding window which consisted of the last T time slots to compute trend features. A second sliding window consisted of all past time slots since the start of the streaming session to compute session-progression features. The features from within each time slot, along with the features from the corresponding windows, were fed into a RF model, which determined whether the current time slot of 1 second contained stalling or not. The approach is similar to that described in [147]. Input features utilized in the study included: the numbers of total uplink and downlink packets, count of bytes transferred, volume of TCP and UDP packets, TCP ratio, and UDP ratio, computed from the counts of the packets and byte counts. Other QoD measurements utilized in the study included, the average throughput of the slot (i.e., traffic volume divided by slot length) and the burst throughput (i.e., traffic volume divided by burst duration). Initial results reported in [154] showed that the technique realized good predictions using all extracted features. In [155], the authors evaluated the relevance of different feature sets. By examining independent time slot stalling, they investigated which features were specifically relevant for accurate prediction of stalling. They investigated the effects of adding prior predictions (i.e., recurrent feature set) from past time slots on the prediction accuracy. Results showed that the full dataset was not required for real-time prediction of rebuffering events, as they realized similar performance using a reduced dataset. Their findings also indicated that by using the recurrent features, the prediction error was reduced. They did not investigate this further. However, it may desirable to do so and also consider a range of other streaming services.

The work in [156] presented BUFFEST. The authors considered the problem of classifying video streaming flows depending on the present buffer conditions of the clients. The authors asserted that the ability to estimate buffer conditions accurately is vital to understand how video streaming flows are experienced by users. Specifically, BUFFEST's focus was on detecting rebuffering events in YouTube. The authors emulated a player that resides on a client's Network Interface Card (NIC) (or wherever the proxies was located) and registered HTTP-level, TCP/IP-level, and stream metadata. Encoding rates, chunk boundaries, and other information generally found in metadata files were included in this data. The authors focused on features based on simple one-pass metrics generated using only packet-level information to achieve fast processing. The performances of ML classifiers were compared with threshold-based classifiers. Two ML models were evaluated for this study: SVM and DT. Given the reported throughput measurements (using real and synthetic data) over various time periods, the classification problem was to detect whether or not a playback stall would occur. The DT model recorded a better performance than the

SVM model. Based on their findings, the authors hypothesized that even if the video data stream was encrypted, network operators could distinguish a significant fraction of low buffer instances.

Mazhar et al. in [157] leveraged network and transport-layer measurements as input features to train ML classifiers for predicting startup delay and rebuffering events in encrypted video data. They considered encrypted video traffic streamed over HTTPS and QUIC, which runs over UDP. Information in IP headers were used to determine network-layer features. TCP/UDP header information were used to extract transport-layer features. Some network-level measurements used in the study included: packet counts, byte counts, throughput, packet inter-arrival times, packet sizes in bytes. For the classification tasks, the authors employed DT classification algorithms. The tree-based ML models achieved up to 90% classification accuracy for HTTPS and up to 85% classification accuracy for QUIC in the experimental evaluations. Given the diversity of ML algorithms, the study requires further investigation. The works described in [156,157] proposed models that operate in controlled environments for certain services; the ideas could be advanced by exploring these techniques in real-world network deployment scenarios and to a broader range of services.

4.2.2. Session-Level Video Quality Prediction:

Bronzino et al. in [158] (which builds upon the works in [156,157]), extended the framework to account for a wider variety of services. The authors proposed models that work in deployment scenarios where video sessions and segments could be detected from a mix of traffic. The gathered traffic statistics have a coarser time precision. The authors devised a single composite model that could be used for a range of platforms (e.g., Netflix, YouTube, Amazon, and Twitch), rather than just one. They created a complete labeled set with over 13,000 video sessions for the four video services. Using network, transport, and application-layer measurements as inputs, the study investigated the feasibility of designing ML models that could predict startup delay and resolution. Some QoD features used in the study included: throughput, average packet count, average downstream throughput difference between consecutive time slots, and packet inter-arrival times. For the task of predicting the startup delay, the authors evaluated different regression methods, including: LR, RR, SVR, DT, and RF regressors. Using the average absolute error as a performance metric, the study reported that the RF model accounted for the lowest prediction errors. Similarly to [147], the authors trained ML classifiers with five classes, 240p, 360p, 480p, 720p, and 1080p, for the video resolution. The RF was reported as having achieved the best performance in terms of precision and recall. For the composite models, the authors reported findings that suggested that models which included the network and application-layer features outperformed models that depended solely on network and transport-layer features for the startup delay and resolution prediction tasks. For most video sessions data used in the study, startup delay models achieved less than one second error; the average precision of resolution models recorded was above 0.93.

In a related effort, the authors of [159] considered a composite model capable of predicting video quality for the popular VoD services (Netflix, Amazon Prime and YouTube). They extracted network-related features from streaming sessions running over the VoD services based on the streaming patterns and characteristics. Using the extracted features, they trained a single-layer perceptron NN. QoD features used in the study included packet size, port information, bandwidth, resolution, rebuffering time, and bitrate data. Other features used included block size, buffering phase, standard deviation of duration, and progressive download ratio. Results reported indicated that the NN achieved prediction accuracies of 0.929, 0.857, and 0.9333 for YouTube, Amazon and Netflix, respectively. It is not clear if the authors evaluated the NN model using trained models from another service—for example, by evaluating the NN on Netflix data using a YouTube trained NN model or a dataset containing all the services.

Schwarzmann et al. in [160] investigated session-level MOS prediction using ML regression models based on 5G monitoring data. Using QoD statistics from traces generated within an OMNeT++ simulation, the authors tested the feasibility of their technique. QoD measurements used in the study included throughput measurements (access node, user equipment (UE)), downlink channel quality indicator (CQI), uplink CQI measured at UE, RTT measured at the UE, and smoothed RTT computed with a moving average. Two ML regression models, LR and SVR, were evaluated in the study. Results reported indicated that subjective QoE scores could be reliably predicted, solely from network-QoD monitoring data. It was also shown that a limited set of features could lead to a reasonable accuracy in predictions. However, the practicality of this approach could be advanced by testing with real 5G network traces.

As the traffic characteristics of realistic video sessions monitored in the network are inherently affected by end user behavior [161] (e.g., seeking, pausing, and abandoning), Bartolec et al. in [162] investigated these events in their work on video quality KPI classification. The research examined the impact of user interactions on video KPI classification accuracy (resolution, initial delay, video bitrate). Using YouTube as the case study, the authors trained models on datasets that included and excluded user interactions. Two datasets containing IP-level traffic features were used in the study. A dataset containing 299 videos (without user interaction) and another comprising 307 videos (with user interactions) were utilized in training the RF model adopted in the work. The authors compared the performance of their models with ground-truth data from YouTube Stats for Nerds. The Stats for Nerds option displays some video meta-data regarding the stream playback. Results showed that the performance of models trained on datasets not including user interactions were worse when applied to the interactive dataset. The system performance will need to be evaluated with more realistic dataset and under additional scenarios such as a user-initiated quality switch.

In [163,164], Orsolic et al. investigated session-level MOS and video KPI classification using IP-level traffic features. The work in [163] used ML models namely, OneR, J48, and RF, in predicting the video resolution, stalling, startup delay, and bitrates. Two datasets were collected in laboratory emulated environments. A total of 394 videos were collected on Android and 383 videos on iOS. Two additional datasets were collected from mobile networks running in iOS. To compare their prediction with ground-truth data, the authors collected data from Stats from Nerds. QoD-level measurements used in the study included: average throughput and average packet size. The initial results reported in the work demonstrated promising applicability; however, further tests using datasets with different quality degradation scenarios maybe required. The study in [164] considered the same targets and used similar QoD inputs in the prediction tasks. In this study, four datasets were used. With the data collected on a laboratory WiFi network, the authors trained ML models to classify the video quality in terms of a range of video streaming KPI metrics (bitrate, resolution, stall events, initial delay) on the iOS platform. The authors used the same ML models: OneR, RF, and J48 (DT). The results reported in the study showed that the prediction accuracy, for all classification targets, ranged from 70 to 90%. Table 3 provides a summary of the studies surveyed in this section.

Table 3. Summary of video quality prediction from encrypted traffic.

Author, Year	Objective, ML Techniques	Features	Key Results
Orsolic et al. [138]	• Predicting startup delay and resolution in real-time • OneR, SMO, J48 DT, RF and NB	• Throughput, packet count, inter-arrival times, byte counts	• Best performance recorded by RF
Dimopoulos et al. [142]	• Prediction of video resolution and stall events in cellular networks • RF	• RTT, paket loss and video chunk sizes	• RF model recorded ≈94% accuracy
Wassermann et al. [144]	• Prediction of initial delay, stalling ratio, stall frequency, stall duration and user engagement for YouTube videos streamed on smartphones • RF only for video KPI predictions • User-engagement prediction: RF, SVM, kNN, NB, GB and AdaBoost	• Byte count, signal strength, network switches, packet-level data	• Achieved ≈90% accuracy for user-engagement predictions by using ensemble stacked trees
Wassermann et al. [147]	• Real-time prediction of YouTube video resolution • RF10, DT, AdaBoost, ERT10, BAGGING, NB, kNN, FFN, SVM	• Packet count, byte count	• Framework for real-time prediction of video resolution • RF10 recorded the best performance
Wassermann et al. [151]	• Real-time prediction of average bitrate and video resolution • RF10 and kNN	• Packet sizes, arrival times and DNS lookup responses	• Fine-grained bitrate and resolution on one second windows
Gutterman et al. [152,153]	• Requet: prediction of buffer warning, video state and resolution in real-time • RF	• TTFB, IP header information, download times, slack time, chunk duration and size	• Requet best performance was achieved with ≈98% accuracy for buffer warning; worst performance recorded was ≈67% accuracy for video resolution
Seufert et al. [154,155]	• ViCrypt: Prediction of rebuffering events and stalling • RF	• Packet counts, byte counts, TCP ratio, UDP ratio, throughput	• RF model recorded good performance at detecting stalls in real-time • Evaluated effects of relevant features; results showed that not all features were important to detect rebuffering events

Table 3. *Cont.*

Author, Year	Objective, ML Techniques	Features	Key Results
Krishnamoorthi et al. [156]	• BUFFEST: Video classification based on client buffer conditions as a proxy to estimate video quality; detection of rebuffering events in YouTube video • SVM and DT	• HTTP data, throughput and stream data	• DT model performed better than the SVM • Model able to detect buffer conditions accurately
Mazhar and Shafiq [157]	• Prediciton of start-up and rebuffering events for HTTPS and QUIC streaming • DT	• Packet counts, byte counts, throughput, inter-arrival times, packet sizes	• The DT model recorded 90% accuracy for HTTPS streaming and up to 85% accuracy for QUIC
Bronzino et al. [158]	• Prediciton of start-up and video resolution across multiple services (Netflix, YouTube, Amazon and Twitch) • LR, RR, SVR, DT and RF	• Throughput, packet count, inter-arrival times	• RF recorded lowest prediction errors for startup delay • Average precision for resolution classification models was above 0.93 • Framework able to generize to a range of services along as the training data included data from all services
Pandey et al. [159]	• Prediction of resolution, bitrate, rebuffering in VoD services (YouTube, Amazon and Netflix) • Single-layer Perceptron NN	• Bandwidth, resolution, bitrate, block size, progressive download ratio, buffering data	• NN achieved prediction accuracies of 0.929, 0.857 and 0.9333 for YouTube, Amazon and Netflix, respectively
Schwarzmann et al. [160]	• Session-level MOS prediction on 5G data • LR and SVR	• Throughput, CQI, RTT	• Novel work on inferring the correlation of 5G data to video KPIs
Bartolec et al. [162]	• Impact of user interactions on video KPI classification (resolution, startup delay, video bitrate) • RF	• Two IP-level traffic dataset (with user interactions and without user interactions)	• Models trained on data without user interactions recorded worse performance on data containing user interactions • Extended experiments should include more realistic scenarios such as a user-intiated switch
Orsolic et al. [163] Oršolić et al. [164]	• Session-level MOS and video KPI classification (video resolution, stalling, startup delay and bitrate) • OneR, J48 and RF	• IP-level traffic	• Prediction accuracy ranged from 70 to 90% • Models results showed the feasibility of accurate classification of YouTube videos into various KPI classes

In this section, we show that the input data which were used to train the models varies significantly across different applications, and among different usage scenarios within a single service. For most of the described scenarios, extracting ground-truth data from the application was not feasible. For instance, while YouTube provides video performance metrics in its Stats for Nerds window on desktop and mobile applications, the case is different for services such as Twitch and Netflix. These services only offer video performance reports in the browser. We note the differences in datasets, methodology, validation setups, and ML models for the same target objectives in these published papers.

However, the features and labels to be extracted from the collected data should be based on the desired objectives. If the objective is to achieve real-time network management, the ground-truth data used for model training should be representative of the conditions that need to be identified. For example, in order to predict instantaneous stall conditions in network traffic, the model must be trained on short time-windows of network traffic. If the desired objective is to provision the network to meet the time-varying needs of the streaming population, estimating the video quality on a session-level might be more suitable. What is more challenging is identifying features in the network that correlate with the target video quality KPIs. The authors of [67] highlighted some network-level measurements appropriate to detecting certain video quality KPIs, such the startup delay, stalling events, resolution, and quality switches.

From the reviews conducted in this section, we note that the current best candidate approach for ML is the RF algorithm. The RF works well for both regression and classification tasks. The majority of works surveyed in this section reported the RF as the sole ML model or evaluated the RF along with a range of other ML algorithms.

The RF seems to be an attractive choice in these settings due to the fact that its default parameters produce good prediction results without much tuning. The hyperparameters are very easy to understand, and there are not a lot of them. The RF is a great choice when dealing with high dimensional data. This set of algorithms is a good candidate when handling unbalanced data. This is because the model attempts to minimize the overall error rate by assigning a low error rate to the larger class, and the smaller class is assigned a larger error rate. However, when dealing with large datasets, the RF may take up a large amount of memory, making it quite resource intensive. In such cases, the DT may offer a lightweight approach for in-network computations at the expense of performance.

4.3. QoD Prediction for HAS and DASH

Following the widespread deployment of HAS, a lot of interest has been generated in the area. Research in this area has included some data-driven approaches, deployed with ML models for the prediction of network QoD changes. These studies investigated predictions of network metrics—namely, the throughput and trigger adaptation mechanism aimed at reducing rebuffering at the video client players [165] and selection of the appropriate adaptation action in HAS [166].

Sun et al. proposed cross section stateful predictor (CS2P) to enhance bitrate selection and adaptation in HAS clients in [165]. They made three contributions in their paper. The authors analyzed the throughput characteristics using a dataset sourced from a Chinese video provider, which comprised 20 million sessions covering three million unique client IPs, 8 server IPs, and 87 ISPs. They concluded from the analysis that sessions sharing similar key features (ISP, geographical region, etc.) have similarities in network-layer throughput values and dynamic patterns. They also alluded that there was also an inherent natural stateful presence in throughput variability within a session. However, the relationship between session features and throughput is rather a complex one. A video client's perceived throughput is influenced by multiple factors, such as the load on the server [167], network congestion, and last-mile link technology, which implies that clients sharing one of these features do not have similarities in throughput. A second contribution in the paper was the implementation of CS2P using a hidden Markov model (HMM) [168]. The authors used the HMM to model the state transition evolution of the throughput, one model per session cluster, where sessions were clustered by similar characteristics, as mentioned above (e.g., ISP and region). Finally, the authors integrated CS2P in a dash.js client. They demonstrated that it achieved accurate throughput prediction in a real world environment. Using the above-mentioned dataset, the HMM model was trained offline using the EM algorithm, and 4-fold cross validation was applied to tune the number of states. The authors reported that CS2P achieved ≈40% and 50% improvements over existing predictive approaches in terms of initial and midstream throughput prediction error. They compared their results with the model predictive control (MPC) [104], and reported that CS2P achieved an improvement of

3.2% in overall video quality, and 10.9% higher average bitrate compared to state-of-the-art MPC models which used the harmonic mean for throughput prediction.

The authors of [169] described a ML framework that used a range of regression models to predict the video quality of a streaming session. In their approach, each streaming session was defined by five statistics: the average of video segment quality values, the time over which segment quality decreases occurred, the time since the last impairment event, the total stalling duration, and the number of stalling events. The authors adopted three regression models: LR; SVR; and an ensemble approach using RF, GB, and extra trees regression. SVR was determined to have the best average prediction performance when using a dataset of 112 sessions with a length of about 72 s. However, these streaming session measurements do not adequately reflect the magnitude of quality degradation. This is because the temporal relationships between the impairment events are normally lost. This sort of predictor could be integrated with practical video-quality-aware decision algorithms. The use of such systems could aid in intelligent network resource allocation, thereby increasing the video quality and leading to a reduction in the costs incurred by streaming providers and content delivery networks.

A related study which considered the prediction of video quality for HAS streaming sessions was proposed in [170]. In this study, the authors presented a ML approach for predicting the overall quality of HAS sessions. The authors adopted LSTM networks [171] in the study. The authors made the case that LSTMs can utilize memory to explore temporal relationships between QoD impairment events. In the proposed technique, inputs were taken on a segment-by-segment basis rather than on a session-by-session basis. The authors evaluated their proposed technique over a range of datasets using two different types of LSTM, namely, basic and advanced. Segment quality, content qualities, stalling duration, and padding were some features considered in the study. The authors reported high performance for the LSTM networks in the overall quality prediction of HAS. However, this came at the expense of the method being computationally expensive. Although, the study reported promising results for all use cases considered, the scalability of this technique due to its high computational cost poses a challenge for practical large-scale deployments.

A hybrid-DL architecture for proactive prediction of the video quality from multivariate time-series data was proposed in [172]. The authors hypothesized that using QoD and some video KPI information, their DL framework could predict the client video quality at the next time step before the event occurred. The authors evaluated their technique using data from an industry video streaming testbed for three different scenarios: congestion-free, congestion in the client network, and congested ISP network. QoD features used in the experimental evaluations included buffering length and frequency, and bitrate data. To evaluate the proposed approach, the authors employed a hybrid model comprising a bidirectional LSTM (BiLSTM) and a convolutional neural network (CNN), which they called BiLSTM-CNN. The proposed hybrid model was compared with some other ML algorithms, such as SVR, and some DL models, such as MLP, LSTM, and BiLSTM. The results showed that the proposed hybrid model, BiLSTM-CNN, outperformed the other models. The SVR had the worst performance, followed by the MLP. Similar results were achieved by the Bi-LSTM and the LSTM. These results are promising, as they could enable network managers initiate remedial or proactive actions in advance of the occurrence of failures. Table 4 contains a summary of the key contributions discussed in this section.

4.4. Software-Defined Networking (SDN)

The heterogeneous nature of today's networks increases the complexity of networks and presents a number of challenges in effectively organizing, managing, and optimizing network resources. SDN is a networking architecture that decouples the control plane from the data plane [173]. The separation of concerns in SDN gives more flexibility to the network. Network resources in SDN are centrally managed by a logical controller. In addition to this, SDN has more advantages over traditional networks, including network automation, reduced network complexity, and more agility within the network.

Table 4. Summary of QoD prediction for HAS and DASH.

Author, Year	Objective, ML Techniques	Features	Key Results
Sun et al. [165]	• CS2P: Throughput prediction for bitrate adaption in HAS clients to improve video quality • HMM	• Throughput measurements	• Compared with baseline MPC methods, CS2P recorded ≈3.2% and 10.9% performance gains in video quality and average bitrate predictions, respectively
Bampis and Bovik [169]	• Video quality prediction in the presence of bitrate changes and stalling • LR, SVR, RF, ET, GB	• Time since video impairment, stall duration, frequency of stalls, segment degradation duration	• SVR is determined to have the best average prediction performance when using a dataset of 112 sessions with a length of about 72 s
Tran et al. [170]	• Prediction of overall video quality for HAS streaming sessions • LSTM	• Segment quality, content quality, stalling duration, padding	• Achieved high performance in overall video quality prediction with LSTMs
Dinaki et al. [172]	• BiLSTM-CNN: Hybrid DL architecture for predicting video quality • SVR, MLP, LSTM, BiLSTM, BiLSTM-CNN	• Buffer length and frequency, bitrate	• BiLSTM-CNN outperformed the other models; worst performance recorded by the SVR model

Owing to the centralized role of the SDN controller, it can monitor and gather real-time network state, flow statistics (e.g., throughput, packet loss), and configuration data. These features make ML algorithms appealing for SDN. The work in [174] is a recent survey of ML deployments in SDN.

Network operators rely on QoD metrics such as packet loss ratio, delay, jitter, and network throughput to assess network performance. These QoD metrics correspond to the network KPIs—for instance, transmission rates, queue length, etc.—upon which service-level agreements are made. Quantifying the relationships between these QoD metrics and the KPIs can help improve QoD management in SDNs by offering QoD prediction ability in relation to the KPIs. SDN's centralized architecture makes it possible to gather network statistics from the switches at per port and per flow granularity levels, based on which ML can be utilized to realize QoD predictions [174]. Josep et al. in [175] evaluated the performances of two models to characterize the network delay in an SDN network given the network load and the overlay routing policy. The authors considered a traditional M/M/1-inspired ML regressor network model and a NN model in their delay estimation study. Their experimental simulation results showed that the NN estimator performed better than the M/M/1-based delay estimator. However, both models treated the network as a *black box* without considering the network dynamics within the system. We argue that to quantify the effectiveness of these delay estimators in a streaming infrastructure, it would be beneficial to model dynamicity in the network. This could include the effects of concurrent users or interfering loads on the system, the variance in the delay, routing changes, congestion, etc. In addition, a more-detailed description of the trained NN model would aid better understanding and comparison of the approach in relation to NN models.

The authors of [176] proposed a two-step systematic technique towards QoD improvements in an SDN. They gathered a large number of metrics from virtualized and real environments. They then applied ML algorithms to automatically discover a formulae that could quantify the relationships between the network KPIs and the QoD metrics. They first used DTs to determine the correlations between the network KPIs and the QoD parameters. Thereafter, using a LR ML model, they performed root cause analysis to uncover each KPI's quantitative impact. The proposed approach could also be used to

predict traffic congestion and proffer recommendations on QoD improvements to the SDN controller [177]. However, because of a lack of uniformity in networks, rules in one may not work in another. It is therefore pertinent that the network operator adopts this approach and uncovers the correlations or rules as they relate to the network.

The work described in [178] by Rafael et al. focused on application-aware QoD predictions. The work described the use of two ML techniques, namely, RF and regression trees, to predict two QoD parameters: VFR and response time. The authors collected device (or kernel) metrics from a VoD server. Additionally, they collected statistics from the switches on a per port and flow granularity levels in the SDN. To reduce the computational cost, the authors employed subset selection to reduce the feature set size while preserving a low level of QoD prediction error. The authors reported an application-aware QoD prediction accuracy of over 90%. However, given that the setup for this was similar to [9], as was the feature set, incorporating the system load [40] would have better modeled the problem, as demonstrated by the work in [9]. The study in [179] utilized regression ML models to predict mean opinion score (MOS). The authors estimated the MOS value from the QoD parameters such as RTT, jitter, link bandwidth, and delay. The authors reported that the SDN controller was able to dynamically adjust the video parameters (e.g., screen resolution, frames per second, and bitrate) to improve the video quality received at the client devices. This evaluation was carried out using synthetic data, and not a lot of details were provided on the ML models.

To prevent the occurence of video freezes for HAS clients, Petrangeli et al. [180] proposed an SDN-based approach, where intermediate network elements support video delivery. The authors presented a ML-based framework designed for helping clients avoid video freezes due to network congestion. The ML engine is based on the Random Undersampling Boosting (RUSBoost) algorithm [181] and fuzzy logic. In the proposed framework, an SDN controller used an ML engine to predict when a HAS client would experience a video freeze so that it could determine if the currently downloaded segment should be prioritized. The decision was made using measurements collected only from network nodes, with no additional input from clients required. The ML freeze predictor used QoD input parameters such as the following: (i) the bandwidth for HAS traffic upon segment request; (ii) the difference in HAS bandwidth between two consecutive samples; (iii) the inter-arrival time between consecutive *get* requests. Other input features used included the quality level for the segment requested (expressed from 0 to q_{max}, the maximum available), and the difference in segment quality between consecutive requests. The authors collected 4500 clients' logs, which amounted to about 1.5 million individual segment requests, with 2.5% being affected by video freeze. They trained the predictor using 85% of the training data and set apart the remaining 15% as the validation dataset.

The performance of the RUSBoost algorithm was compared wit RF, AdaBoost, GB, and 1-nearest neighborhood (1-NN). The authors reported findings which showed that the RUSBoost algorithm outperformed the other classifiers with classification accuracies of 99% and 85% for the training data and validation set, respectively. Results showed that the RUSBoost algorithm could detect if a client was close to freezing, and fuzzy logic confirmed if the queue was good enough to successfully prioritize video segments. Table 5 summarizes the contributions surveyed in this section.

Table 5. Summary of QoD prediction in SDN environments.

Author, Year	Objective, ML Techniques	Features	Key Results
Pasquini and Stadler [178]	• Application-aware QoD (VFR and response time) prediction • RF and regressed trees with subset selection	• VoD server kernel metrics, switch port and flow statistics	• RF outperformed the regression tree models • This performance gain was achieved at the cost of the RF having a 100 times longer model computation time
Ben Letaifa [179]	• MOS prediction • Regression models	• RTT, jitter, bandwidth and delay	• Video quality improvement recorded by the adaptive response of the SDN controller
Petrangeli et al. [180]	• Preventing video freezes due to network congestion • RUSBoost, RF, AdaBoost, GB	• Bandwidth, inter-arrival times, segment quality level and difference in segement quality levels for consecutive requests	• Best performance recorded by the RUSBoost algorithm with a classification accuracy of 99% and 85% for the training data and validation set, respectively • RUSBoost algorithm able to detect if a client was close to freezing

4.5. Predicting the Video Quality in Wireless Settings

Video streaming quality can degrade in wireless settings due to performance issues. In such environments, there are various performance indicators, such as link speed, signal strength, interference, medium availability, and latency. In this section, we review the literature on studies that have investigated the prediction of the video quality based on network performance parameters, specifically in Wi-Fi networks.

Hora et al. [182] measured the frequency and duration of rebuffering events, join time, and resolution switches of the YouTube application in an attempt to predict the MOS. The authors collated QoD parameters such as the transmit physical rate, total number of frames sent and retransmitted to station, received signal strength indicator (RSSI), and medium busy time. They used existing models which predicted the YouTube application's quality as a function of rebuffering events and video bitrate as proposed in [183]. Similarly, for video bitrate changes, they used SSIM to estimate the effect of video resolution on the streamed video quality as proposed in [184]. During the training phase, the authors employed stepwise feature engineering to incrementally increase the number of features used to a maximum of 36 features. They trained an SVR algorithm to predict the video quality and reported a root mean squared error (RMSE) of 0.60–0.79.

Ligata et al. [185] defined four binary classes of the video quality in terms of video streaming buffer ratio, initial buffer delay, rebuffering frequency, and rebuffering duration. In the study, QoD parameters considered were signal strength (SS), retransmission ratio, error ratio, activity factor, channel utilization, and exposed node load. To determine which QoD parameters correlate with the four WiFi problems, such as coverage, overload, contention, and interference, the researchers used Pearson correlation coefficients and selected the features that had the highest correlations. The authors then trained a RF algorithm to predict the impact of these QoD metrics on the binary classes. They reported prediction accuracies of 85–95%.

For priority queue management, clients can measure or report the video quality directly to the access point (AP) and controller. FlowBazaar system was presented by Bhattacharyya et al. in [186]. The system leveraged middleware on the client side which sampled the video state, and mapped this state to a delivery quality score (DQS). Appli-

cation performance metrics considered by DQS included stall duration and number of stalls during streaming. Clients then requested bids in the range of [0–5], with the top two bidders being selected for the queue assignment, and the third highest bid for other clients being charged. Using the OpenFlow protocol, the controller made policy decisions (such as allocating flows to queues) that were communicated to the AP. At the AP, the Statistics Collection component collected QoD statistics such as throughput, drop rates, and RTT. These measurements were then sent to the controller in a predefined message format using the OpenFlow protocol. The authors used RL to select which queue should receive clients based on network latency, packet loss, and throughput for the next period of time. The auction-based approach achieved DSQ 5 for almost 85–90% of clients. Table 6 provides a summary of the contributions surveyed in this section.

Table 6. Summary of video quality prediction in wireless settings.

Author, Year	Objective, ML Techniques	Features	Key Results
Da Hora et al. [182]	• Prediction of MOS with rebuffering events, join time and resolution switches in YouTube • SVR	• Transmit rates, frame count, RSSI, medium busy time	• SVR models achieved accurate video quality prediction with a RMSE of 0.60–0.79
Ligata et al. [185]	• Prediction of streaming buffer ratio, initial buffer delay, rebuffering frequency and rebuffering duration • RF	• SS, retransmission ratio, error ratio, activity factor, channel utilization and exposed node load	• RF models recorded accuracies of ≈85–95%
Bhattacharyya et al. [186]	• FlowBazaar: priority queue management • RL	• Throughput, drop rates, RTT	• FlowBazaar auction-based technique recorded DSQ 5 accuracies of ≈85–95%

4.6. Predicting Video Quality in WebRTC

The authors of [187] proposed a method to address the problem service providers face in monitoring streaming quality of WebRTC-based audiovisual communication services. They examined the use of ML models in identifying root causes of video quality problems by extracting features from various application-layer performance statistics. They targets features considered in the prediction tasks were video blockiness, audio distortion, and identification of the root causes of impairments. They authors employed six ML models in these classification tasks, namely: DT, RF, NB, SMO, kNN, and bagging. QoD input features used in the study included the frame rate, packet loss count, jitter, jitter buffer size, and bitrate. The DT model achieved high accuracy when detecting all three targets. According to the authors, due to the high interpretability of DTs and space constraints, they reported only results with the DT models. Nevertheless, they reported that the other models had comparable results.

In a related effort, the authors of [188] conducted evaluations using WebRTC measurements from a WiFi network. Using QoD metrics such as WiFi RTT, Link Quality, and RSSI, the authors employed ML models to predict whether the video quality would be acceptable during the next time window. They focused on video freezing events. ML models used in the study were DTs, RFs, SVMs, and extra trees classifier. Results reported showed that the RF outperformed the other models. These techniques aim to produce self-healing frameworks where the system can adjust its servicing strategy. The results can be improved by adopting multi-dimensional models and by incorporating a broader range of realistic and perceptible audio-visual impairments into the evaluations.

5. Discussion and Future Directions

We discuss the future of this field of research in this section, focusing on areas to which we believe more attention needs to be given in the years to come.

(A) ML Dominance over DL: The applications of ML-based techniques for video quality prediction from QoD measurements show much promise, given that these ML models are capable of making accurate predictions from the video stream data. As we noticed in this study, the vast majority of studies used ML over DL in video streaming networks. Most of these studies were based on offline models, for which they authors used ML algorithms in training the data in batches before they could be applied to decision making. Video streaming networks, however, often exhibit dynamic variations over time, e.g., due to network state changes or QoD degradation [189]. Network state acquisition via QoD metrics can be fed into ML models at the same pace as the rate of change in the service [190]. We envisage that once an algorithm has been trained using past samples, it may be possible to implement various types of ML algorithms in an online fashion [191] to gradually include new input data as they are made available by the network. During the training process of an offline learning model, the ML model's parameters and weights are updated while trying to optimize the cost function [192] using the data it was trained on. When an online learning process is used, the learned parameters are dependent on the currently seen samples, and possibly on the state of the model at this stage. As a result, the model is continuously learning new data and improving the learned parameters, which makes the learning framework adaptive. SDN and ML integrated frameworks could be used in cases where the dynamics of online learning may pose some challenges, as demonstrated in [73,193]. These studies make a case for an adaptive learning framework considering the real-time constraints for video streaming.

(B) Adaptive Deep Learning for Improved Video Delivery: Advances in DL technology present new possibilities that could transform the video delivery system. Recent developments in big data, and advances in algorithm development, virtualization, and cloud computing enable DL to be used in a range of applications, such as computer vision and speech recognition. From the studies surveyed, we found that the dominant DL architectures for video quality predictions were the LSTM and CNN architectures. LSTM networks have been used in time-series problems [194] and offer the possibility of pooling inputs. They are also able to exploit temporal dependencies between impairment events in a sequence by the use of memory [195]. For most prediction tasks, LSTM's capability to retain knowledge of previous states makes it an ideal algorithm for most experimental evaluations described in this survey. LSTM, a variant of the recurrent neural network (RNN), provides an effective solution to the problem of vanishing gradients during backpropagation of errors in an RNN. The vanishing gradient problem occurs when the error signal used to train the network gradually reduces as one moves backwards in the network during backpropagation. This has the consequence that the layers closer to the input do not get trained. An LSTM employs a gating mechanism that controls the memoizing process. Information in LSTMs is written, read, or stored by opening and closing gates. Previous studies show the feasibility of LSTM networks for real-time video quality predictions [196] and service response time predictions [197]. CNNs are very effective at image processing and computer vision [198,199]. CNNs have also proven useful for video streaming services [200]. The studies in [200,201] presented some directions in which DL could be applied to improve the quality of video delivery. The authors of [200] proposed the use of a CNN to enable parallel encoding of video for HAS. In parallel encoding, frames of a compressed video serve as a reference to define future frames. This speeds up the process of encoding multiple representations of video data. Most state-of-art techniques utilize the highest quality representation as a reference for encoding the video data. The authors hypothesized that by using the representation with the lowest quality, the encoding process would be relatively improved. The authors of [201]

demonstrated how by using DNNs and the improved computational power of the client devices, their technique could leverage redundant information in video data to boost the streaming quality when bandwidth availability was constrained. DNNs allow for the extraction of important features from images. The authors proposed a content-aware DNN model that achieves a significant boost in image resolution and uses the improved computational power of client devices to improve the video quality. These studies highlight the potential of DL in the video delivery system. Given the glut of video-driven data and high computational requirements of DL-based models, it is imperative that these techniques enable real-time, online, and adaptive analysis of the video data. The performances of trained network models may decrease over time due to changes in the video data, network conditions, or even unknown features. In such cases, the inputs used in training the network will vary significantly. The use of an adaptive DL would enable on-the-fly learning, as such a model detects and reacts to changes after deployment in highly dimensional data streams. The studies in [202,203] proposed adaptive DL frameworks for dynamic image classification in IoT environments and real-time image classification, respectively. However, enabling real-time DL poses some challenges. Additional layers of a network may increase accuracy, but they require considerably more compute power and memory. At present, ML has the advantage of having being evaluated first for video QoD predictions and is therefore running in current deployments and future deployments of ML-based solutions are underway. The slower uptake of DL solutions is explained by (1) the complexity of the data models which make it extremely expensive to train; (2) issues with interpreting results; and (3) the need for retraining and up-skilling network engineers.

(C) Computational Cost and Interpretability: A good number of the studies surveyed used DT, RF, NB, and SVM. These four ML algorithms seem popular due to their simplicity and easier interpretation in comparison with DL. The use of RF in batch settings is becoming increasingly popular due to the benefits it provides in terms of learning performance and having little demand for input preparation and hyper-parameter tuning [204]. These models in the majority of cases resulted in the best prediction and classification accuracies. Interpretability has been emphasized alongside accuracy in the literature [205]. Some authors note the importance of comparing other parameters than accuracy when two models exhibit the same accuracy [206,207]. They have attempted to establish a link between the interpretability and usability of models. They argue that it is beneficial for ML and network practitioners to work with easy to understand ML models. This may be important in model selection, feature engineering, and in trusting the prediction outcomes [208]. These algorithms incur shorter training times compared to DL. This makes their use ideal for these prediction tasks. Another possible reason for the dominance of ML models over DL could be attributed to the significant DL computational requirements in terms of power, memory, and resources. In centralized networks without resource constraints, such as SDNs, DL can be implemented by leveraging the centralized controller [209]. In limited storage settings such as with IoT, implementing DL can be challenging. The network provider has a choice among high computational requirements, accuracy, and interpretability. Future research should focus on identifying ways to transfer knowledge between tasks, which can be adapted to changing network environments and contexts [210].

(D) Self-Healing Networks and Failure Recovery: ML applications with SDN control offer some innovative possibilities for network failure recovery in video streaming services. Smart routing has been proposed to tackle some of these challenges posed by data link failures [211]. In contrast to existing approaches, the proposed approach allows the SDN controller to reconfigure the network before the anticipated failure of a link. This approach can not only reduce interruptions caused by links failing, but also bring significant benefits to increasing availability of the video streaming service. From the

studies we surveyed, we note some popular QoD KPIs such as rebuffering, quality switching, video resolution, and initial startup delay. High availability and smart routing mechanisms can aid the network in reducing or mitigating video artifacts which may arise as a result of rebuffering events, quality switches, and stalling. Integrating ML with SDN in this manner will aid in providing intelligence to ensure the streaming service continues interrupted.

When it comes to video quality prediction from network measurements, major difficulties are the dimensionality of the problem and the unpredictable changes that can occur over time within a single service [212]. These are major motivating factors for using QoD measurements, which are service-agnostic for these ML tasks. The issue of concept drift where the distribution for the evolving streams of data vary with time may be a challenge for real-time video streaming services. For example, in classification tasks, the feature set may change over a time period such that they do not reflect the class labels any more. We notice that a good number of the studies surveyed attempted to handle concept drift by using sliding windows [213]. These window-based approaches used sliding windows to detect drifts, rather than relying on the entire distribution. In some other cases, the models proposed seemed to suit a particular service or application. An interesting direction for future research lies in investigating the feasibility of applying representational and transfer learning in these areas, especially when suitable training data are be readily available.

Recent breakthroughs in DL have provided new techniques for handling missing data. For example, current imputation methods employ deep generative models, such as GANs [214] and autoencoders [215], to handle missing values. A limiting factor of these models is that they fail to incorporate the feature values from other candidate observations when estimating missing values for a given observation. The use of graphical models [216] has been proposed to train the learning framework in cases where suitable training data are unavailable. The study in [217] proposed a graph neural network (GNN), GRAPE, a graph-based system for feature imputation and label prediction. In the proposed framework, feature imputation was modeled as an edge-level prediction problem and label prediction as a node-level prediction problem. Referring back to the introductory example in this survey in Section 2, the authors of [34] formulated the LA technique as a graphical model in [167] to handle missing data in [218], for cases where there were no RTP packet count observations in the system for a determined number of users. The mappings from the features to the RTP packet count was conditional on a latent variable, which was the load on the system. Predictions of the RTP packet count for system loads where no previous data had been collected were possible by interpolating between the different states. Experimental evaluations of the proposed model demonstrated the suitability of the technique for the RTP packet count, i.e., the QoD metric prediction when there was missing data. These models are promising solutions for handling the lack of suitable training data.

6. Conclusions

In this paper, we surveyed the applications of ML techniques for video quality prediction with QoD metrics. The survey covered the video QoD prediction via ML in QoD degrading conditions, encrypted video stream traffic, HAS video services, SDN video streaming, video streaming over wireless networks, and WebRTC video streaming applications. It is of paramount importance for service providers and network operators to develop reliable models that are capable of monitoring, predicting, and even controlling the video quality in order to satisfy the ever-increasing demand for video services. We found that there are a number of research efforts that are looking at video quality prediction from QoD measurements for encrypted video traffic. The lack of visibility caused by encryption and digital rights management drives the need for ML-based QoD prediction. Furthermore, the survey showed that ML models are increasingly being used to predict video quality under poor QoD conditions. Several studies evaluated similar QoD video quality KPIs, such as rebuffering, quality switching, video resolution, and startup delay. For the application designers, the media player buffer design is crucial, because rebuffering impacts the

quality of the video stream. The buffer size affects startup delay and rebuffering time [219]. Most players are typically designed to buffer a predetermined number of segments, before initiating playback. With 1 and 2 second segments, this has been shown to be fine, and it provides under 10 second latency if there are no more than 3 segments being buffered [220].

Generally, it is envisaged that to design accurate ML models that can predict video quality from QoD statistics, the models should be reliable given the QoD metrics and all externally influencing factors. For instance, there is evidence that models which assume independency among QoD variables may be inaccurate, since, for example, bitrate and buffering are correlated [221]. In order to perform real-time video QoD via ML, the models must have acceptable computational complexities and storage requirements. Finally, to aid with the scalability of such proposals, the models should be able to readily take new variables into account as the network evolves over time.

Funding: This publication has emanated from research conducted with the financial support of Science Foundation Ireland (SFI) under the grant number 15/SIRG/3459.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Alzubaidi, L.; Zhang, J.; Humaidi, A.J.; Al-Dujaili, A.; Duan, Y.; Al-Shamma, O.; Santamaría, J.; Fadhel, M.A.; Al-Amidie, M.; Farhan, L. Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions. *J. Big Data* **2021**, *8*, 1–74.
2. What is Deep Learning? | IBM. Available online: https://www.ibm.com/cloud/learn/deep-learning (accessed on 14 October 2021).
3. Khan, A.I.; Al-Habsi, S. Machine Learning in Computer Vision. *Procedia Comput. Sci.* **2020**, *167*, 1444–1451. [CrossRef]
4. Zhiyan, H.; Jian, W. Speech Emotion Recognition Based on Deep Learning and Kernel Nonlinear PSVM. In Proceedings of the 2019 Chinese Control and Decision Conference (CCDC), Nanchang, China, 3–5 June 2019; pp. 1426–1430.
5. Padmanabhan, J.; Johnson Premkumar, M.J. Machine Learning in Automatic Speech Recognition: A Survey. *IETE Tech. Rev.* **2015**, *32*, 240–251. [CrossRef]
6. Haseeb, K.; Ahmad, I.; Awan, I.I.; Lloret, J.; Bosch, I. A Machine Learning SDN-Enabled Big Data Model for IoMT Systems. *Electronics* **2021**, *10*, 2228. [CrossRef]
7. Hashima, S.; ElHalawany, B.M.; Hatano, K.; Wu, K.; Mohamed, E.M. Leveraging Machine-Learning for D2D Communications in 5G/Beyond 5G Networks. *Electronics* **2021**, *10*, 169. [CrossRef]
8. Najm, I.A.; Hamoud, A.K.; Lloret, J.; Bosch, I. Machine learning prediction approach to enhance congestion control in 5G IoT environment. *Electronics* **2019**, *8*, 607. [CrossRef]
9. Izima, O.; de Fréin, R.; Davis, M. Video Quality Prediction Under Time-Varying Loads. In Proceedings of the 2018 IEEE International Conference on Cloud Computing Technology and Science (CloudCom), Nicosia, Cyprus, 10–13 December 2018; pp. 129–132.
10. Vega, M.T.; Perra, C.; Liotta, A. Resilience of Video Streaming Services to Network Impairments. *IEEE Trans. Broadcast.* **2018**, *64*, 220–234. [CrossRef]
11. Cisco Visual Networking Index: Forecast and Trends, 2017–2022. White Paper. Available online: https://tinyurl.com/29rtya2b (accessed on 1 October 2021).
12. ITU. 910. Subjective Video Quality Assessment Methods for Multimedia Applications. International Telecommunication Union Telecommunication Section 1999. Available online: https://www.itu.int/rec/T-REC-P.910-200804-I (accessed on 1 October 2021).
13. Chikkerur, S.; Sundaram, V.; Reisslein, M.; Karam, L.J. Objective Video Quality Assessment Methods: A classification, Review, and Performance Comparison. *IEEE Trans. Broadcast.* **2011**, *57*, 165–182. [CrossRef]
14. Recommendation of ITU. 1011-Reference Guide to Quality of Experience Assessment Methodologies. 2013. Available online: https://www.itu.int/rec/T-REC-G.1011-201607-I/en (accessed on 1 October 2021).
15. Bentaleb, A.; Taani, B.; Begen, A.C.; Timmerer, C.; Zimmermann, R. A Survey on Bitrate Adaptation Schemes for Streaming Media over HTTP. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 562–585. [CrossRef]
16. Duanmu, Z.; Zeng, K.; Ma, K.; Rehman, A.; Wang, Z. A Quality-of-Experience Index for Streaming Video. *IEEE J. Sel. Top. Signal Process.* **2016**, *11*, 154–166. [CrossRef]
17. Huynh-Thu, Q.; Ghanbari, M. Scope of Validity of PSNR in Image/Video Quality Assessment. *Electron. Lett.* **2008**, *44*, 800–801. [CrossRef]
18. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image Quality Assessment: From Error Visibility to Structural Similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612. [CrossRef]
19. Wolf, S.; Pinson, M. Reference Algorithm for Computing Peak Signal to Noise Ratio (PSNR) of a Video Sequence with a Constant Delay. *ITU-T Contribution COM9-C6-E.* Geneva Switzerland, 2–6 February 2009. Available online: https://www.its.bldrdoc.gov/publications/details.aspx?pub=2571 (accessed on 4 October 2021).

20. Varela, M.; Skorin-Kapov, L.; Ebrahimi, T. Quality of Service versus Quality of Experience. In *Quality of Experience*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 85–96.
21. E.800: Definitions of Terms Related to Quality of Service. Available online: https://www.itu.int/rec/T-REC-E.800-200809-I (accessed on 5 October 2021).
22. ETSI TR 102 157—V1.1.1—Satellite Earth Stations and Systems (SES); Broadband Satellite Multimedia; IP Interworking over Satellite; Performance, Availability and Quality of Service. Available online: https://tinyurl.com/kpu7w3m (accessed on 6 October 2021).
23. Minhas, T.N. Network Impact on Quality of Experience of Mobile Video. Ph.D. Thesis, Blekinge Institute of Technology, Karlshamn, Sweden, 2012.
24. Fiedler, M.; Zepernick, H.J.; Lundberg, L.; Arlos, P.; Pettersson, M.I. QoE-based Cross-layer Design of Mobile Video Systems: Challenges and Concepts. In Proceedings of the 2009 IEEE-RIVF International Conference on Computing and Communication Technologies, Danang, Vietnam, 13–17 July 2009; pp. 1–4.
25. Shinde, P.P.; Shah, S. A Review of Machine Learning and Deep Learning Applications. In Proceedings of the 2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA), Pune, India, 16–18 August 2018; pp. 1–6. [CrossRef]
26. Abiodun, O.I.; Jantan, A.; Omolara, A.E.; Dada, K.V.; Umar, A.M.; Linus, O.U.; Arshad, H.; Kazaure, A.A.; Gana, U.; Kiru, M.U. Comprehensive Review of Artificial Neural Network Applications to Pattern Recognition. *IEEE Access* **2019**, *7*, 158820–158846. [CrossRef]
27. Brink, H.; Richards, J.; Fetherolf, M. *Real-World Machine Learning*; Manning Publications: Shelter Island, NY, USA, 2017.
28. Weerts, H.J.; Mueller, A.C.; Vanschoren, J. Importance of Tuning Hyperparameters of Machine Learning Algorithms. *arXiv* **2020**, arXiv:2007.07588.
29. Wang, M.; Cui, Y.; Wang, X.; Xiao, S.; Jiang, J. Machine Learning for Networking: Workflow, Advances and Opportunities. *IEEE Netw.* **2017**, *32*, 92–99. [CrossRef]
30. Seber, G.A.; Lee, A.J. *Linear Regression Analysis*; John Wiley & Sons: Hoboken, NJ, USA, 2012; Volume 329.
31. Quinlan, J.R. Decision Trees and Decision-making. *IEEE Trans. Syst. Man Cybern.* **1990**, *20*, 339–346. [CrossRef]
32. Breiman, L. Random Forests. *Mach. Learn.* **2001**, *45*, 5–32. [CrossRef]
33. Wang, B.; Zou, D.; Ding, R. Support Vector Regression based Video Quality Prediction. In Proceedings of the 2011 IEEE International Symposium on Multimedia, Dana Point, CA, USA, 5–7 December 2011; pp. 476–481.
34. de Fréin, R. Effect of System Load on Video Service Metrics. In Proceedings of the 2015 26th Irish Signals and Systems Conference (ISSC), Carlow, Ireland, 24–25 June 2015; pp. 1–6.
35. Xu, R.; Wunsch, D. Survey of Clustering Algorithms. *IEEE Trans. Neural Net.* **2005**, *16*, 645–678. [CrossRef]
36. Hartigan, J.A.; Wong, M.A. Algorithm AS 136: A K-means Clustering Algorithm. *J. R. Stat. Soc. Ser. C* **1979**, *28*, 100–108. [CrossRef]
37. Van Hulle, M.M. *Self-Organizing Maps*; 2012; pp. 585–622.
38. Moon, T.K. The Expectation-Maximization Algorithm. *IEEE Signal Process. Mag.* **1996**, *13*, 47–60. [CrossRef]
39. Goodfellow, I.; Pouget-Abadie, J.; Mirza, M.; Xu, B.; Warde-Farley, D.; Ozair, S.; Courville, A.; Bengio, Y. Generative Adversarial Nets. *Adv. Neural Inf. Process. Syst.* **2014**, *27*. [CrossRef]
40. de Fréin, R. Source Separation Approach to Video Quality Prediction in Computer Networks. *IEEE Commun. Lett.* **2016**, *20*, 1333–1336. [CrossRef]
41. Kaelbling, L.P.; Littman, M.L.; Moore, A.W. Reinforcement Learning: A Survey. *J. Artif. Intell. Res.* **1996**, *4*, 237–285. [CrossRef]
42. Mammeri, Z. Reinforcement Learning based Routing in Networks: Review and Classification of Approaches. *IEEE Access* **2019**, *7*, 55916–55950. [CrossRef]
43. Watkins, C.J.; Dayan, P. Q-learning. *Mach. Learn.* **1992**, *8*, 279–292. [CrossRef]
44. Li, Y. Deep Reinforcement Learning: An Overview. *arXiv* **2017**, arXiv:1701.07274.
45. Imran; Ghaffar, Z.; Alshahrani, A.; Fayaz, M.; Alghamdi, A.M.; Gwak, J. A Topical Review on Machine Learning, Software Defined Networking, Internet of Things Applications: Research Limitations and Challenges. *Electronics* **2021**, *10*, 880. [CrossRef]
46. Al-Garadi, M.A.; Mohamed, A.; Al-Ali, A.K.; Du, X.; Ali, I.; Guizani, M. A Survey of Machine and Deep Learning Methods for Internet of Things (IoT) Security. *IEEE Commun. Surv. Tutor.* **2020**, *22*, 1646–1685. [CrossRef]
47. Mahdavinejad, M.S.; Rezvan, M.; Barekatain, M.; Adibi, P.; Barnaghi, P.; Sheth, A.P. Machine learning for Internet of Things Data Analysis: A Survey. *Digit. Commun. Netw.* **2018**, *4*, 161–175. [CrossRef]
48. Cui, L.; Yang, S.; Chen, F.; Ming, Z.; Lu, N.; Qin, J. A Survey on Application of Machine Learning for Internet of Things. *Int. J. Mach. Learn. Cybern.* **2018**, *9*, 1399–1417. [CrossRef]
49. Miller, D.J.; Xiang, Z.; Kesidis, G. Adversarial Learning Targeting Deep Neural Network Classification: A Comprehensive Review of Defenses Against Attacks. *Process. IEEE* **2020**, *108*, 402–433. [CrossRef]
50. Tang, L.; Mahmoud, Q.H. A Survey of Machine Learning-Based Solutions for Phishing Website Detection. *Mach. Learn. Knowl. Extr.* **2021**, *3*, 672–694. [CrossRef]
51. Meshram, A.; Haas, C. Anomaly Detection in Industrial Networks using Machine Learning: A Roadmap. In *Machine Learning for Cyber Physical Systems*; Springer: Berlin/Heidelberg, Germany, 2017; pp. 65–72.

52. Hodo, E.; Bellekens, X.; Hamilton, A.; Tachtatzis, C.; Atkinson, R. Shallow and Deep Networks Intrusion Detection System: A Taxonomy and Survey. *arXiv* **2017**, arXiv:1701.02145.
53. Sultana, N.; Chilamkurti, N.; Peng, W.; Alhadad, R. Survey on SDN based Network Intrusion Detection System Using Machine Learning Approaches. *Peer -Peer Netw. Appl.* **2019**, *12*, 493–501. [CrossRef]
54. Buczak, A.L.; Guven, E. A Survey of Data Mining and Machine Learning Methods for Cyber Security Intrusion Detection. *IEEE Commun. Surv. Tutor.* **2016**, *18*, 1153–1176. [CrossRef]
55. Otoum, S.; Kantarci, B.; Mouftah, H.T. On the Feasibility of Deep Learning in Sensor Network Intrusion Detection. *IEEE Netw. Lett.* **2019**, *1*, 68–71. [CrossRef]
56. Sharma, H.; Haque, A.; Blaabjerg, F. Machine Learning in Wireless Sensor Networks for Smart Cities: A Survey. *Electronics* **2021**, *10*, 1012. [CrossRef]
57. Zhang, C.; Patras, P.; Haddadi, H. Deep Learning in Mobile and Wireless Networking: A Survey. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 2224–2287. [CrossRef]
58. Klaine, P.V.; Imran, M.A.; Onireti, O.; Souza, R.D. A Survey of Machine Learning Techniques Applied to Self-Organizing Cellular Networks. *IEEE Commun.* **2017**, *19*, 2392–2431. [CrossRef]
59. Musumeci, F.; Rottondi, C.; Nag, A.; Macaluso, I.; Zibar, D.; Ruffini, M.; Tornatore, M. An Overview on Application of Machine Learning Techniques in Optical Networks. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 1383–1408. [CrossRef]
60. Usama, M.; Qadir, J.; Raza, A.; Arif, H.; Yau, K.L.A.; Elkhatib, Y.; Hussain, A.; Al-Fuqaha, A. Unsupervised Machine Learning for Networking: Techniques, Applications and Research Challenges. *IEEE Access* **2019**, *7*, 65579–65615. [CrossRef]
61. Fadlullah, Z.M.; Tang, F.; Mao, B.; Kato, N.; Akashi, O.; Inoue, T.; Mizutani, K. State-of-the-Art Deep Learning: Evolving Machine Intelligence Toward Tomorrow's Intelligent Network Traffic Control Systems. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2432–2455. [CrossRef]
62. Boutaba, R.; Salahuddin, M.A.; Limam, N.; Ayoubi, S.; Shahriar, N.; Estrada-Solano, F.; Caicedo, O.M. A Comprehensive Survey on Machine Learning for Networking: Evolution, Applications and Research Opportunities. *J. Internet Serv. Appl.* **2018**, *9*, 1–99. [CrossRef]
63. Izima, O.; de Fréin, R.; Davis, M. Evaluating Load Adjusted Learning Strategies for Client Service Levels Prediction from Cloud-hosted Video Servers. In Proceedings of the 26th AIAI Irish Conference on Artificial Intelligence and Cognitive Science, Dublin, Ireland, 6–7 December 2018; Volume 2259, pp. 198–209.
64. Ridwan, M.A.; Radzi, N.A.M.; Abdullah, F.; Jalil, Y.E. Applications of Machine Learning in Networking: A Survey of Current Issues and Future Challenges. *IEEE Access* **2021**, *9*, 52523–52556. [CrossRef]
65. Oprea, S.; Martinez-Gonzalez, P.; Garcia-Garcia, A.; Castro-Vargas, J.A.; Orts-Escolano, S.; Rodríguez, J.G.; Argyros, A.A. A Review on Deep Learning Techniques for Video Prediction. *IEEE Trans. Pattern Anal. Mach. Intell.* **2020**. [CrossRef] [PubMed]
66. Aroussi, S.; Mellouk, A. Survey on Machine Learning-based QoE-QoS Correlation Models. In Proceedings of the 2014 International Conference on Computing, Management and Telecommunications (ComManTel), Da Nang, Vietnam, 27–29 April 2014; pp. 200–204.
67. Khokhar, M.J.; Ehlinger, T.; Barakat, C. From Network Traffic Measurements to QoE for Internet Video. In Proceedings of the 2019 IFIP Networking Conference (IFIP Networking), Warsaw, Poland, 22–22 May 2019; pp. 1–9.
68. Cheng, Y.; Geng, J.; Wang, Y.; Li, J.; Li, D.; Wu, J. Bridging Machine Learning and Computer Network Research: A Survey. *CCF Trans. Netw.* **2019**, *1*, 1–5. [CrossRef]
69. Marquardt, D.W.; Snee, R.D. Ridge regression in practice. *Am. Stat.* **1975**, *29*, 3–20.
70. Kukreja, S.L.; Löfberg, J.; Brenner, M.J. A Least Absolute Shrinkage and Selection Operator (LASSO) for Nonlinear System identification. *IFAC Proc. Vol.* **2006**, *39*, 814–819. [CrossRef]
71. Zou, H.; Hastie, T. Regularization and Variable Selection via the Elastic Net. *J. R. Stat. Soc. Ser. B* **2005**, *67*, 301–320. [CrossRef]
72. Izima, O.; de Fréin, R.; Davis, M. Predicting Quality of Delivery Metrics for Adaptive Video Codec Sessions. In Proceedings of the 2020 IEEE 9th International Conference on Cloud Networking (CloudNet), Piscataway, NJ, USA, 9–11 November 2020; pp. 1–7.
73. Izima, O.; de Fréin, R.; Malik, A. Codec-Aware Video Delivery Over SDNs. In Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), Bordeaux, France, 17–21 May 2021; pp. 732–733.
74. Hartsell, T.; Yuen, S.C.Y. Video Streaming in Online Learning. *AACE J.* **2006**, *14*, 31–43.
75. Li, X.; Darwich, M.; Salehi, M.A.; Bayoumi, M. A Survey on Cloud-based Video Streaming Services. In *Advances in Computers*; Elsevier: Amsterdam, The Netherlands, 2021; Volume 123, pp. 193–244.
76. Lao, F.; Zhang, X.; Guo, Z. Parallelizing Video Transcoding Using Map-Reduce-based Cloud Computing. In Proceedings of the 2012 IEEE International Symposium on Circuits and Systems (ISCAS), Seoul, Korea, 20–23 May 2012; pp. 2905–2908.
77. Varma, S. *Internet Congestion Control*, 1st ed.; Morgan Kaufmann Publishers Inc.: San Francisco, CA, USA, 2015; pp. 173–203.
78. Wu, D.; Hou, Y.T.; Zhu, W.; Zhang, Y.Q.; Peha, J.M. Streaming Video Over the Internet: Approaches and Directions. *IEEE Trans.* **2001**, *11*, 282–300.
79. Pereira, R.; Pereira, E. Video Streaming: Overview and Challenges in the Internet of Things. In *Pervasive Computing*; Intelligent Data-Centric Systems, Academic Press: Boston, MA, USA, 2016; pp. 417–444.
80. RFC 3550—RTP: A Transport Protocol for Real-Time Applications. Available online: https://tools.ietf.org/html/rfc3550 (accessed on 2 October 2021).

81. RFC 2326—Real Time Streaming Protocol (RTSP). Available online: https://tools.ietf.org/html/rfc2326 (accessed on 2 October 2021).
82. Handley, M.; Jacobson, V.; Perkins, C. SDP: Session Description Protocol. 1998. Available online: https://www.hjp.at/doc/rfc/rfc4566.html (accessed on 3 October 2021).
83. Friedman, T.; Caceres, R.; Clark, A. RFC 3611—RTP Control Protocol Extended Reports (RTCP XR). 2003. Available online: https://tools.ietf.org/html/rfc3611 (accessed on 3 October 2021).
84. RFC 7825—A Network Address Translator (NAT) Traversal Mechanism for Media Controlled by the Real-Time Streaming Protocol (RTSP). Available online: https://tools.ietf.org/html/rfc7825 (accessed on 3 October 2021).
85. Kamvar, S.D.; Schlosser, M.T.; Garcia-Molina, H. The Eigentrust Algorithm for Reputation Management in P2P Networks. In Proceedings of the 12th International Conference on World Wide Web, Budapest, Hungary, 20–24 May 2003; pp. 640–651.
86. Camarillo, G. RFC 5694 Peer-to-Peer (P2P) Architecture: Definition, Taxonomies, Examples, and Applicability. Network Workshop Group IETF. 2009. Available online: https://datatracker.ietf.org/doc/rfc5694/ (accessed on 4 October 2021).
87. Ramzan, N.; Park, H.; Izquierdo, E. Video Streaming over P2P Networks: Challenges and Opportunities. *Image Commun.* **2012**, *27*, 401–411. [CrossRef]
88. Chu, Y.h.; Rao, S.G.; Seshan, S.; Zhang, H. A Case for End System Multicast. *IEEE J. Sel. Areas Commun.* **2002**, *20*, 1456–1471. [CrossRef]
89. Gifford, D.; Johnson, K.L.; Kaashoek, M.F.; O'Toole, J.W., Jr. Overcast: Reliable Multicasting with An Overlay Network. In Proceedings of the USENIX Symposium on OSDI, San Diego, CA, USA, 23–25 October 2000.
90. Magharei, N.; Rejaie, R. Prime: Peer-to-Peer Receiver-driven Mesh-based Streaming. *IEEE/ACM Trans. Netw.* **2009**, *17*, 1052–1065. [CrossRef]
91. Pai, V.; Kumar, K.; Tamilmani, K.; Sambamurthy, V.; Mohr, A.E. Chainsaw: Eliminating Trees from Overlay Multicast. In *International Workshop on Peer-to-Peer Systems*; Springer: Berlin/Heidelberg, Germany, 2005; pp. 127–140.
92. Stutzbach, D.; Rejaie, R. Understanding Churn in Peer-to-Peer Networks. In Proceedings of the 6th ACM SIGCOMM Conference on Internet Measurement, New York, NY, USA, 25–27 October 2006; pp. 189–202.
93. Liu, Y.; Guo, Y.; Liang, C. A Survey on Peer-to-Peer Video Streaming Systems. *Peer-to-Peer Netw. Appl.* **2008**, *1*, 18–28. [CrossRef]
94. Sani, Y.; Mauthe, A.; Edwards, C. Adaptive Bitrate Selection: A Survey. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 2985–3014. [CrossRef]
95. Pantos, R. HTTP Live Streaming, 1 May 2009. Internet Engineering Task Force. Available online: https://datatracker.ietf.org/doc/html/draft-pantos-http-live-streaming (accessed on 5 October 2021).
96. Robinson, D. Live Streaming Ecosystems. *Adv. Content Deliv. Stream. Cloud Serv.* **2014**, *2014*, 33–49. [CrossRef]
97. Sodagar, I. The MPEG-DASH Standard for Multimedia Streaming Over the Internet. *IEEE Multimed.* **2011**, *18*, 62–67. [CrossRef]
98. Kua, J.; Armitage, G.; Branch, P. A Survey of Rate Adaptation Techniques for Dynamic Adaptive Streaming Over HTTP. *IEEE Commun. Surv. Tutor.* **2017**, *19*, 1842–1866. [CrossRef]
99. Huang, T.Y.; Johari, R.; McKeown, N.; Trunnell, M.; Watson, M. A Buffer-based Approach To Rate Adaptation: Evidence from a Large Video Streaming Service. In Proceedings of the 2014 ACM Conference on SIGCOMM, Chicago, IL, USA, 17–22 August 2014; pp. 187–198.
100. Spiteri, K.; Urgaonkar, R.; Sitaraman, R.K. BOLA: Near-optimal Bitrate Adaptation for Online Videos. *IEEE/ACM Trans. Netw.* **2020**, *28*, 1698–1711. [CrossRef]
101. Li, Z.; Zhu, X.; Gahm, J.; Pan, R.; Hu, H.; Begen, A.C.; Oran, D. Probe and Adapt: Rate Adaptation for HTTP Video Streaming at Scale. *IEEE J. Sel. Areas Commun.* **2014**, *32*, 719–733. [CrossRef]
102. Jiang, J.; Sekar, V.; Zhang, H. Improving Fairness, Efficiency, and Stability in HTTP-based Adaptive Video Streaming with Festive. In Proceedings of the 8th International Conference on Emerging Networking Experiments and Technologies, Nice, France, 10–13 December 2012; pp. 97–108.
103. Yousef, H.; Feuvre, J.L.; Storelli, A. ABR Prediction Using Supervised Learning Algorithms. In Proceedings of the 2020 IEEE 22nd International Workshop on Multimedia Signal Processing (MMSP), Tampere, Finland, 20–23 September 2020; pp. 1–6.
104. Yin, X.; Jindal, A.; Sekar, V.; Sinopoli, B. A Control-theoretic Approach for Dynamic Adaptive Video Streaming Over HTTP. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, London, UK, 17–21 August 2015; pp. 325–338.
105. Microsoft Silverlight Smooth Streaming. Available online: https://mssilverlight.azurewebsites.net/silverlight/smoothstreaming/ (accessed on 8 October 2021).
106. Live video streaming online | Adobe HTTP Dynamic Streaming. Available online: https://business.adobe.com/ie/products/primetime/adobe-media-server/hds-dynamic-streaming.html (accessed on 8 October 2021).
107. Apple HTTP Live Streaming (HLS), Apple. Available online: https://developer.apple.com/streaming/ (accessed on 8 October 2021).
108. Cermak, G.; Pinson, M.; Wolf, S. The Relationship Among Video Quality, Screen Resolution, and Bit Rate. *IEEE Trans. Broadcast.* **2011**, *57*, 258–262. [CrossRef]
109. Li, Z.; Begen, A.C.; Gahm, J.; Shan, Y.; Osler, B.; Oran, D. Streaming Video Over HTTP with Consistent Quality. In Proceedings of the 5th ACM Multimedia Systems Conference, Singapore, 19–21 March 2014; pp. 248–258.
110. Wang, B.; Kurose, J.; Shenoy, P.; Towsley, D. Multimedia streaming via TCP: An Analytic Performance Study. *ACM Trans. Multimed. Comput. Commun. Appl.* **2008**, *4*, 1–22. [CrossRef]

111. Yu, S.S.; Zhang, J.; Zhou, J.L.; Zhou, X. A Flow Control Scheme in Video Surveillance Applications. *Comput. Eng. Sci.* **2005**, *9*, 3–5.
112. Frnda, J.; Voznak, M.; Sevcik, L. Impact of Packet Loss and Delay Variation on the Quality of Real-time Video Streaming. *Telecommun. Syst.* **2016**, *62*, 265–275. [CrossRef]
113. Seufert, M.; Egger, S.; Slanina, M.; Zinner, T.; Hoßfeld, T.; Tran-Gia, P. A Survey on Quality of Experience of HTTP Adaptive Streaming. *IEEE Commun. Surv. Tutor.* **2015**, *17*, 469–492. [CrossRef]
114. Vega, M.T.; Mocanu, D.C.; Liotta, A. Unsupervised Deep Learning for Real-Time Assessment of Video Streaming Services. *Multimed. Tools Appl.* **2017**, *76*, 22303–22327. [CrossRef]
115. Hinton, G.E. Training Products of Experts by Minimizing Contrastive Divergence. *Neural Comput.* **2002**, *14*, 1771–1800. [CrossRef]
116. de Fréin, R.; Olariu, C.; Song, Y.; Brennan, R.; McDonagh, P.; Hava, A.; Thorpe, C.; Murphy, J.; Murphy, L.; French, P. Integration of QoS Metrics, Rules and Semantic Uplift for Advanced IPTV Monitoring. *J. Netw. Syst. Manag.* **2015**, *23*, 673–708. [CrossRef]
117. Raca, D.; Zahran, A.H.; Sreenan, C.J.; Sinha, R.K.; Halepovic, E.; Jana, R.; Gopalakrishnan, V. On Leveraging Machine and Deep Learning for Throughput Prediction in Cellular Networks: Design, Performance, and Challenges. *IEEE Commun. Mag.* **2020**, *58*, 11–17. [CrossRef]
118. Bentaleb, A.; Timmerer, C.; Begen, A.C.; Zimmermann, R. Bandwidth Prediction in Low-Latency Chunked Streaming. In Proceedings of the 29th ACM Workshop on Network and Operating Systems Support for Digital Audio and Video, Amherst, MA, USA, 21 June 2019; pp. 7–13.
119. Essaili, A.E.; Lohmar, T.; Ibrahim, M. Realization and Evaluation of an End-to-End Low Latency Live DASH System. In Proceedings of the 2018 IEEE International Symposium on Broadband Multimedia Systems and Broadcasting (BMSB), Valencia, Spain, 6–8 June 2018; pp. 1–5.
120. Engel, Y.; Mannor, S.; Meir, R. The Kernel Recursive Least-squares Algorithm. *IEEE Trans. Signal Process.* **2004**, *52*, 2275–2285. [CrossRef]
121. Video Quality of Service (QoS) Tutorial—Cisco. Available online: https://tinyurl.com/vw92pypc (accessed on 10 October 2021).
122. Mao, H.; Netravali, R.; Alizadeh, M. Neural Adaptive Video Streaming with Pensieve. In Proceedings of the Conference of the ACM Special Interest Group on Data Communication, Los Angeles, CA, USA, 21–25 August 2017; pp. 197–210.
123. Mao, H.; Chen, S.; Dimmery, D.; Singh, S.; Blaisdell, D.; Tian, Y.; Alizadeh, M.; Bakshy, E. Real-world Video Adaptation with Reinforcement Learning. *arXiv* **2020**, arXiv:2008.12858.
124. Zhao, Y.; Shen, Q.W.; Li, W.; Xu, T.; Niu, W.H.; Xu, S.R. Latency Aware Adaptive Video Streaming Using Ensemble Deep Reinforcement Learning. In Proceedings of the 27th ACM International Conference on Multimedia, Nice, France, 21–25 October 2019; pp. 2647–2651.
125. Hosmer, D.W., Jr.; Lemeshow, S.; Sturdivant, R.X. *Applied Logistic Regression*; John Wiley & Sons: Hoboken, NJ, USA, 2013; Volume 398.
126. Hastie, T.; Rosset, S.; Zhu, J.; Zou, H. Multi-class Adaboost. *Stat. Its Interface* **2009**, *2*, 349–360. [CrossRef]
127. Friedman, J.H. Stochastic Gradient Boosting. *Comput. Stat. Data Anal.* **2002**, *38*, 367–378. [CrossRef]
128. Rish, I. An Empirical Study of the Naive Bayes Classifier. In Proceedings of the IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence, Seattle, WA, USA, 4–10 August 2001; Volume 3, pp. 41–46.
129. Kataria, A.; Singh, M. A Review of Data Classification Using K-nearest Neighbour Algorithm. *Int. J. Emerg. Technol. Adv. Eng.* **2013**, *3*, 354–360.
130. Sani, Y.; Raca, D.; Quinlan, J.J.; Sreenan, C.J. SMASH: A Supervised Machine Learning Approach to Adaptive Video Streaming over HTTP. In Proceedings of the 2020 Twelfth International Conference on Quality of Multimedia Experience (QoMEX), Athlone, Ireland, 26–28 May 2020; pp. 1–6.
131. Srivastava, S.; Gupta, M.R.; Frigyik, B.A. Bayesian Quadratic Discriminant Analysis. *J. Mach. Learn. Res.* **2007**, *8*, 1277–1305.
132. Piramuthu, S.; Shaw, M.J.; Gentry, J.A. A Classification Approach Using Multi-layered Neural Networks. *Decis. Support Syst.* **1994**, *11*, 509–525. [CrossRef]
133. Mok, R.K.; Chan, E.W.; Chang, R.K. Measuring the Quality of Experience of HTTP Video Streaming. In Proceedings of the 12th IFIP/IEEE International Symposium on Integrated Network Management (IM 2011) and Workshops, Dublin, Ireland, 23–27 May 2011; pp. 485–492.
134. Feamster, N.; Rexford, J. Why (and how) Networks Should Run Themselves. *arXiv* **2017**, arXiv:1710.11583.
135. Naylor, D.; Finamore, A.; Leontiadis, I.; Grunenberger, Y.; Mellia, M.; Munafò, M.; Papagiannaki, K.; Steenkiste, P. The Cost of the "S" in HTTPS. In Proceedings of the 10th ACM International on Conference on Emerging Networking Experiments and Technologies, Sydney, NSW, Australia, 2–5 December 2014; pp. 133–140.
136. Jarmoc, J.; Unit, D. SSL/TLS Interception Proxies and Transitive Trust. Black Hat Europe March 2012. Available online: https://www.semanticscholar.org/paper/SSL%2FTLS-Interception-Proxies-and-Transitive-Trust-Jarmoc/bd1e35fc81e8d3d1751f1d7443fef2dfdbdc2394#citing-papers (accessed on 5 October 2021).
137. Sherry, J.; Lan, C.; Popa, R.A.; Ratnasamy, S. Blindbox: Deep Packet Inspection Over Encrypted Traffic. In Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication, London, UK, 17–21 August 2015; pp. 213–226.
138. Orsolic, I.; Pevec, D.; Suznjevic, M.; Skorin-Kapov, L. YouTube QoE Estimation Based on the Analysis of Encrypted Network Traffic Using Machine Learning. In Proceedings of the 2016 IEEE Globecom Workshops (GC Wkshps), Washington, DC USA, 4–8 December 2016; pp. 1–6.

139. Buddhinath, G.; Derry, D. *A Simple Enhancement To One Rule Classification*; Department Computer Science Software Engeering, University of Melbourne: Melbourne, Australia, 2006; p. 40.
140. Platt, J. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines. 1998. Available online: https://tinyurl.com/9xt2zkaf (accessed on 11 October 2021).
141. Mathuria, M. Decision Tree Analysis on J48 Algorithm for Data Mining. *Int. J. Adv. Res. Comput. Sci. Softw. Eng.* **2013**, *3*, 1114–1119.
142. Dimopoulos, G.; Leontiadis, I.; Barlet-Ros, P.; Papagiannaki, K. Measuring Video QoE from Encrypted Traffic. In Proceedings of the 2016 Internet Measurement Conference, Santa Monica, CA, USA, 14–16 November 2016; pp. 513–526.
143. Casas, P.; Seufert, M.; Wamser, F.; Gardlo, B.; Sackl, A.; Schatz, R. Next to You: Monitoring Quality of Experience in Cellular Networks from the End-devices. *IEEE Trans. Netw. Serv. Manag.* **2016**, *13*, 181–196. [CrossRef]
144. Wassermann, S.; Wehner, N.; Casas, P. Machine Learning Models for YouTube QoE and User Engagement Prediction in Smartphones. *SIGMETRICS* **2019**, *46*, 155–158. [CrossRef]
145. Pal, K.; Patel, B.V. Data Classification with K-fold Cross Validation and Holdout Accuracy Estimation Methods with 5 Different Machine Learning Techniques. In Proceedings of the 2020 Fourth International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 11–13 March 2020; pp. 83–87.
146. Didona, D.; Romano, P. On Bootstrapping Machine Learning Performance Predictors via Analytical Models. *arXiv* **2014**, arXiv:1410.5102.
147. Wassermann, S.; Seufert, M.; Casas, P.; Gang, L.; Li, K. I See What You See: Real Time Prediction of Video Quality from Encrypted Streaming Traffic. In Proceedings of the 4th Internet-QoE Workshop on QoE-based Analysis and Management of Data Communication Networks, Los Cabos, Mexico, 21 October 2019; pp. 1–6.
148. Saeed, U.; Jan, S.U.; Lee, Y.D.; Koo, I. Fault Diagnosis based on Extremely Randomized Trees in Wireless Sensor Networks. *Reliab. Eng. Syst. Saf.* **2021**, *205*, 107284. [CrossRef]
149. Breiman, L. Bagging Predictors. *Mach. Learn.* **1996**, *24*, 123–140. [CrossRef]
150. Chakravarthy, A.D.; Bonthu, S.; Chen, Z.; Zhu, Q. Predictive Models with Resampling: A Comparative Study of Machine Learning Algorithms and their Performances on Handling Imbalanced Datasets. In Proceedings of the 2019 18th IEEE International Conference On Machine Learning Additionally, Applications (ICMLA), Boca Raton, FL, USA, 16–19 December 2019; pp. 1492–1495.
151. Wassermann, S.; Seufert, M.; Casas, P.; Gang, L.; Li, K. Let Me Decrypt Your Beauty: Real-time Prediction of Video Resolution and Bitrate for Encrypted Video Streaming. In Proceedings of the 2019 Network Traffic Measurement and Analysis Conference (TMA), Paris, France, 19–21 June 2019; pp. 199–200.
152. Gutterman, C.; Guo, K.; Arora, S.; Wang, X.; Wu, L.; Katz-Bassett, E.; Zussman, G. Requet: Real-Time QoE Detection for Encrypted YouTube Traffic. In Proceedings of the 10th ACM Multimedia Systems Conference, Amherst, MA, USA, 18–21 June 2019; pp. 48–59.
153. Gutterman, C.; Guo, K.; Arora, S.; Gilliland, T.; Wang, X.; Wu, L.; Katz-Bassett, E.; Zussman, G. Requet: Real-Time QoE Metric Detection for Encrypted YouTube Traffic. *ACM Trans. Multimed. Comput. Commun. Appl.* **2020**, *16*, 1–28. [CrossRef]
154. Seufert, M.; Casas, P.; Wehner, N.; Gang, L.; Li, K. Stream-based Machine Learning for Real-time QoE analysis of Encrypted Video Streaming Traffic. In Proceedings of the 2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN), Paris, France, 19–21 February 2019; pp. 76–81.
155. Seufert, M.; Casas, P.; Wehner, N.; Gang, L.; Li, K. Features That Matter: Feature Selection for On-line Stalling Prediction in Encrypted Video Streaming. In Proceedings of the IEEE INFOCOM 2019-IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS), Paris, France, 29 April–2 May 2019; pp. 688–695.
156. Krishnamoorthi, V.; Carlsson, N.; Halepovic, E.; Petajan, E. BUFFEST: Predicting Buffer Conditions and Real-Time Requirements of HTTP(S) Adaptive Streaming Clients. In Proceedings of the 8th ACM on Multimedia Systems Conference, Taipei, Taiwan, 20–23 June 2017; pp. 76–87.
157. Mazhar, M.H.; Shafiq, Z. Real-time Video Quality of Experience Monitoring for HTTPS and QUIC. In Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, HI, USA, 15–19 April 2018; pp. 1331–1339.
158. Bronzino, F.; Schmitt, P.; Ayoubi, S.; Martins, G.; Teixeira, R.; Feamster, N. Inferring Streaming Video Quality from Encrypted Traffic: Practical Models and Deployment Experience. *Proc. Acm Meas. Anal. Comput. Syst.* **2019**, *3*, 1–25. [CrossRef]
159. Pandey, S.; Choi, M.J.; Yoo, J.H.; Hong, J.W.K. Streaming Pattern Based Feature Extraction for Training Neural Network Classifier to Predict Quality of VOD services. In Proceedings of the 2021 IFIP/IEEE International Symposium on Integrated Network Management (IM), Bordeaux, France, 17–21 May 2021; pp. 551–557.
160. Schwarzmann, S.; Cassales Marquezan, C.; Bosk, M.; Liu, H.; Trivisonno, R.; Zinner, T. Estimating Video Streaming QoE in the 5G Architecture Using Machine Learning. In Proceedings of the 4th Internet-QoE Workshop on QoE-Based Analysis and Management of Data Communication Networks, Los Angeles, CA, USA, 21 August 2019; pp. 7–12.
161. Baraković, S.; Skorin-Kapov, L. Survey and Challenges of QoE Management Issues in Wireless Networks. *J. Comput. Netw. Commun.* **2013**, *2013*. [CrossRef]
162. Bartolec, I.; Orsolic, I.; Skorin-Kapov, L. In-network YouTube Performance Estimation in Light of End User Playback-Related Interactions. In Proceedings of the 2019 Eleventh International Conference on Quality of Multimedia Experience (QoMEX), Berlin, Germany, 5–7 June 2019; pp. 1–3.

163. Orsolic, I.; Suznjevic, M.; Skorin-Kapov, L. Youtube QoE Estimation From Encrypted Traffic: Comparison of Test Methodologies and Machine Learning Based Models. In Proceedings of the 2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX), Sardinia, Italy, 29–31 May 2018; pp. 1–6.
164. Oršolić, I.; Rebernjak, P.; Sužnjević, M.; Skorin-Kapov, L. In-network QoE and KPI Monitoring of Mobile YouTube Traffic: Insights for encrypted ios flows. In Proceedings of the 2018 14th International Conference on Network and Service Management (CNSM), Rome, Italy, 5–9 November 2018; pp. 233–239.
165. Sun, Y.; Yin, X.; Jiang, J.; Sekar, V.; Lin, F.; Wang, N.; Liu, T.; Sinopoli, B. CS2P: Improving Video Bitrate Selection and Adaptation with Data-Driven Throughput Prediction. In Proceedings of the 2016 ACM SIGCOMM Conference, Florianopolis, Brazil, 22–26 August 2016; pp. 272–285.
166. Claeys, M.; Latré, S.; Famaey, J.; De Turck, F. Design and Evaluation of a Self-Learning HTTP Adaptive Video Streaming Client. *IEEE Commun. Lett.* **2014**, *18*, 716–719. [CrossRef]
167. de Fréin, R. Take off a load: Load-Adjusted Video Quality Prediction and Measurement. In Proceedings of the 2015 IEEE International Conference on Computer and Information Technology; Ubiquitous Computing and Communications; Dependable, Autonomic and Secure Computing; Pervasive Intelligence and Computing, Liverpool, UK, 26 October 2015; pp. 1886–1894. [CrossRef]
168. Rossi, L.; Chakareski, J.; Frossard, P.; Colonnese, S. A Poisson Hidden Markov Model for Multiview Video Traffic. *IEEE/ACM Trans. Netw.* **2014**, *23*, 547–558. [CrossRef]
169. Bampis, C.G.; Bovik, A.C. Feature-based Prediction of Streaming Video QoE: Distortions, Stalling and Memory. *Signal Process. Image Commun.* **2018**, *68*, 218–228. [CrossRef]
170. Tran, H.T.T.; Nguyen, D.V.; Ngoc, N.P.; Thang, T.C. Overall Quality Prediction for HTTP Adaptive Streaming Using LSTM Network. *IEEE Trans. Circuits Syst. Video Technol.* **2021**, *31*, 3212–3226. [CrossRef]
171. Wang, Y.; Jiang, L.; Yang, M.H.; Li, L.J.; Long, M.; Fei-Fei, L. Eidetic 3d LSTM: A Model for Video Prediction and Beyond. In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
172. Dinaki, H.E.; Shirmohammadi, S.; Janulewicz, E.; Côté, D. Forecasting Video QoE With Deep Learning From Multivariate Time-Series. *IEEE Open J. Signal Process.* **2021**, *2*, 512–521. [CrossRef]
173. Kirkpatrick, K. Software-Defined Networking. *Commun. ACM* **2013**, *56*, 16–19. [CrossRef]
174. Xie, J.; Yu, F.R.; Huang, T.; Xie, R.; Liu, J.; Wang, C.; Liu, Y. A Survey of Machine Learning Techniques Applied to Software Defined Networking (SDN): Research Issues and Challenges. *IEEE Commun. Surv. Tutor.* **2019**, *21*, 393–430. [CrossRef]
175. Carner, J.; Mestres, A.; Alarcón, E.; Cabellos, A. Machine Learning-based Network Modeling: An Artificial Neural Network Model vs. a Theoretical Inspired Model. In Proceedings of the 2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN), Milan, Italy, 4–7 July 2017; pp. 522–524.
176. Jain, S.; Khandelwal, M.; Katkar, A.; Nygate, J. Applying Big Data Technologies to Manage QoS in an SDN. In Proceedings of the 2016 12th International Conference on Network and Service Management (CNSM), Montreal, QC, Canada, 3 October–4 November 2016; pp. 302–306.
177. Malik, A.; de Fréin, R.; Aziz, B. Rapid Restoration Techniques for Software-Defined Networks. *Appl. Sci.* **2020**, *10*, 3411. [CrossRef]
178. Pasquini, R.; Stadler, R. Learning End-to-End Application QoS from Openflow Switch Statistics. In Proceedings of the 2017 IEEE Conference on Network Softwarization (NetSoft), Bologna, Italy, 3–7 July 2017; pp. 1–9.
179. Ben Letaifa, A. Adaptive QoE Monitoring Architecture in SDN Networks: Video Streaming Services Case. In Proceedings of the 2017 13th International Wireless Communications and Mobile Computing Conference (IWCMC), Valencia, Spain, 26–30 June 2017; pp. 1383–1388.
180. Petrangeli, S.; Wu, T.; Wauters, T.; Huysegems, R.; Bostoen, T.; De Turck, F. A Machine Learning-Based Framework for Preventing Video Freezes in HTTP Adaptive Streaming. *J. Netw. Comput. Appl.* **2017**, *94*, 78–92. [CrossRef]
181. Seiffert, C.; Khoshgoftaar, T.M.; Van Hulse, J.; Napolitano, A. RUSBoost: A Hybrid Approach to Alleviating Class Imbalance. *IEEE Trans. Syst. Man Cybern. Part A Syst. Humans* **2009**, *40*, 185–197. [CrossRef]
182. Da Hora, D.; Van Doorselaer, K.; Van Oost, K.; Teixeira, R. Predicting the Effect of Home Wi-Fi Quality on QoE. In Proceedings of the IEEE INFOCOM 2018-IEEE Conference on Computer Communications, Honolulu, HI, USA, 15–19 April 2018; pp. 944–952.
183. Wamser, F.; Casas, P.; Seufert, M.; Moldovan, C.; Tran-Gia, P.; Hossfeld, T. Modeling the YouTube stack: From Packets to Quality of Experience. *Comput. Netw.* **2016**, *109*, 211–224. [CrossRef]
184. Zinner, T.; Hohlfeld, O.; Abboud, O.; Hoßfeld, T. Impact of Frame Rate and Resolution on Objective QoE Metrics. In Proceedings of the 2010 Second International Workshop on Quality of Multimedia Experience (QoMEX), Trondheim, Norway, 21–23 June 2010; pp. 29–34.
185. Ligata, A.; Perenda, E.; Gacanin, H. Quality of Experience Inference for Video Services in Home WiFi Networks. *IEEE Commun. Mag.* **2018**, *56*, 187–193. [CrossRef]
186. Bhattacharyya, R.; Xia, B.; Rengarajan, D.; Shakkottai, S.; Kalathil, D. Flowbazaar: A Market-Mediated Software Defined Communications Ecosystem at the Wireless Edge. *arXiv* **2018**, arXiv:1801.00825.
187. Ammar, D.; De Moor, K.; Skorin-Kapov, L.; Fiedler, M.; Heegaard, P.E. Exploring the Usefulness of Machine Learning in the Context of WebRTC Performance Estimation. In Proceedings of the 2019 IEEE 44th Conference on Local Computer Networks (LCN), Osnabruck, Germany, 14–17 October 2019; pp. 406–413.

188. Yan, S.; Guo, Y.; Chen, Y.; Xie, F. Predicting Freezing of WebRTC Videos in WiFi Networks. In *International Conference on Ad Hoc Networks*; Springer: Berlin/Heidelberg, Germany, 2018; pp. 292–301.
189. Reiter, U.; Brunnström, K.; De Moor, K.; Larabi, M.C.; Pereira, M.; Pinheiro, A.; You, J.; Zgank, A. Factors Influencing Quality of Experience. In *QoE*; Springer: Berlin/Heidelberg, Germany, 2014; pp. 55–72.
190. de Fréin, R. State Acquisition in Computer Networks. In Proceedings of the 2018 IFIP Networking Conference (IFIP Networking) and Workshops, Zurich, Switzerland, 14–16 May 2018; pp. 1–9. [CrossRef]
191. Suto, J.; Oniga, S.; Lung, C.; Orha, I. Comparison of Offline and Real-time Human Activity Recognition Results Using Machine Learning Techniques. *Neural Comput. Appl.* **2020**, *32*, 15673–15686. [CrossRef]
192. Ayodele, T.O. Types of Machine Learning Algorithms. *New Adv. Mach. Learn.* **2010**, *3*, 19–48.
193. Malik, A.; de Fréin, R. A Proactive-Restoration Technique for SDNs. In Proceedings of the 2020 IEEE Symposium on Computers and Communications (ISCC), Rennes, France, 7–10 July 2020.
194. Siami-Namini, S.; Tavakoli, N.; Namin, A.S. The Performance of LSTM and BiLSTM in Forecasting Time Series. In Proceedings of the 2019 IEEE International Conference on Big Data (Big Data), Los Angeles, CA, USA, 9–12 December 2019; pp. 3285–3292.
195. Längkvist, M.; Karlsson, L.; Loutfi, A. A Review of Unsupervised Feature Learning and Deep Learning for Time-series Modeling. *Pattern Recognit. Lett.* **2014**, *42*, 11–24. [CrossRef]
196. Eswara, N.; Ashique, S.; Panchbhai, A.; Chakraborty, S.; Sethuram, H.P.; Kuchi, K.; Kumar, A.; Channappayya, S.S. Streaming Video QoE Modeling and Prediction: A Long Short-Term Memory Approach. *IEEE Trans. Circuits Syst. Video Technol.* **2019**, *30*, 661–673. [CrossRef]
197. White, G.; Palade, A.; Clarke, S. Forecasting QoS Attributes Using LSTM Networks. In Proceedings of the 2018 International Joint Conference on Neural Networks (IJCNN), Rio de Janeiro, Brazil, 8–13 July 2018; pp. 1–8.
198. Zhang, Z.; Cui, P.; Zhu, W. Deep Learning on Graphs: A Survey. *IEEE Trans. Knowl. Data Eng.* **2020**, *1*, 5555. [CrossRef]
199. Russakovsky, O.; Deng, J.; Su, H.; Krause, J.; Satheesh, S.; Ma, S.; Huang, Z.; Karpathy, A.; Khosla, A.; Bernstein, M.; et al. Imagenet Large Scale Visual Recognition Challenge. *Int. J. Comput. Vis.* **2015**, *115*, 211–252. [CrossRef]
200. Çetinkaya, E.; Amirpour, H.; Timmerer, C.; Ghanbari, M. FaME-ML: Fast Multirate Encoding for HTTP Adaptive Streaming Using Machine Learning. In Proceedings of the 2020 IEEE International Conference on Visual Communications and Image Processing (VCIP), Munich, Germany, 5–8 December 2020; pp. 87–90.
201. Yeo, H.; Do, S.; Han, D. How Will Deep Learning Change Internet Video Delivery? In Proceedings of the 16th ACM Workshop on Hot Topics in Networks, Palo Alto, CA, USA, 30 November–1 December 2017; pp. 57–64.
202. Jameel, S.M.; Hashmani, M.A.; Rehman, M.; Budiman, A. An Adaptive Deep Learning Framework for Dynamic Image Classification in the Internet of Things Environment. *Sensors* **2020**, *20*, 5811. [CrossRef]
203. Chai, F.; Kang, K.D. Adaptive Deep Learning for Soft Real-Time Image Classification. *Technologies* **2021**, *9*, 20. [CrossRef]
204. Gomes, H.M.; Bifet, A.; Read, J.; Barddal, J.P.; Enembreck, F.; Pfharinger, B.; Holmes, G.; Abdessalem, T. Adaptive Random Forests for Evolving Data Stream Classification. *Mach. Learn.* **2017**, *106*, 1469–1495. [CrossRef]
205. Bibal, A.; Frénay, B. Interpretability of Machine Learning Models and Representations: An Introduction. In Proceedings of the ESANN 2016 Proceedings, European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning, Bruges, Belgium, 27–29 April 2016.
206. Yin, M.; Wortman Vaughan, J.; Wallach, H. Understanding the Effect of Accuracy on Trust in Machine Learning Models. In Proceedings of the 2019 Chi Conference on Human Factors in Computing Systems, Glasgow, UK, 4–9 May 2019; pp. 1–12.
207. Carvalho, D.V.; Pereira, E.M.; Cardoso, J.S. Machine Learning interpretability: A Survey on Methods and Metrics. *Electronics* **2019**, *8*, 832. [CrossRef]
208. Ribeiro, M.T.; Singh, S.; Guestrin, C. Model-Agnostic Interpretability of Machine Learning. *arXiv* **2016**, arXiv:1606.05386.
209. Malik, A.; de Fréin, R.; Al-Zeyadi, M.; Andreu-Perez, J. Intelligent SDN Traffic Classification Using Deep Learning: Deep-SDN. In Proceedings of the 2020 2nd International Conference on Computer Communication and the Internet (ICCCI), Nagoya, Japan, 26–29 June 2020; pp. 184–189.
210. Greenwald, H.S.; Oertel, C.K. Future Directions in Machine Learning. *Front. Robot. AI* **2017**, *3*, 79. [CrossRef]
211. Malik, A.; Aziz, B.; Adda, M.; Ke, C.H. Smart Routing: Towards Proactive Fault Handling of Software-Defined Networks. *Comput. Netw.* **2020**, *170*, 107104. [CrossRef]
212. Orsolic, I.; Skorin-Kapov, L. A Framework for in-Network QoE Monitoring of Encrypted Video Streaming. *IEEE Access* **2020**, *8*, 74691–74706. [CrossRef]
213. Wares, S.; Isaacs, J.; Elyan, E. Data Stream Mining: Methods and Challenges for Handling Concept Drift. *SN Appl. Sci.* **2019**, *1*, 1–19. [CrossRef]
214. Yoon, J.; James, J.; Van Der Schaar, M. Missing Data Imputation Using Generative Adversarial Nets. In Proceedings of the 35th International Conference on Machine Learning, Stockholm, Sweden, 10–15 July 2018.
215. Gondara, L.; Wang, K. Multiple imputation using deep denoising autoencoders. *arXiv* **2017**, arXiv:1705.02737.
216. Frey, B.J.; Brendan, J.F.; Frey, B.J. *Graphical Models for Machine Learning and Digital Communication*; MIT Press: Cambridge, MA, USA, 1998.
217. You, J.; Ma, X.; Ding, D.Y.; Kochenderfer, M.; Leskovec, J. Handling Missing Data with Graph Representation Learning. *arXiv* **2020**, arXiv:2010.16418.

218. de Fréin, R. Load-adjusted video quality prediction methods for missing data. In Proceedings of the 2015 10th International Conference for Internet Technology and Secured Transactions (ICITST), London, UK, 14–16 December 2015; pp. 314–319.
219. Chen, Y.; Wu, K.; Zhang, Q. From QoS to QoE: A Tutorial on Video Quality Assessment. *IEEE Commun. Surv. Tutor.* **2014**, *17*, 1126–1165. [CrossRef]
220. Part 3: How to Compete With Broadcast Latency Using Current Adaptive Bitrate Technologies | AWS Media Blog. Available online: https://tinyurl.com/8wmrv5cr (accessed on 12 October 2021).
221. Balachandran, A.; Sekar, V.; Akella, A.; Seshan, S.; Stoica, I.; Zhang, H. Developing a Predictive Model of Quality of Experience for Internet Video. *ACM SIGCOMM Comput. Commun. Rev.* **2013**, *43*, 339–350. [CrossRef]