Demystifying the QoS and QoE of Edge-hosted Video Streaming Applications in the Wild with SNESet

YANAN LI, Alibaba Group & State Key Laboratory of Networking and Switching Technology, China
GUANGQING DENG*, Alibaba Group, China
CHANGMING BAI, Alibaba Group, China
JINGYU YANG, Alibaba Group, China
GANG WANG, Alibaba Group, China
HAO ZHANG, Alibaba Group, China
JIN BAI, Alibaba Group, China
HAITAO YUAN*, Nanyang Technological University, Singapore
MENGWEI XU, State Key Laboratory of Networking and Switching Technology, China
SHANGGUANG WANG, State Key Laboratory of Networking and Switching Technology, China

Video streaming applications (VSAs) are increasingly being deployed on large-scale edge platforms, which have the potential to significantly improve the quality of service (QoS) and end-user experience (QoE), ultimately maximizing business outcomes. However, there is currently very little understanding of how QoS, QoE, and the impact of QoS on QoE for VSAs on edge platforms in the wild and at scale. To close the knowledge gap, we collect SNESet, an active measurement dataset comprising QoS and QoE telemetry metrics of 8 VSAs over four months, covering end-users from 798 edge sites, 30 cities, and 3 ISPs in one country. We characterize and compare the QoS and QoE metrics in SNESet with existing publicly available datasets, highlighting that SNESet includes a significantly greater number of metrics (horizontal diversity and vertical hierarchy) and provides more comprehensive coverage of specific metrics. Moreover, we *qualitatively* and *quantitatively* analyze the impact of QoS on QoE in both domain-general and domain-specific scenarios. Our findings can inform the system design decisions that different entities in the video ecosystem (content providers, video player designers, third-party optimizers, edge vendors) make to maximize end-users experience and ultimately maximize the business outcomes. We hope SNESet can attract more research efforts in the data management community, computer network community, and beyond.

CCS Concepts: • **Networks** → **Network measurement**; *Network performance analysis*; • **Computer systems organization** → Client-server architectures.

Additional Key Words and Phrases: QoE metrics, QoS metrics, edge computing, network measurements

*Haitao Yuan (yht19@tsinghua.org.cn) and Guangqing Deng (dgq124221@alibaba-inc.com) are the corresponding authors.

Authors' addresses: Yanan Li, Alibaba Group & State Key Laboratory of Networking and Switching Technology, China, YaNanLi@bupt.edu.cn; Guangqing Deng, Alibaba Group, China, dgq124221@alibaba-inc.com; Changming Bai, Alibaba Group, China, changming.bcm@alibaba-inc.com; Jingyu Yang, Alibaba Group, China, yangjingyu@alibaba-inc.com; Gang Wang, Alibaba Group, China, cangqu.wg@alibaba-inc.com; Hao Zhang, Alibaba Group, China, guxun.zh@alibaba-inc.com; Jin Bai, Alibaba Group, China, pishi.bj@alibaba-inc.com; Haitao Yuan, Nanyang Technological University, Singapore, yht19@tsinghua.org.cn; Mengwei Xu, State Key Laboratory of Networking and Switching Technology, China, mwx@bupt.edu.cn; Shangguang Wang, State Key Laboratory of Networking and Switching Technology, China, sgwang@bupt.edu.cn.

2836-6573/2023/12-ART236 $15.00
https://doi.org/10.1145/3626723

ACM Reference Format:
Yanan Li, Guangqing Deng, Changming Bai, Jingyu Yang, Gang Wang, Hao Zhang, Jin Bai, Haitao Yuan, Mengwei Xu, and Shangguang Wang. 2023. Demystifying the QoS and QoE of Edge-hosted Video Streaming Applications in the Wild with SNESet. *Proc. ACM Manag. Data* 1, 4 (SIGMOD), Article 236 (December 2023), 29 pages. https://doi.org/10.1145/3626723

1 INTRODUCTION

Video streaming applications (VSAs) such as TikTok [99], Zoom [125], and YouTube Live [67] have experienced unprecedented popularity during the post-COVID-19 pandemic period. According to [15], video streaming traffic increased by as much as three-fold during an eight-month period in 2020. Moreover, with ever-stringent user performance expectations (low latency and high bandwidth), traditional centralized cloud are struggling to provide a one-size-fits-all solution, leading more video content providers to deploy their services in a decentralized manner on large-scale public edge platforms like Azure Edge Zone [123] and AWS Local Zones [124].

Ideally, the adoption of public edge platforms has the potential to improve the overall quality of service (QoS) for VSAs, which can ultimately affect the end-user experience (QoE), and impact business outcomes such as viewership and revenues. However, the lack of large-scale, realistic measurements, analyses, and publicly available datasets has limited exploration of QoS, QoE, and the impact of QoS on QoE for VSAs on edge platforms in the wild. QoS and QoE are crucial metrics for VSAs, with QoS referring to the service quality of each component within end-to-end video delivery system, including content origin sites, edge platforms, and ISPs involved in last-mile delivery to end-users [3]. The International Telecommunication Union (ITU) defines QoE as the measure of user satisfaction using the Mean Opinion Score (MOS), a subjective metric ranging from 1 to 5 [50]. However, subjective metrics can be expensive and complicated to collect [23]. In this study, we focus primarily on objective QoE metrics that are easily measurable and comparable, such as the stall ratio, which is widely used to summarize VSA performance [60, 117].

Table 1. Comparison with existing publicly available cloud/edge datasets.

Dataset	QoS (server-side)	QoE	Platform	Year	Duration	# of Sites	# of Network Protocol Stacks
Alibaba Dataset [38]	✓	✗	Alibaba ECS	2018	8 days	1	1
Google Dataset [100]	✓	✗	Google Borg	2019	1 month	1	1
Azure Dataset [27]	✓	✗	Azure Cloud	2019	30 days	1	1
Edge Dataset [112]	✓	✗	Alibaba ENS	2020	3 months	139	2
LiSSi lab Dataset [7]	✗	✓	Testbed	2015	1 week	1	1
Puffer [114]	✗	✓	Testbed*	Since Jan 2019	Keep updating	1	2
Huawei Dataset [102]	✓	✓	Testbed	2018	1 month	1	3
SNESet (Ours)	✓	✓	NEP	2022	4 months	789	4

* While we classify the Puffer [114] as a testbed, it falls somewhere between a traditional testbed and a real-world commercial system.

While several publicly available datasets provide QoS and QoE metrics, only a few of them include both metrics and are collected from large-scale edge platforms in the wild. As is shown in Table 1, most open-source datasets collected from traditional centralized clouds primarily focus on sever-side QoS metrics and provide limited client-side QoE metrics (e.g., Alibaba Dataset [38], Azure Dataset [27], and Google Dataset [100]). Existing open-source datasets for VSAs mainly focus on client-side QoE metrics, with little inclusion of server-side QoS metrics such as the inner network lossrate and average first-byte time. Most importantly, few of these datasets fully characterize the QoS and QoE of VSAs on large-scale edge platforms in the wild, as most are collected from simulated testbeds that may not accurately represent real-world scenarios. Although some prior works have implemented passive measurement in the wild [18, 74], their methodologies are designed for specific applications and may not generalize to others.

To close the knowledge gap, we share our experiences in running a leading public edge platform, NEP (**N**ext-generation **E**dge **P**latform[1]), which comprises 2800+ edge sites globally, with a peak query rate of more than 100 million per second (§ 2.1). Leveraging the NEP, we have conducted a large-scale, in-depth study and collected SNESet, an active measurement of 8 VSAs spanning four months in 2022, covering end-users from 798 edge sites, 30 cities, and 3 ISPs in one country (§ 2.2). To the best of our knowledge, SNESet is the first publicly available dataset [2] for VSAs on large-scale edge platforms, which includes a significantly greater number of metrics and more comprehensive coverage of specific metrics (§ 2.3). Moreover, we also include data-centric problems and challenges from the perspective of data collection (§ 2.4), data processing (§ 3.4), and data management (§ 5).

We first characterize and compare the QoS and QoE metrics in SNESet with existing publicly available datasets, which enabling us to gain valuable experiences and insights of VSAs' performance on large-scale edge platform. Then, we *qualitatively* investigate the impact of QoS on QoE using Kendall correlation and relative information gain (§ 3). The Kendall correlation is useful for investigating the interaction between variables when the relationship is roughly monotone. The information gain can corroborate or augment the correlation when the relationship is not monotone and extends to the multivariate case. Our main observations are as follows:

- Given the intricate nature of the underlying network and the variability in traffic volume, the average stall ratio observed in SNESet is about 4× higher than that of the simulated testbed dataset. Moreover, the stall ratio exhibits significant spatio-temporal variations across applications, cities, and ISPs, with maximum differences reaching up to 3× (§ 3.1).
- SNESet collects a wider range of QoS and QoE metrics, encompassing various network protocol stacks. Most importantly, it includes a greater number of server-side inner network metrics, such as average first byte time, inner network RTT, and inner network packet loss rate. Moreover, SNESet offers a more comprehensive coverage of specific metrics (§ 3.1 - § 3.2).
- Regarding the Kendall correlation, the results indicate that CPU utilization and the day of the month have the strongest monotonic relationships with stall ratio, with values of 0.83 and 0.82, respectively. The former can be attributed to the fact that both metrics have similar peak values during the same hour of the day and the latter is due to ISPs intentionally throttling bandwidth for end-users at the end of the month (§ 3.3).
- Regarding relative information gain, we observed that network-related metrics have higher values, while other QoS metrics are mostly around 1% (§ 3.3).

QoE metrics are commonly located in the upper application layer, while QoS metrics typically reside in the lower system layer. Directly obtaining QoE metrics can be challenging due to permissions or framework limitations. Consequently, QoS-based prediction serves as a non-intrusive approach to assess QoE. To further *quantitatively* analyze the impact of different QoS metrics, we employ seven mainstream regression methods (§ 4). Considering the timeliness of real-world deployment, we benchmark the prediction accuracy and the time efficiency in both domain-general (for all the applications) and domain-specific scenarios (for specific applications). We find that the most influential features differ significantly between these two scenarios. Specifically, `domain_name`, `isp`, and `reset_ratio` are the most influential feature in domain-general scenarios, while in domain-specific scenarios, features such as `date`, `hour`, and I/O-related features take precedence.

Qualitative analytics can be seen as a preprocessing stage for quantitative analytics, which aims to reduce the feature complexity (feature size and sample size). By selecting the most relevant features and reducing interference (irrelevant) features, it helps to improve not only the interpretability but

[1]NEP is commercially known as Alibaba ENS [6].

[2]We publish our code and SNESet on https://github.com/YananLi18/SNESet.

also the robustness of the quantitative model. It's worth noting that the combination of qualitative analytics and quantitative analytics extends beyond the scope of demystifying the QoS and QoE of VSAs and can serve as a general mechanism to study similar topics. For instance, adaptive bitrate selection (ABR) [96, 114, 117] requires studying the impact of different factors on network throughput or transmission time to determine the optimal chunk bitrate.

Finally, we point out future research directions with promising research opportunities (§ 5). In particular, with the increasing diversity of underlying data types (structured data, semi-structured, and graph data) and concurrent upper-layer query requests (batch processing, OLTP, and OLAP), a cloud-edge collaborative distributed data management system might be a promising alternative for these large-scale (hundreds and thousands of) WAN-connected edge sites rather than the current centralized approach. We hope our work can attract more research efforts in these directions and be used to inform the system design decisions of the data management community, computer network community, and beyond.

In summary, we make the following contributions in this paper:

- We have collected `SNESet`, which contains both QoS and QoE metrics for eight VSAs on a large-scale edge platform using active measurements. Additionally, to facilitate future research studies, we make the raw data of `SNESet` and our code publicly available.
- We characterize and compare the QoS and QoE metrics in `SNESet` with existing publicly available datasets, highlighting that `SNESet` includes a significantly greater number of metrics and more comprehensive coverage of specific metrics (vertical hierarchy and horizontal diversity).
- We *qualitatively* and *quantitatively* analyze the impact of QoS on QoE in both domain-general and domain-specific scenarios, which can be used to inform the system design decisions.

2 DATASET

In this section, we first introduce the public edge platform, NEP, which serves as the foundation platform for our data collection. Next, we present the specific system architecture of our data collection process and highlight the unique properties of our dataset, `SNESet`, in comparison with existing open-source datasets.

2.1 Platform Description

NEP is an emerging leading-edge platform in its home country. Unlike traditional centralized clouds with a limited number of sites within a single country, NEP boasts significantly more sites, approximately two orders of magnitude larger, and is continuously expanding. This difference in scale profoundly influences various aspects of the platform, including application performance and resource utilization, which we will delve into in subsequent sections.

Despite the increasing number of edge sites, the capacity of each NEP site remains relatively limited compared to traditional centralized cloud platforms. Traditional platforms can host thousands or even millions of physical servers [59], whereas NEP edge sites are constrained by physical infrastructure such as limited space and electricity, typically accommodating only tens or hundreds of servers. The physical servers of NEP are obtained from various locations, including CDN PoPs, third-party IDCs, and network operators. Currently, NEP primarily relies on micro data centers and has not been widely implemented in cellular core networks, as originally envisioned by MECs [44].

In contrast to CDN platforms, NEP currently provides a comprehensive range of services, which includes not only content caching, but also edge containers, virtual machine instances, bare metal machines, and heterogeneous computing (e.g., ARM SoC [121], FPGA [120]). While NEP supports many types of services, the current dominant usage among video content providers is Platform-as-a-Service (PaaS).

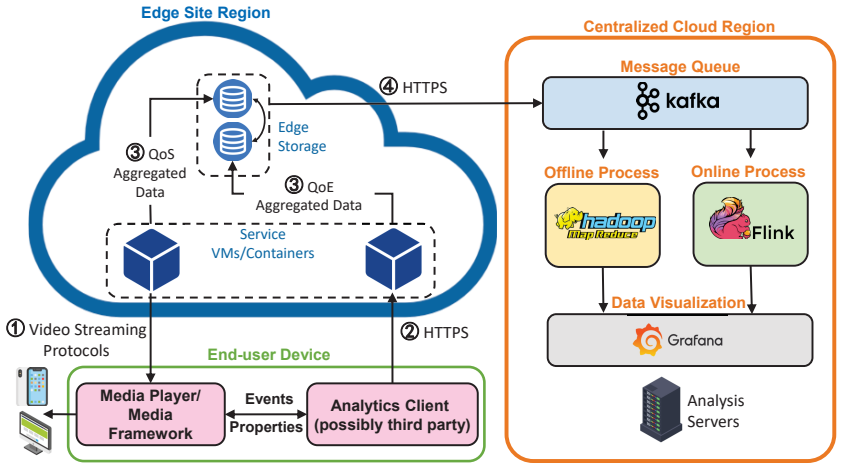


Fig. 1. Overall architecture of data collection and analytics.

2.2 System Architecture of Data Collection

Although NEP offers a wide range of services, VSAs currently exhibit the highest network throughput growth rate, with a median queries-per-second (QPS) per edge site exceeding 20,000 in a single day. Thus, it is imperative to quantify the QoS and QoE of VSAs to enhance the core competitiveness of the edge platform. As depicted in Figure 1, we have designed a measurement system that aims to optimize the overall performance of the edge platform through a data-driven approach. The measurement system comprises three components, located at end-user devices, edge sites, and the centralized cloud, respectively.

The Media Player on end-users' devices utilizes various protocols such as HTTPS [87], RTMP [83], and WebRTC [16] to download video chunks from nearby edge sites. The Analytics Client collects various events (e.g., playbackPause, playbackStall) and properties (e.g., video codec, resolution) from the Media Player and transmits the necessary metrics to an analytics server in the same edge sites. Currently, the Analytics Client does not consider the users' own processing and rendering latency, which is related to the users' devices, hardware configuration, and the number of concurrent applications. The above metrics collected by the Analytics Client are still session-level metrics. To robustly assess group-level QoE performance, the analytics server aggregates these metrics from end-users at the same edge site within a coarser time granularity (e.g., 5 minutes). The aggregation encompass various forms, including maximum, median, average, and 95th percentile, etc. Unless explicitly stated otherwise, the metrics of `SNESet` utilize the average aggregation by default. With the egress network bandwidth of the edge sites typically being 100Gbps and VSA requests generally requiring 1Mbps-2Mbps bandwidth, the analytics server of each edge site aggregates across approximately 50K to 100K end-users.

In each edge site, we also employ event tracking techniques to collect performance metrics of various software, such as reverse proxy software-Tegine [98]. Moreover, Linux `tsar` [58] is used to collect minute-level hardware metrics (CPU, memory, and storage) and network metrics. The QoS telemetry data are then stored in QoS edge storage. The QoS and QoE data stored at each edge site are merged and transmitted to the central cloud via HTTPS, and then fed into Apache Kafka [36] message queues. Then offline and online analysis is performed using Hadoop-based [78] and

Table 2. The detailed schema of SNESet. The network protocol stack is classified into the Application Layer (5 for short), Transport Layer (4), Network Layer (3), Link Layer (2), and Physical Layer (1).

Metrics	Taxonomy	Type	Net Stack	Description	Example
date_sequence	Context	BIGINT	-	The day of the month, which ranges from [1, 31].	21
hour_sequence	Context	BIGINT	-	The hour of the day, which ranges from [0, 23].	5
domain_name	Context	STRING	-	The domain name identifier, which represents different applications.	domain_18
city	Context	STRING	-	The city identifier.	city_14
isp	Context	STRING	-	The ISP identifier.	isp_10
node_name	Context	STRING	-	The edge site identifier.	node_12
avg_fbt_time	QoS (server-side)*	BIGINT	5	The average first-byte time represents the time from the edge site receiving the request to sending the first response packet (unit ms).	12
inner_net_droprate	QoS (server-side)	DOUBLE	3	Edge site internal network packet loss rate (%)	0.001
inner_net_rtt	QoS (server-side)	BIGINT	3	Edge site internal network round trip time (ms)	2
cpu_util	QoS (server-side)	DOUBLE	1	Edge site average CPU utilization	0.56
mem_util	QoS (server-side)	DOUBLE	1	Edge site average memory utilization	0.62
io_await_avg	QoS (server-side)	BIGINT	1	Edge site average I/O waiting time	31
io_await_max	QoS (server-side)	BIGINT	1	Edge site maximum I/O waiting time	34
io_util_avg	QoS (server-side)	DOUBLE	1	Edge site average I/O utlization	0.30
io_util_max	QoS (server-side)	DOUBLE	1	Edge site maximum I/O utilization	0.34
ng_traf_level	QoS (server-side)	DOUBLE	1	Edge site normalized bandwidth level in range [0, 1)	0.88
synack1_ratio	QoS (client-side)	DOUBLE	4	One-time success rate of establishing TCP connections.	0.86
tcp_conntime	QoS (client-side)*	BIGINT	4	The time from when the server receives a SYN packet to when it receives an ACK from the client (ms).	33
icmp_lossrate	QoS (client-side)	DOUBLE	3	The ICMP (Layer 3) packet loss rate	0.05
icmp_rtt	QoS (client-side)	DOUBLE	3	The ICMP (Layer 3) round trip time (ms)	21
499_5xx_ratio	QoS (client-side)	DOUBLE	5	The ratio of 499 and 5xx issued by Tengine	0.01
reset_ratio	QoS (client-side)	DOUBLE	5	Percentage of reset requests	0.22
buffer_rate	QoE	DOUBLE	5	The percentage of the video that experiences frozen or stalling events during the playback period.	0.04

* Moreover, the startup delay QoE metric can be approximated by $4 * \texttt{tcp\_conntime} + \texttt{avg\_fbt\_time}$.

Flink-based [21] framework, respectively, with statistical information presented through Apache Grafana [37].

Ethical concerns. Our data collection system adhered to the agreement established between content providers and end-users. The study participants include voluntarily opted-in end-users, and the analysis was conducted in compliance with Institutional Review Board. To protect participants' privacy, no personally identifiable information, such as phone numbers, IMEIs, or IMSIs, was collected during the study. Additional measures were implemented, including anonymization of customer IDs during QoE metric transmission, secure encryption of client-side metrics using HTTPS, strict prohibition of cookies over HTTPS to avoid linkage with users' personal identifiable information that the user once registered to obtain the necessary services, and aggregation of metrics across end users at each edge site for group-level analytics.

2.3 Comparison with Related Datasets

Based on the aforementioned system, we have released SNESet, a dataset containing 9 million clean records from 8 VSAs collected over a period of four months in 2022, covering end-users from 798 edge sites, 30 cites, and 3 ISPs in one country. Similar to traditional methods [1, 26, 29], our data cleaning primarily focuses on correcting erroneous values and imputing missing values with mean or median using programmatic heuristic approaches. Figure 2 (a) depicts the specific number of records for each application [3].

In Table 1, we highlight the limitations of existing open-source datasets and justify the need for SNESet. Most realistic platform datasets only include sever-side QoS metrics and lack or

[3]The terms "domain" and "application" are used interchangeably in the following sections unless explicitly stated otherwise.

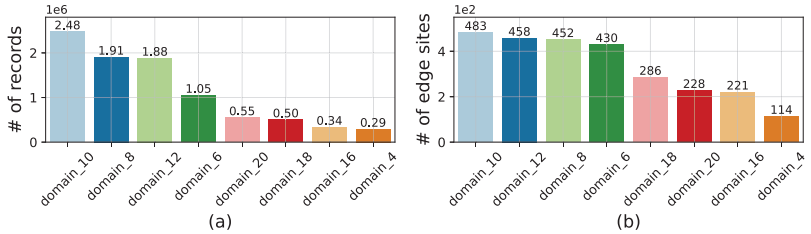


Fig. 2. The number of records (a) and edge sites (b) of each applications. Both x axes represent different applications.

have limited client-side QoE metrics (e.g., Alibaba Dataset [38], Azure Dataset [27] [4], and Google Dataset [100]). QoS encompasses the performance of each component within end-to-end video delivery system, including content origin sites, edge platforms, and ISPs involved in last-mile delivery to end-users [3]. As is defined by International telecommunication union (ITU) [50], QoE refers to the user's delight or annoyance and is typically measured using subjective metrics like MOS, a score between 1 and 5. However, subjective metrics are costly and complex to collect [23]. Therefore, our focus is on objective QoE metrics that are easily measured and compared, such as the stall ratio, which is widely used to characterize the experience of VSAs [60]. In the following sections, we will use "QoE" to refer specifically to objective QoE metrics.

Existing open-source QoE datasets primarily focus on client-side metrics, with little inclusion of QoS metrics, such as LiSSi lab Dataset [7] and Puffer [114] Dataset. These datasets are mostly collected from simulated testbeds, which may not accurately reflect real-world deployment scenarios for commercial VSAs. Moreover, the SNESet dataset offers a more comprehensive scope, covering not only service-side QoS and client-side QoE metrics (more horizontal diversity), but also a wider range of network protocol stacks (more vertical hierarchy). As illustrated in Table 2, SNESet covers four network protocol stack: application layer, transport layer, network layer, and physical layer. In contrast, most existing datasets are limited to one or two protocol stack layers.

The existing datasets mainly target testbeds and traditional centralized cloud platforms, typically with one or few site in a country. While the Edge Dataset [112] does contain a significant number of edge sites, the number of sites in SNESet is approximately 5.7× higher. More specifically, Figure 2 (b) shows the number of edge sites of different applications deployment. Additionally, SNESet offers comprehensive affiliation information of end users such as edge sites, cities, and internet service providers (ISPs), providing a potential understanding of the characteristic of ISP backbone network and scheduling policy.

Recently, similar to Huawei Dataset [102], Cisco has built a testbed to investigate the relationship between QoE and QoS for real-time communication services [23]. However, their dataset is not publicly available. The limitations of current datasets has prompted the creation of SNESet. A comprehensive schema and metric description of SNESet are listed in Table 2.

Unique properties of SNESet. (1) Comprehensive coverage: SNESet stands out in terms of its metric diversity, wider coverage of specific metrics, and provides extensive affiliation information of end users. **(2) Large-scale edge platforms in the wild:** Unlike previous datasets collected from simulated testbeds, SNESet is derived from the real-world, large-scale public edge platform NEP, featuring a 5.7× higher number of edge sites compared to previous datasets.

Limitations of SNESet and future plan. (1) The current version of SNESet primarily focuses on the stall ratio and has limited QoE metrics. However, we can estimate the start-up delay QoE

[4]We only compared the Alibaba Dataset (2018 version) and Azure Dataset (2019 version) as they closely match the edge scenarios. The latest versions of these datasets focus on emerging scenarios such as GPU clusters and microservices, which are not directly relevant to VSA.

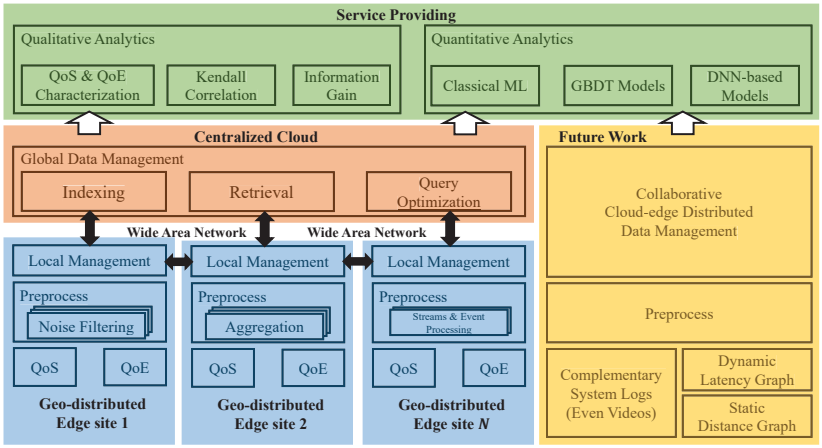


Fig. 3. Paradigm of Potential Data-centric Problems.

using the `tcp_conntime` and `avg_fbt_time` metrics. Specifically, in the country where `SNESet` is collected, TLS 1.2 is predominantly used (about 90%), allowing us to calculate the start-up delay QoE as $4 * \texttt{tcp\_conntime} + \texttt{avg\_fbt\_time}$. We acknowledge the importance of other metrics such as average playback bitrate and playback bitrate jitter for comprehensive QoE evaluation [9]. To enhance the QoE evaluation process, we plan to incorporate these metrics in future versions of `SNESet`. (2) To expand its scope, we also plan to include additional QoS metrics such as TCP telemetry information, BBR bandwidth estimation, and chunk-level video quality assessment (SSIM or resolution) in the future. (3) Currently, `SNESet` mainly covers edge sites within its home country. However, future plans involve expanding the dataset to encompass edge sites from various locations around the world.

2.4 Data-centric Challenges and Problems

As depicted in Figure 3, Section 2.2 introduces the data collection system, which consists of geo-distributed edge sites and the centralized cloud. Section 3 and Section 4 introduce two upper-layer use cases: qualitative analytics and quantitative analytics, respectively. The aforementioned represents our preliminary exploratory investigation into the QoS and QoE of VSAs on a large-scale edge platform. However, there remain numerous data-centric problems and challenges that require effective and efficient solutions in the future. Here we mainly divide these into three aspects: *data collection* (§ 2.4), *data preparation* (§ 3.4), and *data management* (§ 5). For a more comprehensive understanding of data-centric problems and challenges, we refer readers to [119] for an extensive overview.

In Section 2.2, we have introduced our data collection system. Data collection is the process of gathering and acquiring data, which fundamentally determines the data quality and quantity. With the increasing number of edge sites and VSA traffic, it is important for service providers to collect data that accurately reflects end users' QoE while minimizing costs. This objective requires both lightweight implementations and low runtime costs [19]. The former entails collecting data in a non-intrusive manner to minimize modifications to the existing codebase. The latter requires minimizing the impact of data collection on application performance, such as delay and throughput, during runtime. Considering the above aspects, the current version of `SNESet` primarily focuses

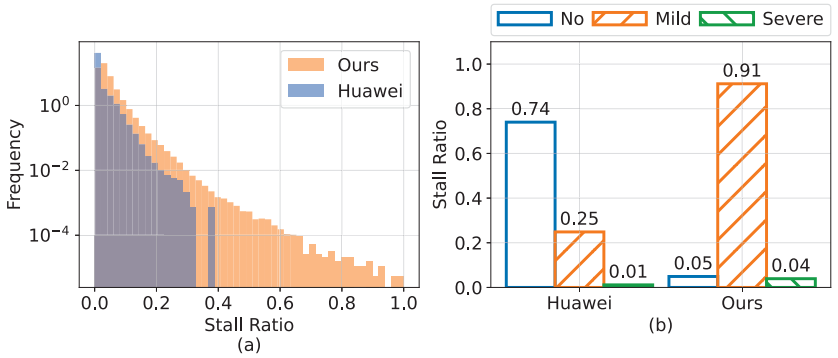


Fig. 4. Comparison of Huawei [102] and SNESet. (a) Histogram with 50 bins between 0 and 1; (b) Classification using the method of [102], where No/Mild/Severe corresponds to $0/(0, 0.1]/(0.1, 1]$ respectively.

on the stall ratio QoE. However, in the future, we plan to optimize these aspects to cover additional QoE metrics such as bitrate jitter. In addition, efficient integration [20, 113] of existing internal datasets [17, 34] and external datasets from third-party platforms [10, 14] is also a key consideration.

3 QOS AND QOE CHARACTERIZATION

In this section, we characterize the QoS and QoE metrics included in SNESet and investigate their relationship using Kendall correlation and information gain. Specifically, we select the most crucial QoS and QoE metrics to assess and evaluate, enabling us to gain valuable experiences and insights of VSAs' performance on large-scale edge platform.

3.1 QoE Characterization

Similar to many prior studies [77, 114, 117], SNESet is specifically focused on a key determinant of end-user QoEs: buffer_rate (namely stall ratio or re-buffering ratio). This metric represents the percentage of the video that experiences frozen or stalling events during the playback period.

SNESet reveals more severe stalling than prior datasets collected on small-scale testbeds, and provides more comprehensive coverage of high-value interval. Figure 4 (a) shows the histogram of the stall ratios of the Huawei testbed and our real system (50 bins between 0 and 1). Although both datasets have similar trends in the $[0, 0.3]$ interval, our dataset has more comprehensive coverage in the range of $[0.4, 1]$. Figure 4 (b) depicts the classification of the stall ratio using the method of [102], where No/Mild/Severe corresponds to $0/(0, 0.1]/(0.1, 1]$ respectively. In the Huawei testbed dataset, 74% of the records have a stall ratio of zero, while in our real scenario this proportion is only 5%. Moreover, in the real scenario, 91% of the stall ratios are concentrated in the interval (0,0.1], while in the Huawei testbed, only 25% of the stall ratios fall within this range. These findings highlight the differences between simulation testbed and real-world scenarios and underscore the need for a more fine-grained classification of stall ratios between (0, 0.1].

Stall ratio exists obvious diurnal pattern. Figure 5 depicts the average stall ratios of all the applications during a week. We observe rough diurnal patterns, with two prevalent peaks around 12 PM and 8 PM. There is a clear trough in the afternoon and early morning. In addition, it is intuitive that the stall ratio on weekends (120th-168th hour-of-week) is significantly higher than that on weekdays due to the backbone congestion.

Stall ratio is diversified across different applications, ISPs, and geographical locations. Figure 6 (a) shows that the mean and standard deviation of different applications' stall ratios vary significantly. For example, domain_18's average stall ratio is almost 3× higher than domain_16's. This difference is due to the distinct business types of different video applications. On-demand

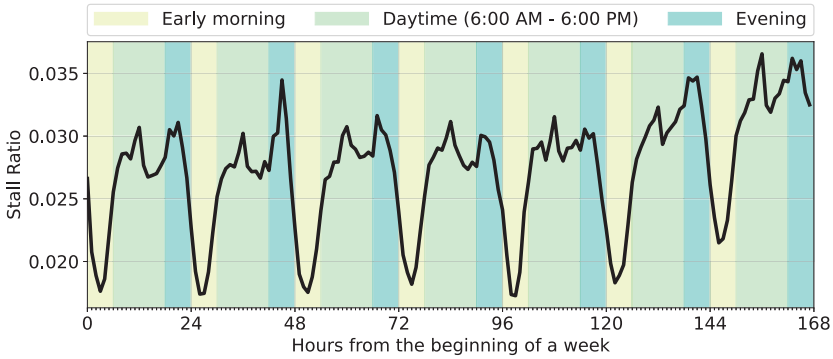


Fig. 5. Average stall ratio of all VSAs during a week.

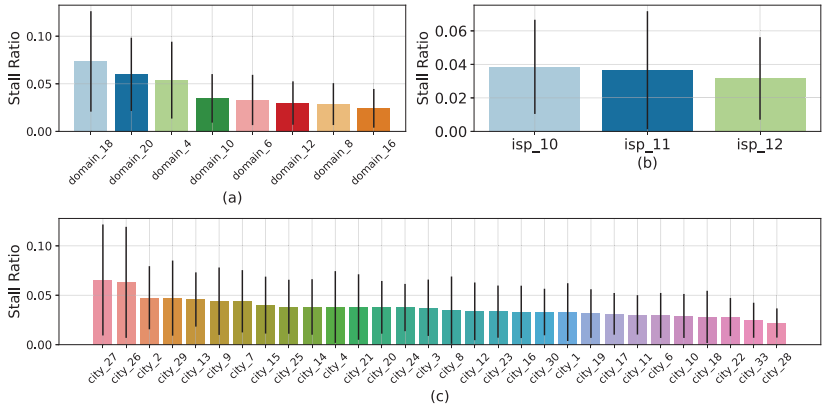


Fig. 6. The mean and standard deviation distribution of stall ratio classified by `domain_name` (a), `isp` (b), and `city` (c), respectively.

video applications like TikTok benefit from the CDN capabilities offered by edge sites, typically resulting in a small stall ratio. On the other hand, live streaming or real-time communication video applications require additional codec and preprocessing processes. They are more sensitive to network fluctuations and weak network conditions and generally experience a higher stall ratio. In Figure 6 (b), we observe that there is not much difference in the stall ratio among different ISPs. The slight difference may result from the ISP's coverage density difference. Finally, Figure 6 (c) shows that most cities have little difference in stall ratio, while some cities may experience a slightly higher stall ratio due to the remote geographical location. It is also worth noting that the mean and standard deviation is of the same order of magnitude, and typically, the more significant the mean, the larger the standard deviation.

Implications: For edge vendors, there are several key considerations. (1) The testbed's lower stall ratio may not accurately reflect the actual scenario, requiring in situ learning. (2) Proactive resource pre-provisioning is essential due to the periodic nature of the stall ratio, mitigating potential future peaks. (3) Customized solutions are necessary for optimizing different applications on a case-by-case basis. (4) Expanding existing edge sites or adding new sites in high-stall-rate cities should be considered for addressing stall issues in the future.

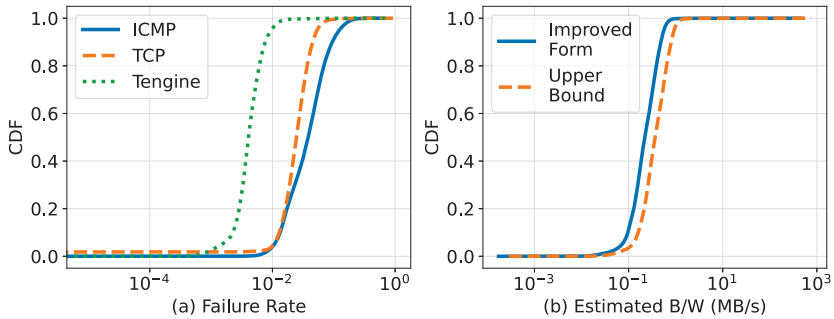


Fig. 7. (a) CDF distribution of different layers' failure rate (ICMP/TCP/Tengine represents Layer 3/4/5 respectively.); (b) CDF distribution of two types of estimated TCP throughput.

3.2 QoS Characterization

SNESet contains multi-layer metrics to measure end-to-edge network latency and various network failure rate. SNESet contains ICMP RTT (icmp_rtt, Layer 3) and TCP connection time (tcp_conntime, Layer 4) to characterize the network latency between end-users and edge sites (end-to-edge). Both of the two metrics are mainly distributed in [10, 100] ms [5]. More specifically, 93.36% of ICMP RTTs were distributed in [10, 100] ms, which is 96.78% for TCP connection establishment time. In addition, we also found that the median of the ratio of TCP connection time to ICMP RTT is 1.6. The additional 0.6× might be caused by: (1) differences between ICMP and TCP transport paths and (2) the processing delay of more protocol stacks.

SNESet contains the failure rate of multiple layers (icmp_lossrate/synack1_ratio/ 499_5xx_ratio represents the failure rate of ICMP/TCP/Tengine, which is Layer 3/4/5 respectively). Figure 7 (a) shows the CDF distribution of them. The P99 value of ICMP loss rate is around 5× higher than its median, while the P99 value of TCP one-time success rate and Tengine failure rate are both about 3× higher than their respective medians. We can also observed that the failure rate tends to decrease as the hierarchy move up, which could be attributed to the error control mechanisms present at lower layers.

Case study: TCP throughput estimation. Although the NEP currently mainly employs BBR [22] as the congestion control algorithm, which offers real-time estimations of the minimum RTT and maximum bottleneck bandwidth for end-to-end links, SNESet currently does not incorporate BBR telemetry information. However, the comprehensive metrics mentioned above aid in estimating the bandwidth of end-to-end network through alternative approaches. [71] provides a short and useful formula for the *upper bound* on the TCP throughput:

$$\mathrm{TP} < \mathrm{MSS}/\mathrm{RTT} * (1/\sqrt{p}) \tag{1}$$

where TP is the TCP throughput; MSS is the maximum segment size (fixed for each Internet path, typically 1460 bytes); RTT is the round trip time (as measured by TCP); p is the packet loss rate.

Moreover, an *improved form* of the above equation can be found in [82] as follows:

$$\mathrm{TP} \approx \min\left(\frac{W_{\max}}{RTT}, \frac{1}{RTT\sqrt{\frac{2bp}{3}} + \min\left(1, 3\sqrt{\frac{3bp}{8}}\right) p\,(1+32p^2)}\right) \tag{2}$$

where W_{max} is the maximum congestion window size. b is the number of packets acknowledged by a delayed ACK. Many TCP receiver implementations send one cumulative ACK for two consecutive packets received [33], so b is typically 2. As shown in In Figure 7 (b), the upper bound and improved

[5]Due to space limitations, the figure is not shown.

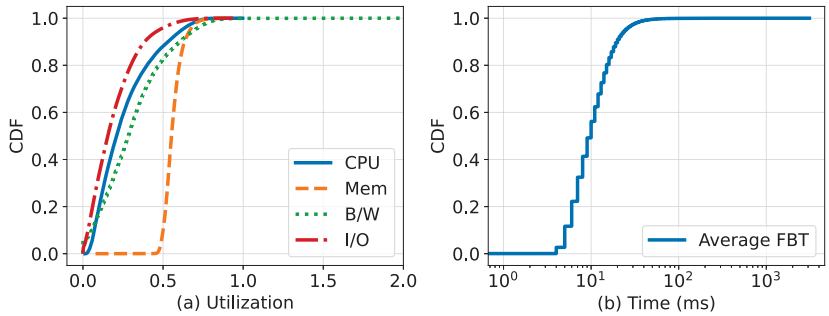


Fig. 8. (a) CDF distribution of SNESet's resource utilization; (b) CDF distribution of SNESet's average first byte return time.

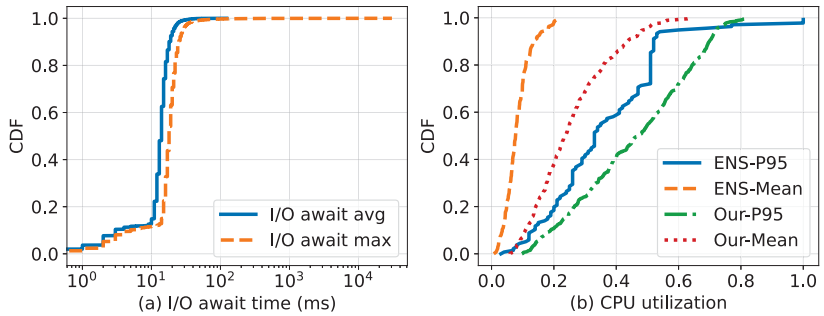


Fig. 9. (a) CDF distribution of SNESet's average and max I/O await time; (b) CDF distribution of P95 and mean CPU utilization of Edge dataset and SNESet;

form of estimated TCP throughput are heavily-tailed and both of them are predominantly distributed in [0.1MB/s, 10MB/s], accounting for 96.52% and 88.77% respectively. Moreover, the P99 upper bound of estimated TCP throughput is around 3.3× its median (0.36 MB/s).

Compared to existing datasets, SNESet contains more service-side resource usage metrics, and provides more comprehensive coverage of CPU and memory utilization. (1) Unlike existing QoS datasets [27, 38, 100, 112], SNESet not only captures CPU, memory, and bandwidth utilization, but also includes comprehensive I/O metrics such as average I/O time (io_await_avg) and utilization (io_util_avg). Figure 8 (a) depicts the CDF distribution of edge sites' CPU utilization, memory utilization, normalized bandwidth utilization and I/O utilization. As shown in Figure 9 (a), the max I/O await time is heavily-tailed and the P99 of it is about 4× higher than its median. (2) Compared to Alibaba Dataset [38] and Edge Dataset [112], SNESet has much more even coverage of CPU and memory utilization. As shown in Figure 9 (b), the average CPU utilization of Edge Dataset is mainly concentrated in [0, 0.2]. The average CPU utilization of SNESet (red dotted line) has much more even coverage. This might be due to an upgrade of the platform internal resources management strategy, or to the proliferation of applications deployment on NEP. Similarly, as shown in Figure 10, we can also observe this phenomenon when compared to Alibaba ECS both in terms of CPU and memory utilization. Despite having more even coverage, the metrics of SNESet are generally less utilized compared to ECS (centralized cloud data centers). This further suggests that NEP will need to deploy more efficient resource management strategies in the future to achieve similar utilization as centralized cloud data centres.

SNESet comprises several metrics that describe the inner networks of each edge site and vary across applications. Specifically, SNESet includes metrics such as avg_fbt_time, inner_network_rtt and inner_network_droprate. The CDF distribution of edge sites' average

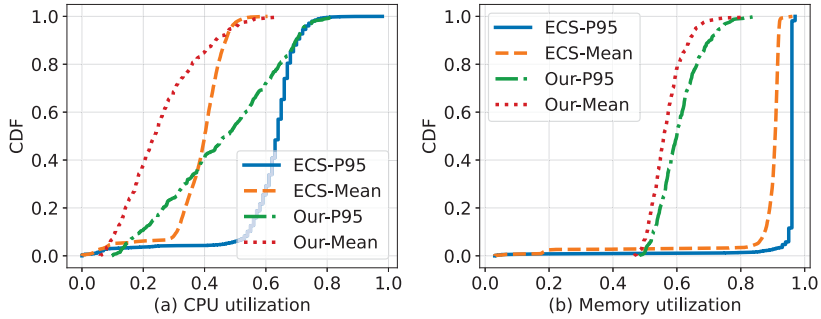


Fig. 10. CDF distribution of P95 and mean CPU/memory utilization of Alibaba ECS and our SNESet

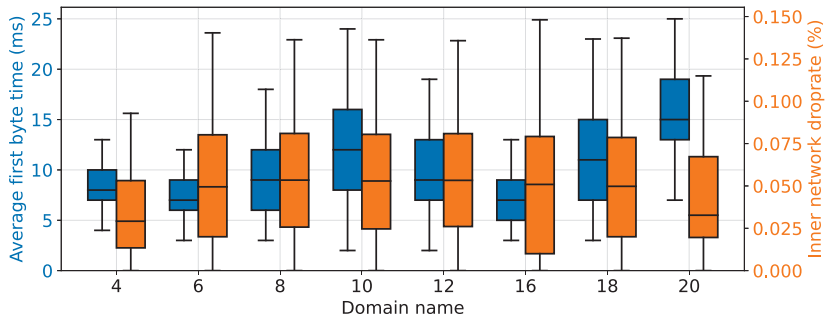


Fig. 11. Box plots of different applications' average first byte time and inner network droprate.The boxes depict the 25th, 50th, and 75th percentiles, respectively; the two whiskers are one interquartile range (IQR) past the low and high quartiles.

first byte return time (AFBT) is illustrated in Figure 8 (b), indicating a heavily-tailed distribution. The 95th-percentile of the AFBT is 2.5× higher than its median (10ms), which could be attributed to initial requests for specific content, leading to retrieval from the source site or centralized cloud.

In Figure 11, we display the distribution of AFBT and inner network droprate across various applications. Most domains have a comparable distribution of average inner network droprate at 0.05%, but domains_4 (0.0353%) and domains_20 (0.0432%) exhibit notably lower droprates than others. Conversely, the differences in AFBT among applications are more pronounced, with most applications having a median AFBT of around 10ms or less. However, domain_20 has the highest median AFBT at 15ms, while domain_16 has the lowest at 7ms. These variations in AFBT and inner network droprate may be due to the differences in the lengths of intra-network resource paths that various applications must traverse.

Implications: SNESet includes diverse QoS metrics from various network stacks, enabling edge vendors and content providers to explore joint optimization of QoE through multi-layer network protocol stacks. Moreover, it also opens up possibilities for edge vendors to develop QoE-aware QoS optimization techniques that maximize resource utilization while ensuring a positive user experience.

3.3 Relationship Between QoS and QoE Metrics

In this section, we investigate the correlation and information gain between QoS and QoE metrics. Note that these two classes of evaluation methods are complementary. Correlation provides a first-order summary of monotone relationships between stall ratio and QoS metrics. The information

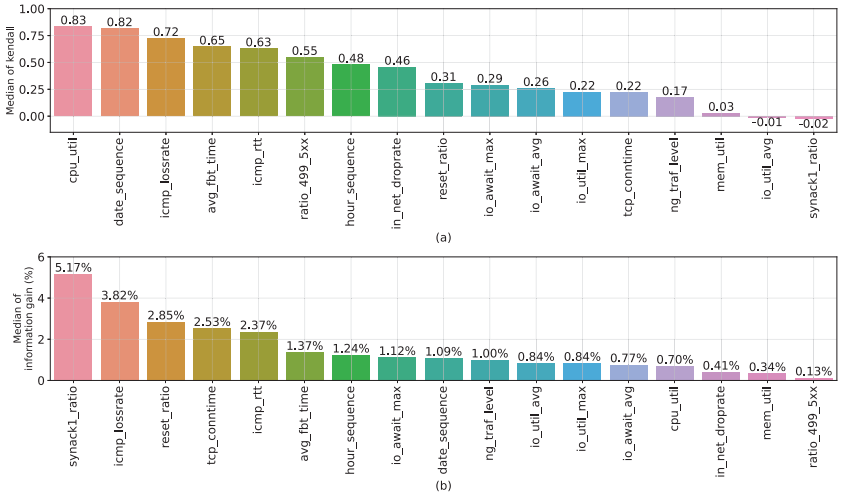


Fig. 12. (a) The median Kendall correlation coefficient of different QoS metrics. (b) The median information gain of different QoS metrics.

gain can corroborate or augment the correlation when the relationship is not monotone. Further, it provides a more in-depth understanding of the interaction between the QoS metrics by extending to the multivariate case.

Correlation. To avoid making assumptions about the relationships between the variables, we choose the *Kendall correlation* instead of the Pearson correlation. *Kendall correlation* is a rank correlation that does not assume anything about the underlying distributions, noise, or nature of relationships. In contrast, Pearson correlation assumes Gaussian noise and a roughly linear relationship between variables [30]. To compute the correlation, we bin stall ratio based on the QoS metrics, using bin sizes appropriate for each QoS metric of interest. For ratio variables between 0 and 1, we use 0.01 bins, and for delay-related variables, we use 1ms intervals. The *Kendall correlation* coefficients are then computed between the mean-per-bin vector and the values of the bin indices for each application. Using these "binned" correlation metrics allows us to retain the qualitative properties we are interested in while reducing computational costs.

Information Gain. Correlations are helpful for understanding the interaction between variables if the relationship is monotonic, but this is not always the case. Moreover, we aim to move beyond single metric analysis as correlation-based analysis cannot address whether QoS metrics complement each other or capture the same effects, nor can it determine which top k metrics to optimize for improved user experience. To address the above challenges, we augment the correlation analysis using the information gain [75], which is based on the concept of entropy. The entropy of random variable Y and the conditional entropy of Y given another random variable X is defined as:

$$H(Y) = -\sum_i P\left[Y = y_i\right] \log P\left[Y = y_i\right] \tag{3}$$

$$H(Y|X) = \sum_j P\left[X = x_j\right] H(Y|X = x_j) \tag{4}$$

where $P\left[Y = y_i\right]$ is the probability that $Y = y_i$. And the relative information gain is $\frac{H(Y)-H(Y|X)}{H(Y)}$. As with the correlation, we bin the data into discrete bins with the same bin specifications.

Figure 12 (a) presents the median Kendall correlation coefficients of various QoS metrics, with CPU utilization having the highest correlation with stall ratio, likely due to both metrics exhibiting peaks during the same hour of the day. Additionally, the correlation between stall ratio and date is relatively high, possibly due to ISPs intentionally throttling bandwidth at the end of the month.

Notably, memory utilization, average I/O utilization, and TCP one-time success rate do not show any clear monotonic relationship with stall ratio. Figure 12 (b) displays the median information gain of different QoS metrics. Despite a near-zero Kendall correlation, the TCP one-time success rate (`synack1_ratio`) has the highest relative information gain, significantly higher than other metrics. The lower correlation indicates a uniform `synack1_ratio` distribution across stall ratio intervals. In contrast, the higher information gain shows that the division of `synack1_ratio` substantially reduce the heterogeneity of stall ratio. Moreover, the top five metrics with the highest gain are all network-related. Finally, most QoS metrics have an information gain of around 1%.

Implications: Improving the end-to-end network performance is still the primary measure to reduce the stall ratio. Due to the lower impact of memory and I/O utilization on stall ratio, for edge vendors, it is promising to colocate memory or I/O-intensive services with VSAs to improve the overall resource utilization.

3.4 Data-centric Challenges and Problems

In this section, we have involved data processing, which generally includes the cleaning [1, 26, 29], transforming [8, 94, 116], and augmenting [63, 73] raw data for downstream tasks. Each edge site generates massive daily data volumes (TB, 10^{12}), contributing to the overall data scale of NEP reaching petabytes (10^{15}). Filtering redundant information and reducing complexity (feature size or sample size) are crucial for resource-constrained edge sites. In contrast, data augmentation enhances diversity and addresses class imbalance issues. Currently, these sub-modules are manually executed based on expert experience. However, automating the search for optimal combinations and parameters, known as pipeline search [85, 93, 110], faces challenges due to the substantial computational overhead and the exponential growth of search space. Therefore, more efficient and effective search strategies are need in real-world scenarios.

4 QOS-TO-QOE BENCHMARK AND RESULTS

In previous sections, we *qualitatively* analyze the relationship between QoS and QoE metrics. In this section, we aim to utilize learning-based models to *quantitatively* measure the impact of different QoS metrics. The reason is that QoE metrics are commonly located in the upper application layer, while QoS metrics reside in the lower system layer. Directly obtaining QoE metrics can be challenging for some VSAs due to permissions or framework limitations. Therefore, QoS-based prediction serves as a non-intrusive approach to assess QoE.

Specifically, in this section, we first put existing methods into a taxonomy and present how each method works. Finally, to meet the real-time requirements of QoS and QoE detection in practical scenarios, we evaluate various methods in terms of the prediction accuracy and time efficiency.

4.1 Problem Statement

Definition 1 (QoS-to-QoE Prediction). *Assume we observe a set of records $\mathcal{D} = \{(\boldsymbol{x}_t, \boldsymbol{y}_t)\}_{t=1,...,T}$, where $\boldsymbol{x}_t = (x_t^1, \ldots, x_t^m)$ is a vector of m QoS metrics (features) and $\boldsymbol{y}_t$ is the vector of corresponding QoE metrics (labels). Records $(\boldsymbol{x}_t, \boldsymbol{y}_t)$ are independent and identically distributed according to some unknown distribution $P(\cdot, \cdot)$. Suppose that our prediction model F has parameters θ. Thus, the QoS-to-QoE prediction can be formulated as follows:*

$$\hat{\boldsymbol{y}}_t = F(\boldsymbol{x}_t, \theta) = F(x_t^1, x_t^2, ..., x_t^m, \theta) \tag{5}$$

where $\hat{\mathbf{y}}_t$ is the model prediction results. The goal is to train a model $F(\cdot, \theta) : \mathbb{R}^m \rightarrow \mathbb{R}$, which minimizes the expected loss $\mathcal{L}(F) := \mathbb{E}[L(\mathbf{y}, F(\mathbf{x}, \theta))]$. Here $L(\cdot, \cdot)$ is the loss function and $(\mathbf{x}, \mathbf{y})$ is a test record sampled from P independently of the training set $\mathcal{D}$.

4.2 Taxonomy and Workflow

The current mainstream methods can be divided into classical ML methods (e.g. linear regression, SVM and decision trees), Gradient Boosting Decision Trees (e.g. LightGBM, XGBoost and CatBoost) and DNN-based methods.

4.2.1 *Classical ML Methods.*

Linear regression-ElasticNet. ElasticNet [126] is a typical linear regression method that combines the residual sum of squares between the predictions and targets with both L_1-norm (Lasso) and L_2-norm (Ridge) regularization.

Support Vector Machine for Regression (SVR). SVR aims to minimize the L_2-norm regularization, not the squared error. The error, however, is maintained in constraint with margin ϵ. For data points fall outside the margin, their deviations are added to the objective with weight C.

Decision Trees (DTs). DTs are a non-parametric supervised learning method that recursively partitions the feature space such that the samples with similar target values are grouped together. Random Forest (RF) [43] is a variant of DTs that builds multiple trees independently. Each tree is constructed from a random subsample of the training set and their predictions are averaged.

4.2.2 *Gradient Boosting Decision Trees (GBDT) Methods.*

GBDT are a typical class of methods where base estimators are built sequentially, and each estimator attempts to reduce the bias of the former combined estimators. The current mainstream GBDT methods include: LightGBM, XGBoost, CatBoost.

LightGBM [56]. LightGBM aims to improve the efficiency and scalability when the feature dimension is high and data size is large. Specifically, LightGBM introduces two novel techniques: *Gradient-based One-Side Sampling (GOSS)* and *Exclusive Feature Bundling (EFB)*. With GOSS, it exclude a significant proportion of data instances with small gradients and use only the remainder for estimation. With EFB, it bundle mutually exclusive features to reduce the number of features.

XGBoost [25]. XGBoost is a scalable end-to-end tree boosting system. The scalability of XGBoost is due to several important algorithmic and system optimizations. From the algorithmic aspect, it proposes a novel *sparsity-aware algorithm* for sparse data and *weighted quantile sketch* for approximate tree learning. From the system aspect, it improves in two unexplored directions: *out-of-core computation* and *cache-aware learning*.

CatBoost [84]. CatBoost (Categorical Boosting) introduce two key algorithmic techniques: *ordered boosting*, a permutation-driven alternative to the classical algorithm, and an innovative algorithm for processing categorical features.

4.2.3 *DNN-based Methods.*

Similar to [47], we introduce DNN-based model and the model architecture is shown in Figure 13. Specifically, each categorical features are first fed into an individual embedding layer. The remaining numerical features are fed into a fully connected layer with output dimensions equal to the embedding dimension. We then perform element-wise product between each pair of embedding vector and numerical output vector, concatenate the resulting vectors with the original embedding vectors, and feed the concatenated vectors into the next multi-layer blocks. Finally, the output layer implemented by a fully connected layer generates the final prediction. Here our DNN-based model is a simple multilayer perceptron. We leave more complicated DNN models built with LSTM or even Transformer as a direction for future work.

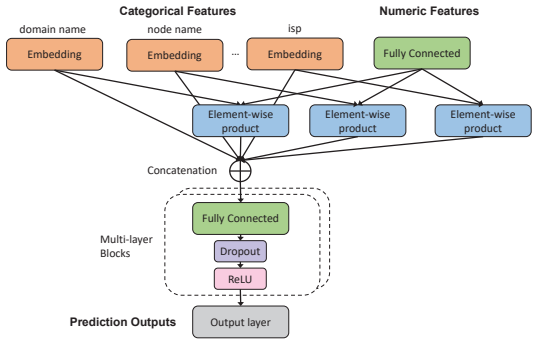


Fig. 13. The architecture of deep neural network model.

4.3 Experimental Setup

Evaluation Metric. We evaluate different models based on their prediction accuracy and time efficiency. (1) For prediction accuracy, we utilize two metrics: Mean Absolute Error (MAE) and Symmetric Mean Absolute Percentage Error (SMAPE). Since the stall ratio typically falls between 0 and 1 and most values are concentrated between 0 and 0.1 (as demonstrated in Figure 4), Mean Squared Error (MSE) - which is a commonly used metric in prediction tasks - can result in very small values. Therefore, we opted not to use MSE. Here is a brief description of MAE and SMAPE:

- The mean absolute error (MAE) reflects the average of the absolute errors. To compare the MAE, we multiply it by 100 since the original value range of the stall ratios is [0, 1].

$$MAE = \frac{1}{T_{test}} \sum_{t=1}^{T_{test}} |\hat{y}_t - y_t| \tag{6}$$

- The symmetric mean absolute percentage error (SMAPE) reflects the percentage of the error to the ground-truth value. Since it is scale-independent, prediction errors are considered regardless of the magnitude of the records.

$$SMAPE = \frac{1}{T_{test}} \sum_{t=1}^{T_{test}} \frac{|\hat{y}_t - y_t|}{(|y_t| + |\hat{y}_t|)/2} * 100\% \tag{7}$$

where T_{test} is the number of records in test datasets, $\hat{y}_t$ is the forecast value of the ground truth y_t. For all two of them, the lower value represents the higher accuracy of prediction.

(2) We evaluate time efficiency using three metrics: the average training time and inference time of each fold, and the time taken for hyper-parameter tuning.

Methods Implementation. We implemented all machine learning methods using Scikit-learn 1.1.3 and employed random search [13] to determine the best hyperparameters for each model. For the DNN model, we used Pytorch 1.8.1 with a batch size of 64 and 50 training epochs. We optimized the DNN model with the Adam optimizer [57], and its learning rate decayed with the ExponentialLR learning rate scheduler starting from 0.01. Additionally, we applied the same feature engineering to process the raw data for both the ML and DNN models and utilized mean absolute error as the loss function for both models. Finally, we evaluated accuracy and time efficiency using 5-fold cross-validation.

Hardware and Platform. All the models are trained and evaluated upon a Linux server with 80 Intel(R) Xeon(R) Gold 5218R @ 2.10GHz, 819GB RAM, and NVIDIA Tesla-V100.

Table 3. Comparison of model accuracy in terms of MAE and SMAPE across varying dataset sizes. "-" indicates that the training time for each fold of cross-validation exceeds 12 hours.

Methods	5K		10K		50K		100K		0.5M	
	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE
ElasticNet	2.3124	63.30%	2.3346	63.41%	2.3175	63.47%	2.3339	63.75%	2.3330	63.54%
SVR	2.2345	71.27%	2.5799	64.76%	2.3310	70.16%	2.5708	67.25%	2.8742	78.96%
RF	2.2211	60.16%	2.2430	60.84%	-	-	-	-	-	-
LightGBM	2.1181	60.83%	2.1819	62.18%	2.2124	63.34%	2.2347	63.52%	2.2546	63.24%
XGBoost	2.2151	61.09%	2.2544	62.05%	2.2694	62.78%	2.2614	62.35%	2.3105	63.05%
CatBoost	2.2161	60.97%	2.2748	62.25%	2.3035	62.91%	2.3274	63.36%	2.3657	63.75%
DNN	2.2157	64.49%	2.4095	70.02%	2.3628	68.93%	2.2432	68.14%	2.3105	72.19%

Table 4. Comparison of model training time and inference time across varying dataset sizes (measured in seconds).

Methods	5K		10K		50K		100K		0.5M	
	Training Time (s)	Inference Time (s)	Training Time (s)	Inference Time (s)	Training Time (s)	Inference Time (s)	Training Time (s)	Inference Time (s)	Training Time (s)	Inference Time (s)
ElasticNet	0.51	0.00	1.86	0.01	3.42	0.01	8.59	0.01	26.05	0.02
SVR	5.29	0.01	6.85	0.01	44.87	0.01	110.72	0.02	555.56	0.03
RF	93.53	0.01	321.82	0.02	-	-	-	-	-	-
LightGBM	13.03	0.02	14.75	0.03	51.02	0.27	70.30	0.45	264.42	2.14
XGBoost	43.15	0.03	87.09	0.07	368.85	0.20	1328.98	0.70	8141.11	5.11
CatBoost	16.45	0.01	18.28	0.01	146.43	0.03	173.31	0.04	351.99	0.10
DNN	37.04	0.06	93.63	0.19	315.98	0.48	622.31	0.84	2288.67	2.68

4.4 Experimental Results and Analysis

This section evaluates the mainstream methods on `SNESet` for QoS-based QoE regression. We present the experimental results regarding prediction accuracy and time efficiency in the domain-general (for all the applications) and domain-specific scenario (for specific applications). Particularly, we aim to answer the following research questions via the experiments:

- **RQ1**: In the domain-general scenario, how does the dataset's scalability affect the performance of different methods?
- **RQ2**: In the domain-specific scenario, how does the model accuracy change over time (by training models using data from the first month and evaluating their performance using data from the subsequent one or two months as the testing set)?
- **RQ3**: Since the accuracy of GBDT methods are generally better, what's the feature importance of different GBDT models?
- **RQ4**: Considering the timeliness of real-world deployment, how GPU affect the performance of GBDT methods?

*4.4.1 **Domain-general Experiments (RQ1)**.*

Table 3 and Table 4 show the comparison of different models' accuracy in terms of MAE and SMAPE and the comparison of models' training time and inference time across varying dataset sizes respectively. From the evaluation results, we summarize three key observations for classical ML methods: (1) With the increase of dataset size, the accuracy of the ElasticNet remains relatively stable. In contrast, the performance fluctuation of SVR is more pronounced, with the maximum MAE/SMAPE being 2.8742/78.96% on the 0.5M/0.5M dataset and the minimum being 2.2345/64.76% on the 5K/10K dataset. This may be related to the optimization objective of the SVR model, which only has the L2-norm of the model coefficient vector and considers the prediction error only as a model optimization constraint. (2) When the size of dataset is small (5K and 10K), the Random Forest model has the highest accuracy in terms of SMAPE due to the aggregation of multiple base estimator predictions, resulting in a smaller relative error. However, due to the need to aggregate multiple base estimators, the increase in time expenditure becomes significant as the dataset size

Table 5. Performance comparison of different models using the first month's dataset for training and the datasets from the following one or two months for testing.

Methods	Train		Test (1 month)		Test (2 months)	
	MAE	SMAPE	MAE	SMAPE	MAE	SMAPE
ElasticNet	0.78	39.80%	1.07 (↑ 0.29)	55.46% (↑ 15.66%)	1.32 (↑ 0.54)	57.95% (↑ 18.15%)
SVR	1.00	56.43%	1.14 (↑ 0.14)	60.59% (↑ 4.16%)	1.40 (↑ 0.39)	62.90% (↑ 6.47%)
RF	0.68	36.75%	0.93 (↑ 0.25)	45.53% (↑ 8.78%)	1.16 (↑ 0.48)	48.13% (↑ 11.38%)
LightGBM	0.61	33.88%	0.842 (↑ 0.24)	41.90% (↑ 8.02%)	1.28 (↑ 0.67)	50.36% (↑ 16.48%)
Xgboost	0.55	31.19%	0.86 (↑ 0.31)	42.37% (↑ 11.18%)	1.20 (↑ 0.65)	47.55% (↑ 16.36%)
Catboost	0.59	33.12%	0.841 (↑ 0.25)	41.60% (↑ 8.48%)	1.21 (↑ 0.62)	47.72% (↑ 14.60%)
DNN	0.99	47.33%	0.96 (↓ 0.03)	44.38% (↓ 2.95%)	1.21 (↑ 0.22)	47.79% (↑ 0.46%)

increases. For datasets larger than 10K samples, the training time for each fold of cross-validation exceeds 12 hours. As a result, we use "-" to indicate in those cases. (3) Across various dataset sizes, classical ML methods generally have a lower average inference time than tree-based boosting methods and DNN-based methods.

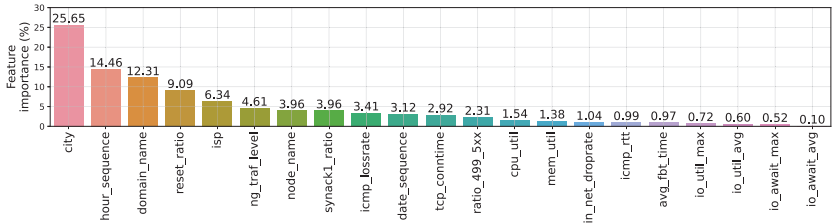
When it comes to the tree-based boosting methods, we first conclude that the tree-based boosting methods generally outperform classical machine learning methods. In addition, among the three gradient boosting decision tree (GBDT) methods, LightGBM has the best MAE accuracy and the shortest training time. When the dataset size is equal to or larger than 50K, XGBoost achieves the highest SMAPE accuracy. However, the average training time of the XGBoost model is also the highest among the three GBDT methods, which becomes more pronounced as the dataset size increases. For example, when the dataset size is 0.5M, the average training time of the XGBoost model is more than 10× that of LightGBM and CatBoost. Correspondingly, the average inference time of the CatBoost model is the smallest among the three GBDT methods. When the dataset size is greater than 10K, it is approximately one-tenth of LightGBM and XGBoost. DNN-based models may underperform linear regression methods in certain cases, which could be due to the unrepresentative training dataset or the simpler model structure that requires more careful design.

4.4.2 *Domain-specific Experiments (RQ2).*

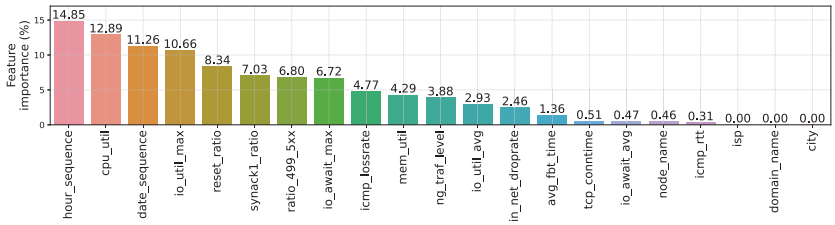
Table 5 shows the performance comparison of different methods in terms of MAE and SMAPE in the specific environment. From the evaluation results, we summarize three key observations as follows: (1) Similar to the general environment, tree-based boosting methods generally outperform classical ML methods in terms of both MAE and SMAPE. (2) Moreover, the prediction errors of both classical ML methods and tree-based boosting methods, measured by MAE and SMAPE, tend to increase gradually over time. Among them, ElasticNet exhibits the most significant error growth in SMAPE, with an increase of 15.66% and 18.15%, respectively. (3) However, it is worth noting that the DNN-based method shows the smallest increase in MAE and SMAPE over time and even achieves a lower error in the following month. Compared to classical ML methods and GBDT methods, DNN-based methods can capture more complex implicit correlations.

4.4.3 *Feature Importance of GBDT Methods (RQ3).*

Figure 14, Figure 15 and Figure 16 respectively illustrate the feature importance of three GBDT methods in domain-general and domain-specific scenarios. In the domain-general scenario, while the ranking order varies, the top five influential features for LightGBM and CatBoost are generally

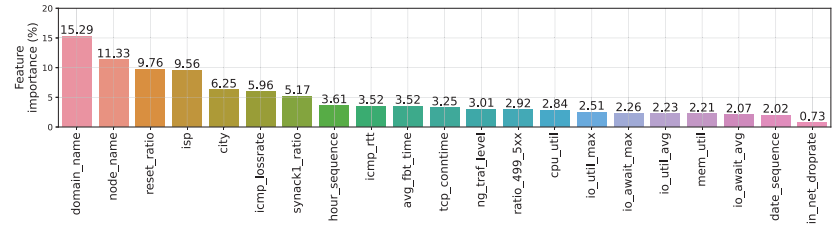


(a) The feature importance of LightGBM in the domain-gerneral scenario.

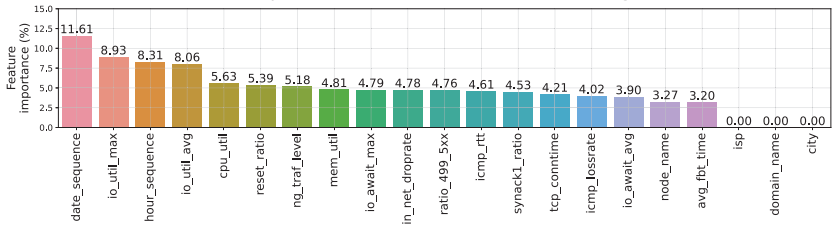


(b) The feature importance of LightGBM in the domain-specific scenario.

Fig. 14. The feature importance of LightGBM in the general and specific environment.



(a) The feature importance of XGBoost in the domain-gerneral scenario.



(b) The feature importance of XGBoost in the domain-specific scenario.

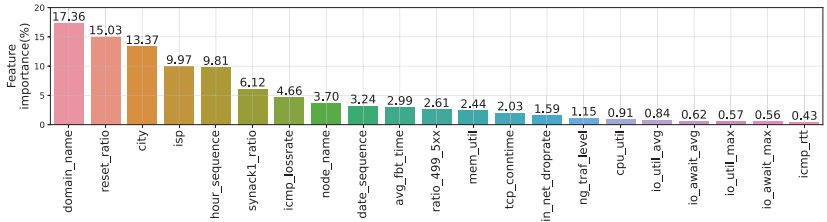
Fig. 15. The feature importance of XGBoost in the general and specific environment.

the same, namely 'domain_name', 'city', 'isp', 'hour', and 'reset_ratio'. However, it is worth noting that XGBoost is unique in that the top five features do not include 'hour' and 'city', but instead introduce 'node_name' and 'icmp_lossrate'.

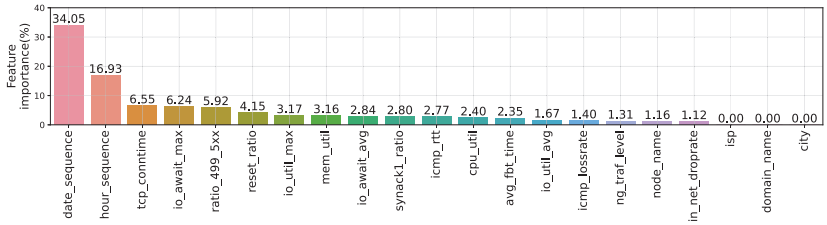
In comparison, in domain-specific senario, the top five most influential features of the three methods differ considerably, but they all include 'date', 'hour', and I/O-related features. Additionally, because the domain-specific senario involves data from the same domain within the same city and the same ISP, the feature importance of 'domain', 'isp', and 'prov' for all three methods is zero. Finally, it is worth noting that for CatBoost, in the specific environment, the impact of 'date' and 'hour' is significantly greater than that of other features, roughly $2 \sim 3\times$ higher.

4.4.4 *GPU Effects for GBDT Methods (RQ4).*

Effects on accuracy. For LightGBM and XGBoost, the same hyperparameters yield the same accuracy on both CPU and GPU. However, for CatBoost, the same hyperparameters result in significantly higher accuracy on GPU compared to CPU. Figure 17 depicts the comparison of MAE and SMAPE accuracies for CatBoost using both CPU and GPU across varying dataset size. For



(a) The feature importance of CatBoost in the domain-gerneral scenario.



(b) The feature importance of CatGBoost in the domain-specific scenario.

Fig. 16. The feature importance of CatBoost in the general and specific environment.

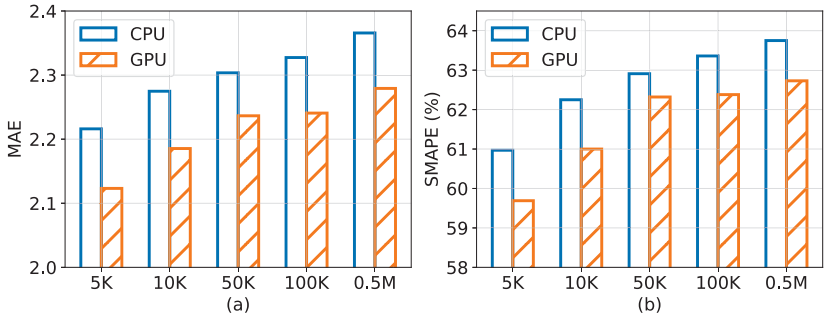


Fig. 17. Comparison of MAE and SMAPE for CatBoost model with CPU and GPU.

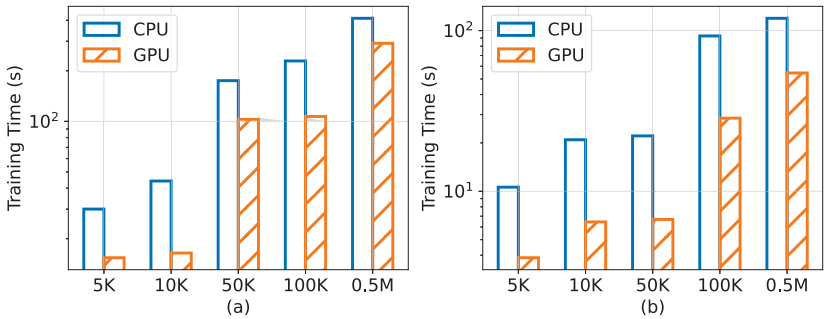


Fig. 18. Training time comparison. (a) The training time of LightGBM with CPU and GPU. (b) The training time of XGBoost with CPU and GPU.

example, for CatBoost, when the dataset size is 5K, the accuracy improvement is the greatest using GPU, reducing MAE and SMAPE by 4.19% and 2.10%, respectively, compared to CPU. The superior performance of the GPU versions of CatBoost can be attributed to their low-level optimization to utilize specific hardware resources on the GPU, such as high-speed memory and parallel computing. This optimization significantly improves the computational efficiency of the models, ultimately resulting in higher accuracy.

Effects on training time. The use of GPU significantly improves the training time of LightGBM and XGBoost. As is shown in Figure 18, for LightGBM, the training time with GPU is reduced by

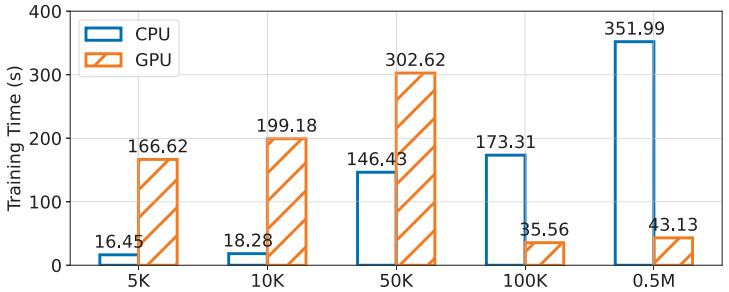


Fig. 19. CatBoost training time comparison.

an average of 46.97% compared to CPU, while for XGBoost, the reduction is 65.21% on average. It is worth noting that as shown in Figure 19, when the dataset size is less than or equal to 50K, the training time of CatBoost on GPU is higher than that on CPU. In these cases, the overhead of transferring data to the GPU and running computations on the GPU may outweigh the benefits of parallelization. Therefore, training on CPU may actually be faster.

4.4.5 *Main Findings*.

Our main findings of this section are summarized as follows:

- Random Forest (RF) generally outperform other classical machine learning methods. However, the training time of RF grows exponentially as the size of dataset increases.
- In both the general and specific environment, the GBDT methods generally outperform classical machine learning methods. Among the three GBDT methods, LightGBM has the best MAE accuracy and the shortest training time.
- DNN-based methods demonstrate more robust performance over time, capturing the complex and implicit correlations.
- In the general environment, the top five influential features for GBDT methods are generally the same. However, in the specific environment, significant differences are exhibited.
- CatBoost achieves improved accuracy on GPUs compared to CPUs with the same hyperparameters. Moreover, GPU can speed up the training time of GBDT methods on large dataset, however it cannot make them as quick as classical ML methods.

5 DISCUSSION AND FUTURE WORK

Effective DNN-based model. Although this study employed a DNN-based method with a fully connected neural network, several advanced neural networks could also be applied to the QoS-based QoE problem. For instance, a graph neural network could capture the spatial correlation among different edge sites, while recurrent neural networks could capture temporal dependencies. Future research could investigate the applicability and effectiveness of these neural networks in addressing the QoS-based QoE problem.

Online processing. Previous sections mainly focus on the offline scenarios of QoS-based QoE prediction. Although Section 4.4.2 briefly discusses the online scenarios, the next phase of online processing (deployment) should consider dealing with dataset shift [90] and catastrophic forgetting [88] with the trade-off between the accuracy and efficiency of the frequent model updates on a real deployment environment.

Diagnosis and cost-efficient QoE optimization. This paper primarily focused on measuring the impact of QoS metrics on user QoE. Future work could explore mechanisms to proactively diagnose quality issues to minimize their impact on users' QoE and find the most cost-efficient methods to maximise QoE improvements.

Data integration. In this paper, we mainly focus on combining both QoS and QoE records from different edge sites. However, to improve data quality, it may be necessary to integrate data from various sources, including third parties. This might present potential challenges such as security risks and inconsistent data sources.

Data Cleaning. Currently, our data cleaning methods primarily center on rectifying erroneous values and imputing missing values with mean or median using programmatic heuristic approaches. However, such heuristics can be inaccurate or ineffective. Thus, exploring learning-based methods [42, 55] that not solely focus on the cleaning itself but also enhance the final model performance holds significant promise.

Cloud-edge collaborative distributed data management. Data management system is responsible for handling the underlying raw data and providing interfaces for upper-layer applications. Currently, each edge site persists the structured QoS and semi-structured QoE datasets on SSDs and transmits them to the centralized cloud for management via HTTPS. However, the centralized data management exist the limitations of significant network bandwidth overhead and transmission time consumption. To address these challenges, there is an urgent need for a cloud-edge collaborative distributed data management system. Unlike existing distributed data management systems [45, 52, 111], the unique challenge is threefold: (1) In the future, the underlying data consist not only of structured QoS data and semi-structured QoE data but also semi-structured system logs, graph data and even videos. (2) The number of geo-distributed edge sites is expected to be in the hundreds or thousands, which is significantly higher than the typical distributed databases that support mostly dozens of nodes. And the connection between these sites is established through wide area networks (WAN) instead of local area networks (LAN), leading to increased network throughput fluctuations. Moreover, compliance with respect to dataflow constraints is also a key consideration, controlling the movement of data across borders [12]. (3) The data management system also need to guarantee low latency and high throughput for concurrent query requests from upper-layer applications, such as batch processing, OLTP, and OLAP. These require new designs of cache and materialized views for these large-scale WAN-connected edge sites, improving query performance and simplifying data access while reducing maintenance overhead.

6 RELATED WORK

QoS characterization of commercial edge/cloud platforms. Recent work on datacenter characterization has mainly focused on the centralized cloud datacenter (Microsoft [27, 40, 92], Facebook [72, 91], Google [76, 100, 103], and Alibaba [39, 68, 109]) and seldom targets the geo-distributed edge sites. Xu *et al.* [112] perform a first-of-its-kind measurement study on a leading public edge platform in China. However, the number of network protocol stacks covered by their dataset is less than ours, and it does not include QoE-related metrics. Wang *et al.* [107] examined the WAN traffic characteristics in Baidu's datacenter network, but the number of geo-distributed sites they studied is much smaller than NEP.

Modeling and optimizing QoE. In academia, the current research on modeling and optimizing QoE can be categorized into two main groups: the multimedia community and the network community. The multimedia community mainly focuses on visual quality assessment, traditionally relying on pixel-level patterns and data-driven models (e.g., PSNR [51], SSIM [108], and VMAF [80]) as well as deep learning models [32, 118, 122] to model users' perception of encoded video. Recent QoE models in this community have incorporated streaming-related incidents (join time, bitrate switches,etc) [31, 89] and visual attention [35, 81]. However, these models often focus on subjective metrics, such as MOS, which can be expensive and complex to collect.

In the network community, researchers have focused on adaptive bitrate algorithms that aim to maximize bitrate based on available network throughput. Traditional approaches include buffer-based [46, 48] and rate-based algorithms [54, 64, 95], while recent advancements utilize control theory [114, 117], ML-based throughput prediction [5, 96], or deep reinforcement learning [70]. However, the performance of these algorithms in emulated environments or synthetic datasets may not generalize to real-world scenario.

Measuring and characterizing video streaming applications (VSAs). Numerous studies on VSAs [2, 4, 11, 53, 101] have been conducted, covering various aspects such as architecture, performance characterization, and user access patterns, etc. Recent research has expanded to specific types of VSAs (live streaming [61, 66] and real-time communications [24, 65, 69]), 5G environments [79], and mobile devices [41]. While some prior works have implemented passive measurements in the wild [18, 74], their methodologies are specific to certain applications and may not be generalizable. Moreover, few studies have included both server-side QoS and client-side QoE. `SNESet` is the first active measurement dataset that captures server-side QoS and client-side QoE metrics for eight VSAs in large-scale edge sites.

Distributed data analytics. This is a well-studied problem in the literature [12, 28, 49, 62, 97, 115], with extensive research covering many aspects of the problem, including distributed query processing, parallel computing frameworks, data placement, and privacy considerations. Some studies have also focused on WAN-connected geo-distributed sites. For instance, [86] address the problem of optimal task placement among geo-distributed sites to minimize query response time. [105, 106] proposed WANalytics and Geode, aiming to minimize data transfer costs. Similarly, [104] proposed the Clarinet system, which considers WAN-aware optimization for achieving low query response times. However, unlike these works that typically involve only a few or dozens of sites, our edge platform consists of hundreds or thousands of WAN-connected geo-distributed edge sites.

7 CONCLUSION

In this paper, we propose `SNESet`, the first active measurement dataset of eight video applications, including both server-side QoS and client-side QoE metrics, on a public edge platform. We characterize the QoS and QoE metrics in `SNESet` and compare them with existing datasets. Our qualitative analysis examines the impact of QoS metrics on QoE using Kendall correlation and information gain. In contrast, our quantitative analysis measures the influence of different QoS metrics through mainstream regression methods. This study represents a preliminary exploratory analysis toward understanding the QoS on QoE of VSAs on a large-scale edge platform.

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their constructive comments. This work was supported by National Key R&D Program of China (No.2021ZD0113001) and Alibaba Group through Alibaba Innovative Research (AIR) Program. Mengwei Xu was partly supported by China Institute of IoT (Wuxi).

REFERENCES

[1] Ziawasch Abedjan, Xu Chu, Dong Deng, Raul Castro Fernandez, Ihab F Ilyas, Mourad Ouzzani, Paolo Papotti, Michael Stonebraker, and Nan Tang. 2016. Detecting data errors: Where are we and what needs to be done? *Proceedings of the VLDB Endowment* 9, 12 (2016), 993–1004.

[2] Vijay K Adhikari, Yang Guo, Fang Hao, Volker Hilt, Zhi-Li Zhang, and Matteo Varvello. 2014. Measurement study of Netflix, Hulu, and a tale of three CDNs. *IEEE/ACM Transactions On Networking* 23, 6 (2014), 1984–1997.

[3] Akamail. 2020. Measuring Video Quality and Performance: Best Practices. https://www.akamai.com/site/it/documents/white-paper/measuring-video-quality-and-performance-best-practices.pdf

[4] Zahaib Akhtar, Yun Seong Nam, Jessica Chen, Ramesh Govindan, Ethan Katz-Bassett, and etc. Rao, Sanjay. 2018. Understanding video management planes. In *Proceedings of the ACM IMC Conference*. 238–251.

[5] Zahaib Akhtar, Yun Seong Nam, Ramesh Govindan, Sanjay Rao, Jessica Chen, Ethan Katz-Bassett, Bruno Ribeiro, Jibin Zhan, and Hui Zhang. 2018. Oboe: Auto-tuning video ABR algorithms to network conditions. In *Proceedings of the ACM SIGCOMM Conference*. 44–58.

[6] Alibaba. 2023. What is ENS? https://www.alibabacloud.com/help/en/ens/latest/e49d6b

[7] Lamine Amour, Souihi Sami, Said Hoceini, and Abdelhamid Mellouk. 2015. Building a large dataset for model-based QoE prediction in the mobile environment. In *Proceedings of the ACM MSWiM Conference*. 313–317.

[8] Arvind Arasu et al. 2009. A grammar-based entity representation framework for data cleaning. In *Proceedings of the ACM SIGMOD Conference*. 233–244.

[9] Consumer Technology Association. 2020. *Streaming Quality of Experience Events, Properties and Metrics*. Technical Report. Arlington, VA, USA.

[10] Santiago Andrés Azcoitia et al. 2022. A survey of data marketplaces and their business models. *ACM SIGMOD Record* 51, 3 (2022), 18–29.

[11] Athula Balachandran, Vyas Sekar, Aditya Akella, Srinivasan Seshan, et al. 2013. Developing a predictive model of quality of experience for internet video. *ACM SIGCOMM Computer Communication Review* 43, 4 (2013), 339–350.

[12] Kaustubh Beedkar et al. 2021. Compliant geo-distributed query processing. In *Proceedings of the ACM SIGMOD Conference*. 181–193.

[13] James Bergstra and Yoshua Bengio. 2012. Random search for hyper-parameter optimization. *Journal of machine learning research* 13, 2 (2012).

[14] Anant Bhardwaj, Souvik Bhattacherjee, Amit Chavan, Amol Deshpande, Aaron J Elmore, et al. 2015. Datahub: Collaborative data science & dataset version management at scale. In *Proceedings of the CIDR Conference*.

[15] BITAG. 2021. 2020 PandemicNetwork Performance. https://bitag.org/documents/bitag_report.pdf

[16] Niklas Blum, Serge Lachapelle, et al. 2021. WebRTC: Real-time communication for the open web platform. *Commun. ACM* 64, 8 (2021), 50–54.

[17] Alex Bogatu, Alvaro AA Fernandes, Norman W Paton, and Nikolaos Konstantinou. 2020. Dataset discovery in data lakes. In *Proceedings of the IEEE ICDE Conference*. IEEE, 709–720.

[18] Francesco Bronzino, Paul Schmitt, Sara Ayoubi, Guilherme Martins, Renata Teixeira, and Nick Feamster. 2019. Inferring streaming video quality from encrypted traffic: Practical models and deployment experience. *Proceedings of the ACM SIGMETRICS Conference* 3, 3 (2019), 1–25.

[19] Wei Cao, Yusong Gao, Bingchen Lin, Xiaojie Feng, Yu Xie, Xiao Lou, and Peng Wang. 2018. Tcprt: Instrument and diagnostic analysis system for service quality of cloud databases at massive scale in real-time. In *Proceedings of the ACM SIGMOD Conference*. 615–627.

[20] Riccardo Cappuzzo, Paolo Papotti, and Saravanan Thirumuruganathan. 2020. Creating embeddings of heterogeneous relational datasets for data integration tasks. In *Proceedings of the ACM SIGMOD Conference*. 1335–1349.

[21] Paris Carbone, Asterios Katsifodimos, Stephan Ewen, Volker Markl, et al. 2015. Apache flink: Stream and batch processing in a single engine. *The Bulletin of the Technical Committee on Data Engineering* 38, 4 (2015).

[22] Neal Cardwell, Yuchung Cheng, and etc. Gunn, C Stephen. 2017. BBR: congestion-based congestion control. *Commun. ACM* 60, 2 (2017), 58–66.

[23] Giovanna Carofiglio, Giulio Grassi, Enrico Loparco, Luca Muscariello, Michele Papalini, and Jacques Samain. 2021. Characterizing the relationship between application QoE and network QoS for real-time services. In *Proceedings of the ACM SIGCOMM Workshop on Network-application Integration*. 20–25.

[24] Hyunseok Chang, Matteo Varvello, Fang Hao, and Sarit Mukherjee. 2021. Can you see me now? A measurement study of Zoom, Webex, and Meet. In *Proceedings of the ACM IMC Conference*. 216–228.

[25] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the ACM SIGKDD Conference*. 785–794.

[26] Xu Chu, Ihab F Ilyas, Sanjay Krishnan, and Jiannan Wang. 2016. Data cleaning: Overview and emerging challenges. In *Proceedings of the ACM SIGMOD Conference*. 2201–2206.

[27] Eli Cortez, Anand Bonde, Alexandre Muzio, Mark Russinovich, Marcus Fontoura, and Ricardo Bianchini. 2017. Resource central: Understanding and predicting workloads for improved resource management in large cloud platforms. In *Proceedings of the ACM SOSP Conference*. 153–167.

[28] Umur Cubukcu, Ozgun Erdogan, Sumedh Pathak, Sudhakar Sannakkayala, and Marco Slot. 2021. Citus: Distributed postgresql for data-intensive applications. In *Proceedings of the ACM SIGMOD Conference*. 2490–2502.

[29] Michele Dallachiesa, Amr Ebaid, Ahmed Eldawy, Ahmed Elmagarmid, Ihab F Ilyas, Mourad Ouzzani, and Nan Tang. 2013. NADEEF: a commodity data cleaning system. In *Proceedings of the ACM SIGMOD Conference*. 541–552.

[30] Florin Dobrian, Vyas Sekar, Asad Awan, Ion Stoica, Dilip Joseph, and Aditya Ganjam. 2011. Understanding the impact of video quality on user engagement. *ACM SIGCOMM Computer Communication Review* 41, 4 (2011), 362–373.

[31] Zhengfang Duanmu, Wentao Liu, Diqi Chen, Zhuoran Li, Zhou Wang, Yizhou Wang, and Wen Gao. 2019. A knowledge-driven quality-of-experience model for adaptive streaming videos. *arXiv preprint arXiv:1911.07944* (2019).
[32] Nagabhushan Eswara, S Ashique, Anand Panchbhai, Soumen Chakraborty, Hemanth P Sethuram, Kiran Kuchi, Abhinav Kumar, and Sumohana S Channappayya. 2019. Streaming video QoE modeling and prediction: A long short-term memory approach. *IEEE Transactions on Circuits and Systems for Video Technology* 30, 3 (2019), 661–673.
[33] Kevin R Fall and W Richard Stevens. 2011. *TCP/IP illustrated, volume 1: The protocols*. addison-Wesley.
[34] Raul Castro Fernandez, Ziawasch Abedjan, Famien Koko, Gina Yuan, Samuel Madden, and Michael Stonebraker. 2018. Aurum: A data discovery system. In *Proceedings of the IEEE ICDE Conference*. IEEE, 1001–1012.
[35] Guanyu Gao, Linsen Dong, Huaizheng Zhang, and Yonggang Wen. 2019. Content-aware personalised rate adaptation for adaptive streaming via deep video analysis. In *Proceedings of the IEEE ICC Conference*. IEEE, 1–8.
[36] Nishant Garg. 2013. *Apache kafka*. Packt Publishing Birmingham, UK.
[37] Apache Grafana. 2023. Apache Grafana. https://grafana.com/
[38] Alibaba Group. 2018. Alibaba Cluster Trace Program. https://github.com/alibaba/clusterdata/blob/master/cluster-trace-v2018/trace_2018.md
[39] Jing Guo, Zihao Chang, Sa Wang, Haiyang Ding, Yihui Feng, Liang Mao, and Yungang Bao. 2019. Who limits the resource efficiency of my datacenter: An analysis of alibaba datacenter traces. In *Proceedings of the IEEE IWQoS Conference*. 1–10.
[40] Ori Hadary, Luke Marshall, Ishai Menache, Abhisek Pan, Esaias E Greeff, David Dion, Star Dorminey, Shailesh Joshi, Yang Chen, Mark Russinovich, et al. 2020. Protean: VM allocation service at scale. In *Proceedings of the USENIX OSDI Conference*. 845–861.
[41] Bo Han, Yu Liu, and Feng Qian. 2020. ViVo: Visibility-aware mobile volumetric video streaming. In *Proceedings of the ACM MobiCom Conference*. 1–13.
[42] Arvid Heise, Gjergji Kasneci, et al. 2014. Estimating the number and sizes of fuzzy-duplicate clusters. In *Proceedings of the ACM CIKM Conference*. 959–968.
[43] Tin Kam Ho. 1995. Random decision forests. In *Proceedings of International Conference on Document Analysis and Recognition*, Vol. 1. IEEE, 278–282.
[44] Yun Chao Hu, Milan Patel, Dario Sabella, Nurit Sprecher, and Valerie Young. 2015. Mobile edge computing—A key technology towards 5G. *ETSI white paper* 11, 11 (2015), 1–16.
[45] Haoyu Huang and Shahram Ghandeharizadeh. 2021. Nova-LSM: a distributed, component-based LSM-tree key-value store. In *Proceedings of the ACM SIGMOD Conference*. 749–763.
[46] Tianchi Huang, Chao Zhou, Rui-Xiao Zhang, Chenglei Wu, et al. 2020. Stick: A harmonious fusion of buffer-based and learning-based approach for adaptive streaming. In *Proceedings of the IEEE INFOCOM Conference*. IEEE, 1967–1976.
[47] Tianchi Huang, Chao Zhou, Rui-Xiao Zhang, Chenglei Wu, and Lifeng Sun. 2022. Learning tailored adaptive bitrate algorithms to heterogeneous network conditions: A domain-specific priors and meta-reinforcement learning approach. *IEEE Journal on Selected Areas in Communications* 40, 8 (2022), 2485–2503.
[48] Te-Yuan Huang, Ramesh Johari, Nick McKeown, et al. 2014. A buffer-based approach to rate adaptation: Evidence from a large video streaming service. In *Proceedings of the ACM SIGCOMM Conference*. 187–198.
[49] Yuzhen Huang, Xiao Yan, Guanxian Jiang, Tatiana Jin, James Cheng, An Xu, et al. 2019. Tangram: bridging immutable and mutable abstractions for distributed data analytics. In *Proceedings of the USENIX ATC Conference*. 191–206.
[50] ITU. 2017. Vocabulary for performance, quality of service and quality of experience. https://www.itu.int/rec/T-REC-P.10-201711-I/en
[51] Subramania Jayaraman, S Esakkirajan, and T Veerakumar. 2009. *Digital image processing*. Vol. 7014. Tata McGraw Hill Education New Delhi.
[52] Martin Jergler, Mohammad Sadoghi, and Hans-Arno Jacobsen. 2015. D2WORM: A management infrastructure for distributed data-centric workflows. In *Proceedings of the ACM SIGMOD Conference*. 1427–1432.
[53] Junchen Jiang, Vyas Sekar, Ion Stoica, et al. 2013. Shedding light on the structure of internet video quality problems in the wild. In *Proceedings of the ACM CoNEXT Conference*. 357–368.
[54] Junchen Jiang, Vyas Sekar, and Hui Zhang. 2012. Improving fairness, efficiency, and stability in http-based adaptive video streaming with festive. In *Proceedings of the ACM CoNEXT Conference*. 97–108.
[55] Zhimeng Jiang, Kaixiong Zhou, Zirui Liu, Li Li, Rui Chen, Soo-Hyun Choi, and Xia Hu. 2021. An information fusion approach to learning with instance-dependent label noise. In *Proceedings of the ICLR Conference*.
[56] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. Lightgbm: A highly efficient gradient boosting decision tree. *Proceedings of the NIPS Conference* 30 (2017).
[57] Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
[58] Kongjian. 2022. Taobao System Activity Reporter. https://github.com/alibaba/tsar

[59] Alok Gautam Kumbhare, Reza Azimi, Ioannis Manousakis, Anand Bonde, et al. 2021. Prediction-Based power oversubscription in cloud platforms.. In *Proceedings of the USENIX ATC Conference*. 473–487.
[60] Adam Langley, Alistair Riddoch, Alyssa Wilk, Antonio Vicente, Charles Krasic, Dan Zhang, Fan Yang, Fedor Kouranov, Ian Swett, Janardhan Iyengar, et al. 2017. The quic transport protocol: Design and internet-scale deployment. In *Proceedings of the ACM SIGCOMM Conference*. 183–196.
[61] Jinyang Li, Zhenyu Li, Ri Lu, Kai Xiao, Songlin Li, Jufeng Chen, Jingyu Yang, Chunli Zong, Aiyun Chen, Qinghua Wu, et al. 2022. LiveNet: a low-latency video transport network for large-scale live streaming. In *Proceedings of the ACM SIGCOMM Conference*. 812–825.
[62] Xiang Li, Fabing Li, and Mingyu Gao. 2023. Flare: A Fast, Secure, and Memory-Efficient Distributed Analytics Framework. *Proceedings of the VLDB Endowment* 16, 6 (2023), 1439–1452.
[63] Yuliang Li, Xiaolan Wang, Zhengjie Miao, and Wang-Chiew Tan. 2021. Data augmentation for ml-driven data preparation and integration. *Proceedings of the VLDB Endowment* 14, 12 (2021), 3182–3185.
[64] Zhi Li, Xiaoqing Zhu, Joshua Gahm, Rong Pan, Hao Hu, Ali C Begen, and David Oran. 2014. Probe and adapt: Rate adaptation for HTTP video streaming at scale. *IEEE Journal on Selected Areas in Communications* 32, 4 (2014), 719–733.
[65] Xianshang Lin, Yunfei Ma, Junshao Zhang, Yao Cui, Jing Li, Shi Bai, Ziyue Zhang, Dennis Cai, Hongqiang Harry Liu, and Ming Zhang. 2022. GSO-simulcast: global stream orchestration in simulcast video conferencing systems. In *Proceedings of the ACM SIGCOMM Conference*. 826–839.
[66] Xingchi Liu, Mahsa Derakhshani, and Lyudmila Mihaylova. 2022. Risk-Aware Contextual Learning for Edge-Assisted Crowdsourced Live Streaming. *IEEE Journal on Selected Areas in Communications* (2022).
[67] YouTube Live. 2023. https://www.youtube.com/live
[68] Shutian Luo, Huanle Xu, Chengzhi Lu, Kejiang Ye, Guoyao Xu, Liping Zhang, Yu Ding, Jian He, and Chengzhong Xu. 2021. Characterizing microservice dependency and performance: Alibaba trace analysis. In *Proceedings of the ACM SoCC Conference*. 412–426.
[69] Kyle MacMillan, Tarun Mangla, and James Saxon. 2021. Measuring the performance and network utilization of popular video conferencing applications. In *Proceedings of the ACM IMC Conference*. 229–244.
[70] Hongzi Mao, Ravi Netravali, et al. 2017. Neural adaptive video streaming with pensieve. In *Proceedings of the ACM SIGCOMM Conference*. 197–210.
[71] Matthew Mathis, Jeffrey Semke, Jamshid Mahdavi, and Teunis Ott. 1997. The macroscopic behavior of the TCP congestion avoidance algorithm. *ACM SIGCOMM Computer Communication Review* 27, 3 (1997), 67–82.
[72] Justin Meza, Tianyin Xu, Kaushik Veeraraghavan, and Onur Mutlu. 2018. A large scale study of data center network reliability. In *Proceedings of the ACM IMC Conference*. 393–407.
[73] Zhengjie Miao, Yuliang Li, and Xiaolan Wang. 2021. Rotom: A meta-learned data augmentation framework for entity matching, data cleaning, text classification, and beyond. In *Proceedings of the ACM SIGMOD Conference*. 1303–1316.
[74] Oliver Michel, Satadal Sengupta, Hyojoon Kim, Ravi Netravali, and Jennifer Rexford. 2022. Enabling passive measurement of zoom performance in production networks. In *Proceedings of the ACM IMC Conference*. 244–260.
[75] Tom Michael Mitchell et al. 2007. *Machine learning*. Vol. 1. McGraw-hill New York.
[76] Ricky KP Mok, Hongyu Zou, Rui Yang, Tom Koch, Ethan Katz-Bassett, and Kimberly C Claffy. 2021. Measuring the network performance of Google Cloud platform. In *Proceedings of the ACM IMC Conference*. 54–61.
[77] Hyunwoo Nam, Kyung-Hwa Kim, and Henning Schulzrinne. 2016. QoE matters more than QoS: Why people stop watching cat videos. In *Proceedings of the IEEE INFOCOM Conference*. IEEE, 1–9.
[78] Jyoti Nandimath, Ekata Banerjee, Ankur Patil, Pratima Kakade, Saumitra Vaidya, et al. 2013. Big data analysis using Apache Hadoop. In *Proceedings of the IEEE IRI Conference*. IEEE, 700–703.
[79] Arvind Narayanan, Xumiao Zhang, Ruiyang Zhu, Ahmad Hassan, Shuowei Jin, Xiao Zhu, Xiaoxuan Zhang, Denis Rybkin, and Zhengxuan Yang. 2021. A variegated look at 5G in the wild: performance, power, and QoE implications. In *Proceedings of the ACM SIGCOMM Conference*. 610–625.
[80] Netflix. 2022. VMAF - Video Multi-Method Assessment Fusion. https://github.com/Netflix/vmaf
[81] Cagri Ozcinar and Julián Cabrera. 2019. Visual attention-aware omnidirectional video streaming using optimal tiles for virtual reality. *IEEE Journal on Emerging and Selected Topics in Circuits and Systems* 9, 1 (2019), 217–230.
[82] Jitendra Padhye, Victor Firoiu, Don Towsley, and Jim Kurose. 1998. Modeling TCP throughput: A simple model and its empirical validation. In *Proceedings of the ACM SIGCOMM Conference*. 303–314.
[83] H Parmar and M Thornburgh. 2012. Adobe's real time messaging protocol. *Copyright Adobe Systems Incorporated* (2012), 1–52.
[84] Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. *Proceedings of the NIPS Conference* 31 (2018).
[85] Fotis Psallidas, Yiwen Zhu, Bojan Karlas, Jordan Henkel, Matteo Interlandi, Subru Krishnan, Brian Kroth, Venkatesh Emani, Wentao Wu, Ce Zhang, et al. 2022. Data Science Through the Looking Glass: Analysis of Millions of GitHub Notebooks and ML. NET Pipelines. *ACM SIGMOD Record* 51, 2 (2022), 30–37.

[86] Qifan Pu, Ganesh Ananthanarayanan, Peter Bodik, Srikanth Kandula, Aditya Akella, et al. 2015. Low latency geo-distributed data analytics. *ACM SIGCOMM Computer Communication Review* 45, 4 (2015), 421–434.
[87] Eric Rescorla. 2000. *Http over tls*. Technical Report.
[88] Anthony Robins. 1995. Catastrophic forgetting, rehearsal and pseudorehearsal. *Connection Science* 7, 2 (1995), 123–146.
[89] Werner Robitza, Marie-Neige Garcia, and Alexander Raake. 2017. A modular http adaptive streaming qoe model—candidate for itu-t p. 1203 ("p. nats"). In *Proceedings of the IEEE QoMEX Conference*. IEEE, 1–6.
[90] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. 2011. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 627–635.
[91] Arjun Roy, Hongyi Zeng, Jasmeet Bagga, George Porter, and Alex C Snoeren. 2015. Inside the social network's (datacenter) network. In *Proceedings of the ACM SIGCOMM Conference*. 123–137.
[92] Mohammad Shahrad, Rodrigo Fonseca, Íñigo Goiri, Gohar Chaudhry, Paul Batum, Jason Cooke, Eduardo Laureano, Colby Tresness, Mark Russinovich, and Ricardo Bianchini. 2020. Serverless in the wild: Characterizing and optimizing the serverless workload at a large cloud provider. In *Proceedings of the USENIX ATC Conference*. 205–218.
[93] Zeyuan Shang, Emanuel Zgraggen, Benedetto Buratti, Ferdinand Kossmann, Philipp Eichmann, Yeounoh Chung, Carsten Binnig, Eli Upfal, and Tim Kraska. 2019. Democratizing data science through interactive curation of ml pipelines. In *Proceedings of the ACM SIGMOD Conference*. 1171–1188.
[94] Jie Song, George Alter, and HV Jagadish. 2019. C2Metadata: Automating the capture of data transformations from statistical scripts in data documentation. In *Proceedings of the ACM SIGMOD Conference*. 2005–2008.
[95] Kevin Spiteri, Rahul Urgaonkar, and Ramesh K Sitaraman. 2020. BOLA: Near-optimal bitrate adaptation for online videos. *IEEE/ACM Transactions On Networking* 28, 4 (2020), 1698–1711.
[96] Yi Sun, Xiaoqi Yin, Junchen Jiang, Vyas Sekar, Fuyuan Lin, Nanshu Wang, Tao Liu, and Bruno Sinopoli. 2016. CS2P: Improving video bitrate selection and adaptation with data-driven throughput prediction. In *Proceedings of the ACM SIGCOMM Conference*. 272–285.
[97] Rebecca Taft, Irfan Sharif, Andrei Matei, Nathan VanBenschoten, Jordan Lewis, Tobias Grieger, Kai Niemi, Andy Woods, Anne Birzin, Raphael Poss, et al. 2020. Cockroachdb: The resilient geo-distributed sql database. In *Proceedings of the ACM SIGMOD Conference*. 1493–1509.
[98] Alibaba Taobao Tengine. 2023. Retrieved Jul 20, 2023 from https://github.com/alibaba/tengine
[99] TikTok. 2023. https://www.tiktok.com/
[100] Muhammad Tirmazi, Adam Barker, Nan Deng, Md E Haque, Zhijing Gene Qin, Steven Hand, et al. 2020. Borg: the next generation. In *Proceedings of the ACM EuroSys Conference*. 1–14.
[101] Ruben Torres, Alessandro Finamore, Jin Ryong Kim, Marco Mellia, et al. 2011. Dissecting video server selection strategies in the youtube cdn. In *Proceedings of the IEEE ICDCS Conference*. IEEE, 248–257.
[102] Vladislav Vasilev, Jérémie Leguay, Stefano Paris, Lorenzo Maggi, and Mérouane Debbah. 2018. Predicting QoE factors with machine learning. In *Proceedings of the IEEE ICC Conference*. IEEE, 1–6.
[103] Abhishek Verma, Luis Pedrosa, Madhukar Korupolu, David Oppenheimer, Eric Tune, and John Wilkes. 2015. Large-scale cluster management at Google with Borg. In *Proceedings of the ACM EuroSys Conference*. 1–17.
[104] Raajay Viswanathan, Ganesh Ananthanarayanan, and Aditya Akella. 2016. CLARINET:WAN-Aware Optimization for Analytics Queries. In *Proceedings of the USENIX OSDI Conference*. 435–450.
[105] Ashish Vulimiri, Carlo Curino, P Brighten Godfrey, Thomas Jungblut, Jitu Padhye, and George Varghese. 2015. Global analytics in the face of bandwidth and regulatory constraints. In *Proceedings of the USENIX NSDI Conference*. 323–336.
[106] Ashish Vulimiriu, Carlo Curinom, Brighten Godfreyu, Konstantinos Karanasosm, and George Varghesem. 2015. WANalytics: Analytics for a geo-distributed data-intensive world. Proceedings of the CIDR Conference.
[107] Zhaohua Wang, Zhenyu Li, Guangming Liu, Yunfei Chen, and Qinghua Wu. 2021. Examination of WAN traffic characteristics in a large-scale data center network. In *Proceedings of the ACM IMC Conference*. 1–14.
[108] Zhou Wang, Eero P Simoncelli, and Alan C Bovik. 2003. Multiscale structural similarity for image quality assessment. In *The Thrity-Seventh Asilomar Conference on Signals, Systems & Computers, 2003*, Vol. 2. Ieee, 1398–1402.
[109] Qizhen Weng, Wencong Xiao, Yinghao Yu, Wei Wang, Cheng Wang, Jian He, Yong Li, Liping Zhang, Wei Lin, and Yu Ding. 2022. MLaaS in the Wild: Workload Analysis and Scheduling in Large-Scale Heterogeneous GPU Clusters. In *Proceedings of the USENIX NSDI Conference*. 945–960.
[110] Doris Xin, Hui Miao, Aditya Parameswaran, and Neoklis Polyzotis. 2021. Production machine learning pipelines: Empirical analysis and optimization opportunities. In *Proceedings of the ACM SIGMOD Conference*. 2639–2652.
[111] Pengcheng Xiong, Hakan Hacigumus, et al. 2014. A software-defined networking based approach for performance management of analytical queries on distributed data stores. In *Proceedings of the ACM SIGMOD Conference*. 955–966.
[112] Mengwei Xu, Zhe Fu, Xiao Ma, Li Zhang, Yanan Li, Feng Qian, Shangguang Wang, Ke Li, et al. 2021. From cloud to edge: a first look at public edge platforms. In *Proceedings of the ACM IMC Conference*. 37–53.

[113] An Yan and Bill Howe. 2021. Equitensors: Learning fair integrations of heterogeneous urban data. In *Proceedings of the ACM SIGMOD Conference*. 2338–2347.
[114] Francis Y Yan, Hudson Ayers, Chenzhi Zhu, Sadjad Fouladi, James Hong, Keyi Zhang, Philip Alexander Levis, and Keith Winstein. 2020. Learning in situ: a randomized experiment in video streaming. In *Proceedings of the USENIX NSDI Conference*, Vol. 20. 495–511.
[115] Xinan Yan, Linguan Yang, Hongbo Zhang, Xiayue Charles Lin, Bernard Wong, et al. 2018. Carousel: Low-latency transaction processing for globally-distributed data. In *Proceedings of the ACM SIGMOD Conference*. 231–243.
[116] Gunce Su Yilmaz, Tana Wattanawaroon, Liqi Xu, Abhishek Nigam, Aaron J Elmore, and Aditya Parameswaran. 2018. Datadiff: User-interpretable data transformation summaries for collaborative data analysis. In *Proceedings of the ACM SIGMOD Conference*. 1769–1772.
[117] Xiaoqi Yin, Abhishek Jindal, Vyas Sekar, and Bruno Sinopoli. 2015. A control-theoretic approach for dynamic adaptive video streaming over HTTP. In *Proceedings of the ACM SIGCOMM Conference*. 325–338.
[118] Junyong You. 2021. Long short-term convolutional transformer for no-reference video quality assessment. In *Proceedings of the ACM MM Conference*. 2112–2120.
[119] Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. 2023. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158* (2023).
[120] Yue Zha and Jing Li. 2020. Virtualizing FPGAs in the cloud. In *Proceedings of the ACM ASPLOS Conference*. 845–858.
[121] Li Zhang, Zhe Fu, Boqing Shi, Xiang Li, Rujin Lai, Chenyang Chen, Ao Zhou, Xiao Ma, Shangguang Wang, and Mengwei Xu. 2022. SoC-Cluster as an Edge Server: an Application-driven Measurement Study. *arXiv preprint arXiv:2212.12842* (2022).
[122] Zicheng Zhang, Wei Wu, Wei Sun, Dangyang Tu, Wei Lu, Xiongkuo Min, Ying Chen, and Guangtao Zhai. 2023. MD-VQA: Multi-Dimensional Quality Assessment for UGC Live Videos. In *Proceedings of the IEEE CVPR Conference*.
[123] Azure Edge Zones. 2020. Microsoft partners with the industry to unlock new 5G scenarios with Azure Edge Zones. Retrieved Mar 31, 2020 from https://azure.microsoft.com/en-us/blog/microsoft-partners-with-the-industry-to-unlock-new-5g-scenarios-with-azure-edge-zones/
[124] AWS Local Zones. 2022. Retrieved Mar 31, 2022 from https://aws.amazon.com/cn/about-aws/global-infrastructure/localzones/locations/
[125] Zoom. 2023. https://zoom.us/
[126] Hui Zou and Trevor Hastie. 2005. Regularization and variable selection via the elastic net. *Journal of the royal statistical society: series B (statistical methodology)* 67, 2 (2005), 301–320.

Received April 2023; revised July 2023; accepted August 2023