RIGA TECHNICAL UNIVERSITY

Tianhua Chen

# FINE-GRAINED MONITORING AND CLASSIFICATION OF INTERACTIVE MEDIA NETWORK TRAFFIC

Summary of the Doctoral Thesis

RTU Press
Riga 2026

**RIGA TECHNICAL UNIVERSITY**
Faculty of Computer Science, Information Technology and Energy
Institute of Photonics, Electronics and Telecommunications

**Tianhua Chen**
Doctoral Student of the Study Programme "Telecommunications"

# FINE-GRAINED MONITORING AND CLASSIFICATION OF INTERACTIVE MEDIA NETWORK TRAFFIC

**Summary of the Doctoral Thesis**

Scientific supervisors:

Associate Professor Dr. sc. ing.
ELANS GRABS

Associate Professor Dr. sc. ing.
ALEKSANDRS IPATOVS

Co-advisor:
Professor PhD
CANO BAÑOS, MARÍA DOLORES

RTU Press
Riga 2026

Chen, T. Fine-Grained Monitoring and Classification of Interactive Media Network Traffic. Summary of the Doctoral Thesis. Riga: RTU Press, 2026. 51 p.

Published in accordance with the decision of the Promotion Council "P-08" of 12 November 2025, Minutes No. 43.

This Thesis has been supported by the EU Recovery and Resilience Facility within the Project No. 5.2.1.1.i.0/2/24/I/CFLA/003 "Implementation of consolidation and management changes at Riga Technical University, Liepaja University, Rezekne Academy of Technology, Latvian Maritime Academy and Liepaja Maritime College for the progress towards excellence in higher education, science and innovation" (ID 1072).


Cover picture from ChatGPT.

https://doi.org/10.7250/9789934372544
ISBN 978-9934-37-254-4 (pdf)

# DOCTORAL THESIS PROPOSED TO RIGA TECHNICAL UNIVERSITY FOR PROMOTION TO THE SCIENTIFIC DEGREE OF DOCTOR OF SCIENCE

To be granted the scientific degree of Doctor of Science (PhD), the present Doctoral Thesis has been submitted for defence at the open meeting of RTU Promotion Council on 13 February 2026 at 11.00 AM at the Faculty of Computer Science, Information Technology and Energy of Riga Technical University, Āzenes Street 12, Room 201.

OFFICIAL REVIEWERS

Professor Dr. sc. ing. Sandis Spolītis
Riga Technical University

Professor Dr. sc. ing. Jaime Lloret
Technical University of Valencia, Spain

Professor Dr. sc. ing. Mario Alberto Montagut Climent
University of Valencia, Spain

## DECLARATION OF ACADEMIC INTEGRITY

I hereby declare that the Doctoral Thesis submitted for review to Riga Technical University for promotion to the scientific degree of Doctor of Science (PhD) is my own. I confirm that this Doctoral Thesis has not been submitted to any other university for promotion to a scientific degree.

Tianhua Chen ……. (signature)
Date: ……

The Doctoral Thesis is written in English and consists of four chapters, 16 figures, 23 tables, and 8 appendices. The total length is 142 pages, including appendices. The bibliography contains 157 references.

# CONTENTS

# LIST OF ABBREVIATIONS

| | |
|---|---|
| AAC | Advanced Audio Coding |
| ABR | Adaptive BitRate |
| AI | Artificial Intelligence |
| BiRNN | Bi-directional Recurrent Neural Network |
| CBR | Constant BitRate |
| CCDF | Complementary Cumulative Distribution Function |
| CDN | Content Delivery Network |
| CNN | Convolutional Neural Network |
| CV | Coefficient of Variation |
| DASH | Dynamic Adaptive Streaming over HTTP |
| Db | Daubechies |
| DBN | Deep Belief Network |
| DLSS | Deep Learning Super Sampling |
| DPI | Deep Packet Inspection |
| DTLS | Datagram Transport Layer Security |
| EBSNN | Extended Byte Segment Neural Network |
| ENN | Edited Nearest Neighbors |
| FPS | Frames Per Second |
| GCN | Graph Convolutional Network |
| GI | General Independent |
| GNN | Graph Neural Network |
| GOP | Group of Pictures |
| GRU | Gated Recurrent Unit |
| GSO | Generic Segmentation Offload |
| GTP-U | GPRS Tunnelling Protocol User Plane |
| HLS | HTTP Live Streaming |
| HTTP | Hyper Text Transfer Protocol |
| HTTP-FLV | HTTP FLV live stream |
| IAT | InterArrival Time |
| ICE | Interactive Connectivity Establishment |
| IDC | International Data Corporation |
| IEC | International Electrotechnical Commission |
| IP | Internet Protocol |
| ISO | International Organization for Standardization |
| ISP | Internet Service Providers |
| KNN | K Nearest Neighbors |
| LAN | Local Area Network |
| LSTM | Long Short-Term Memory |
| MAC | Media Access Control |
| MLP | Multi-Layer Perceptron |
| MPD | Media Presentation Description |
| MSE | Mean Squared Error |
| MTU | Maximum Transmission Unit |
| NAT | Network Address Translation |
| NGAP | Next Generation Application Protocol |
| NTC | Network Traffic Classification |
| NTMA | Network Traffic Monitoring and Analysis |
| OTT | Over The Top |
| PCAP | Packet Capture |
| PDF | Probability Density Function |
| QoE | Quality of Experience |
| QoS | Quality of Service |
| QUIC | Quick UDP Internet Protocol |
| RAN | Radio Access Network |
| RFC | Request for Comments |
| RLC | Radio Link Control |
| RNN | Recurrent Neural Network |
| RTCP | Real Time Control Protocol |
| RTMP | Real Time Messaging Protocol |
| RTP | Real-Time Protocol |
| SMOTE | Synthetic Minority Oversampling TEchnique |
| SNI | Server Name Indication |
| SRS | Simple Realtime Server |
| SRT | Secure Reliable Transport |
| SSIM | Structural Similarity Index |
| SSL | Secure Sockets Layer |
| STUN | Session Traversal Utilities for NAT |
| TCP | Transmission Control Protocol |
| TFSN | TLS Flow Sequence Network |
| TLS | Transport Layer Security |
| Tor | The onion router |
| TPR | True Positive Rate |
| TSO | TCP Segmentation Offload |
| TURN | Traversal Using Relays around NAT |
| UHD | Ultra High Definition |
| UPF | User Plane Function |
| VBR | Variable BitRate |
| VDS | Virtual Desktop Streaming |
| VLC | VideoLAN Client |
| VLS | Video Live Streaming |
| VoD | Video on Demand |
| VPN | Virtual Private Network |
| WebRTC | Web Real Time Communication |
| XR | eXtended Reality |

# LIST OF TABLES

# 1. OVERVIEW

## 1.1. Introduction

According to International Data Corporation (IDC), global Internet users are projected to generate 175 ZB of traffic data by 2025 [1]. Network traffic is a complex collection of interactions, data flows, and endpoints characterized by self-similarity, burstiness, and cyclical patterns. In the face of rapidly increasing data traffic, Network Traffic Monitoring and Analysis (NTMA) plays a crucial role in monitoring network performance and improving resource utilization [2], [3]. Network Traffic Classification (NTC) is one of the most crucial applications in the NTMA. NTC is the process of categorizing network traffic based on network characteristics to identify different applications or services running on the network [4]–[12]. This classification is essential for Internet Service Providers (ISPs) to manage their operations effectively. Categorizing traffic into priority levels helps ensure that critical applications receive adequate bandwidth and processing power. It also aids in anomaly detection and implementing measures to maintain network security and performance. Meanwhile, video-based services have become a major contributor to network traffic data [13]. According to Cisco, video media will account for 82 % of all IP traffic by 2021 [14]. Over-the-top (OTT) applications, such as Netflix and YouTube, dominate this growth, with Sandvine Incorporated reported a 24 % increase in video traffic as of January 2023 [15], [16]. Currently, many prevailing OTT platforms offer 4K/8K Ultra-High-Definition (UHD) resolution content with high frame rates like 60 FPS or 120 FPS, significantly impacting network traffic volume [17]–[24].

Most existing studies on media application traffic monitoring and classification focus primarily on encrypted OTT video applications [25]–[28]. Researchers have collected large volumes of network traffic and constructed several public benchmark datasets, including ISCXVPN2016, ISCXTor2016, MIRAGE-2019, CICDarknet2020, MAppGraph2021, and UTMobileNet2021 [29]–[34]. These applications are labeled based on features including application name, streaming platform, audio and video content, encapsulation, encryption, anonymization methods, transmission protocols, quality of service (QoS) levels, quality of experience (QoE), and other characteristics [35]–[41]. With the use of advanced machine learning and deep learning classifiers, these studies have achieved excellent classification performance at multiple granularity levels, including payload and flow analysis [42]–[86]. Although video-related traffic is present in these datasets, it usually represents only a small portion of the total data. Some proprietary datasets are specifically focused on video traffic, using classification labels derived from metadata such as playback behavior, and video parameters [87]–[94]. However, these datasets still have several limitations. There is often a weak correlation and explainability between the samples and their ground-truth labels, the classification boundaries remain unclear, and the data is frequently mixed with other types of application traffic. This overlap occurs across various dimensions, including network protocols, system architectures, application categories, and traffic distribution patterns, which makes the media application traffic profiles more complex [95], [96].

Additionally, emerging applications such as cloud gaming, social networking, web conferencing, and eXtended Reality (XR)/Metaverse experiences are evolving rapidly [97]–[99]. Most existing works focus on traffic modeling and analysis, while traffic monitoring and classification for emerging applications remain in the early stages of research [100]–[108]. However, these emerging applications are evolving rapidly, making it increasingly difficult to categorize network traffic using traditional methods based on applications or services. The growing convergence and interactivity among applications have led to diminishing distinctions in their traffic characteristics. To gain deeper insights into the traffic patterns of these next-generation applications, fine-grained monitoring and classification of interactive media network traffic have become essential. For OTT providers, it enables them to anticipate trends in interactive media network traffic, optimize network configurations, enhance performance, reduce security risks, and deliver better video services to users.

## 1.2. Rationale

To achieve fine-grained monitoring and classification of interactive media network traffic, a structured multi-step methodology is required. The first essential step involves comprehensive traffic data collection. To ensure diversity and representativeness, interactive media traffic is categorized into three major application domains: Live Streaming, Cloud Gaming, and Metaverse. The Live Streaming category includes traffic from five globally and regionally popular platforms: YouTube, BiliBili, Twitch, Facebook, and TikTok. The Cloud Gaming category consists of traffic captured from applications such as *Spitlings*, *Tomb Raider*, and *Thumper* [98]. The Metaverse category comprises applications including *VRChat, TheLab, SolarSystemAR, RealityMixer, Hellblade, DiRT Rally 2.0,* and *Bigscreen Theatre* [108]. All traffic sources are obtained from public and proprietary datasets and exhibit variation in network standards, capture durations, sniffing tools, and device platforms. Therefore, a critical preprocessing step is necessary to normalize and prepare the data for subsequent analysis.

The second step involves determining the appropriate number and structure of traffic categories to support fine-grained classification. Key factors influencing traffic characteristics, particularly for video-based applications, include resolution, bitrate, encoding format, container type, frame rate, streaming mode such as Video on Demand (VoD) or Video Live Streaming (VLS), and streaming protocols. These parameters, along with prior domain knowledge, help define meaningful and distinguishable traffic categories. For encrypted traffic, the classification granularity significantly affects both accuracy and efficiency. While many studies focus on flow-level and payload-level classification, time series-based approaches have become increasingly relevant. These methods utilize spatiotemporal patterns in traffic data without relying on payload inspection, making them well suited for privacy-preserving classification tasks. In the fourth step, datasets are constructed based on the selected classification granularities. The final step involves selecting relevant training features and applying supervised learning algorithms to perform the classification task. Model performance is then evaluated using standard classification metrics to assess the effectiveness of fine-grained monitoring and classification of interactive media network traffic [13], [2], [3].

## 1.3. The Aim and Theses of the Doctoral Thesis

The objective of this Doctoral Thesis is to achieve fine-grained monitoring and classification of network traffic generated by interactive media applications across diverse network conditions at multi-granularity levels. This study aims to improve the understanding of traffic patterns and the roles of interactive media in today's complex and heterogeneous network environments, while also contributing to traffic behavior analysis in NTC and extending to NTMA tasks. The core findings are summarized as follows:

Thesis 1: Network bandwidth is the primary factor affecting interactive media network traffic volume and distribution. Additionally, video resolution and frame rates also play a direct role in shaping traffic characteristics. Variable bitrate encoding, fluctuations in keyframe bitrate, and the utilization of GOP structures are among the most critical factors affecting traffic behaviors.

Thesis 2: Streaming application protocols can dynamically pair with different transport layer protocols. When WebRTC is combined with UDP, it produces the highest number of packets but with the smallest payload size, where the traffic volume accounts for only 70 % of the corresponding raw video file size, while also showing the lowest latency. When HLS is combined with TCP, it yields the largest total payload volume, up to 1.5 times larger than the raw video file under the same network conditions, and exhibits a high control-plane to data-plane traffic ratio.

Thesis 3: In fine-grained classification tasks, ground truth labels are directly associated with sample characteristics. These labels are derived from traffic-influencing factors and other prior knowledge. All training features and samples are transparent and interpretable. The classification task is conducted at three granularities – time series, flow, and payload – making the approach adaptable to different scenarios.

Thesis 4: The Stacking model achieved over 99 % overall classification accuracy across 86 categories in the Live Streaming dataset, containing more than 200,000 samples. Each category attained an F1-score of at least 0.94 at the flow level. The 1D-CNN model achieved at least 80 % classification accuracy at both the time series and flow levels, outperforming three state-of-the-art deep learning models. At the payload level, the 2D-CNN model reached over 84 % classification accuracy, exceeding other methods by more than 10 %.

The research hypotheses of the Doctoral Thesis:

1. Different interactive media applications generate similar traffic volumes under the same network bandwidth. Variations in raw video parameters result in traffic volume differences of no more than 10 % among all applications. Setting the GOP to 180 frames best reflects the actual traffic burstiness distribution over time.

2. WebRTC, combined with UDP, transmits the smallest payload volume, achieving over 30 % traffic savings compared to the original video size. It also generates at least twice as many packets as other streaming protocols, including HLS, DASH, RTMP, and HTTP-FLV.

3. Labels in fine-grained classification correlate with known traffic generation parameters, allowing for transparent and explainable class mapping. At the payload level, the total number

of interactive media traffic samples exceeds that of the time-series and flow levels by at least 70 %, even without resampling techniques.

4. The 2D-CNN model achieves over 84 % classification accuracy at the payload level, outperforming other baselines by at least 10 %.

## 1.4. The Tasks of the Doctoral Thesis

1. To collect interactive media application network traffic from multiple sources, including public and proprietary datasets. Traffic was actively captured from various platforms, applications, and services under different network conditions. Bidirectional traffic from both client and server sides was obtained in idealized network environments across multiple streaming protocols.

2. To use the standard video quality testing file, Big Buck Bunny, for client-server push and pull stream tests. To generate video files with varying quality levels using the `FFmpeg` video transcoding tool and examine the relationships between bitrate and frame distribution.

3. To conduct global and local statistical analysis of interactive media network traffic. To employ a sliding time window method to analyze traffic distributions across varying time scales and identify patterns and behaviors.

4. To develop algorithms to generate classification datasets for interactive media applications network traffic at three levels of granularity: time series, flow, and payload. To apply preprocessing to construct the corresponding datasets.

5. To design and evaluate classification models including 1D-CNN, 2D-CNN, RNN, GCN, and stacking. To compare their performance with three state-of-the-art models: Deep Packet, FlowPic, and FS-Net.

## 1.5. Research Methods

To carry out the tasks outlined in the Doctoral Thesis and analyze the associated problems, `Wireshark`, `Tcpdump`, and `nDPI` tools were employed to collect diverse network traffic. Preprocessing was conducted using `MATLAB`, `NFStream`, and `CICFlowMeter` to generate proprietary datasets, which were also compared with various public benchmark datasets. For mathematical modeling, multiple state-of-the-art machine learning algorithms were applied while using popular Python libraries, including `PyTorch`, `PyTorch Geometric`, `OpenCV`, `SciPy`, `StellarGraph`, `TensorFlow`, `Keras`, `XGBoost`, `scikit-image` and `scikit-learn` for network traffic classification. Additionally, `srsRAN`, `Simu5G`, and `SRS` tools were utilized to simulate network traffic under specific scenarios.

The core experiments of the Doctoral Thesis research were conducted in the Open RAN Testing and Research Laboratory at the Institute of Photonics, Electronics and Telecommunications (IPEC) of Riga Technical University (RTU). An additional experimental study was carried out at the Department of Information and Communication Technologies, Technical University of Cartagena (UPCT) in Spain.

## 1.6. Scientific Novelty and Main Results

The scientific novelty of the Doctoral Thesis lies in the pioneering work on fine-grained monitoring and classification of network traffic generated by mainstream interactive media applications under heterogeneous network environments. The study achieves high classification performance on multiple metrics with transparent and interpretable processes. The following are the specific novel contributions, supported by 8 publications

1. A robust stacking ensemble learning classifier is proposed that achieved 99 % classification accuracy across 38 categories and an F1-macro score of 0.83 across 101 categories at the flow level on public and proprietary datasets. This model effectively handles datasets with varying sample sizes and severe category imbalance. The Stacking model, introduced in Publication 1, is implemented in the Classification Models section of the Methodology chapter.

2. In the 5G Open RAN environment, it was demonstrated how 5G frequency bands and bandwidth conditions influence video applications traffic distribution. Over 99 % classification accuracy was achieved across 13 categories at the flow level under the enhanced Mobile BroadBand slicing scenario. The corresponding public datasets referenced in Publication 2 are utilised in the Data Acquisition and Preprocessing sections of the Methodology chapter.

3. A novel and explainable *nYFTQC* algorithm has been developed, enabling the construction of 15 proprietary Internet video traffic datasets. These datasets are notable for their completeness, consistency, and transparency. The novel *nYFTQC* algorithm, presented in Publication 3, is applied in the Methodology chapter, in the section on the generation of multi-granularity classification datasets at the flow level.

4. A GCN-based deep learning model is proposed that achieved 86 % classification accuracy across 10 categories at the flow level, incorporating both real-world OTT applications generated network traffic and simulated video traffic using DASH. The GCN-based deep learning model, introduced in Publication 4, is used in the Methodology chapter, in the Classification Models section for performance comparison. Additionally, Publication 4 applies the DASH-based simulated video streaming method in the data acquisition and preprocessing section to generate the streaming VLS simulation datasets.

5. In Publication 5, an improved GNN model was designed for a mixed OTT and XR traffic dataset. Samples were labelled into three QoE-based categories using the Mean Opinion Scores metric, achieving approximately 90 % classification accuracy at the payload level under Wi-Fi network conditions. Publication 5 introduced the mixed traffic dataset generation method, which is applied in the Methodology chapter, in the Data Acquisition and Preprocessing section, for constructing metaverse and cloud gaming datasets. The payload-level classification method, involving byte payload graph sample construction, is also applied in the Methodology chapter, section on the generation of multi-granularity classification datasets for generating payload-level interactive media network traffic datasets.

6. A generative DBN model has been introduced, reaching 92 % classification accuracy across 8 categories at the time series level, where normal OTT applications generated network traffic under the Wi-Fi network environment was mixed with Tor anonymization and VPN

encryption. The same model achieved 94 % accuracy at the flow level on the same dataset. For Publication 6, Tor and VPN-related public datasets were utilized. After data cleaning, these were applied in the Methodology chapter in the Data Acquisition and Preprocessing section to generate the new Tor and VPN sub-dataset within the live streaming dataset.

7. A 1D-CNN model is proposed that achieved 97 % classification accuracy across four categories for the YouTube dataset at the time series level. Based on this model, modified 2D-CNN and RNN models are also developed to expand performance across other granularities. Publication 7 introduced the 1D-CNN model, which was utilized in the Methodology chapter the Classification Models section, for further classification performance comparison.

8. Statistical moment analysis and Fast Fourier Transformationwere applied to address packet burstiness and distribution bias at the time series level. Publication 8 introduced methods that improved interpretability but yielded only limited enhancements in classification performance. These methods were still employed to test interactive media network traffic classification performance at the time-series level, as presented in the Generation of Multi-granularity Classification Datasets section of the Methodology chapter.

According to classification results across three granularities for datasets composed of interactive media application network traffic, the main results are as follows:

1. The 1D-CNN model achieved up to 95 % classification accuracy and a macro F1-score of 0.93 among 95 categories at the packet time series level on the Live Streaming dataset with over 40,000 samples.

2. The Stacking model achieved up to 99 % classification accuracy and a macro F1-score of 0.99 among 86 categories at the flow level on the Live Streaming dataset with over 200,000 samples.

3. The 2D-CNN model achieved up to 81 % classification accuracy and a macro F1-score of 0.82 among 95 categories at the payload level on the Live Streaming dataset with over 180,000 samples.

The results achieved during the Doctoral Thesis development have been applied in the following projects:

1. ERDF within the Activity 1.1.1.2 “Postdoctoral Research Aid” of the Specific Aid Objective 1.1.1, No. 1.1.1.2/VIAA/2/18/332, 01.09.2021-30.11.2021.
2. ESF doctoral and academic staff specialization strategic strengthening grant, No. 8.2.2.0/20/I/008, 01.12.2022-30.11.2023.
3. European Union's Recovery and Resilience Facility within the Project No. 5.2.1.1.i.0/2/24/I/CFLA/003, 01.10.2024-30.09.2025.
4. Multipath Roaming Solution for Open RAN (TRANSFER), project No. 1.1.1.3/1/24/A/018, 24.04.2025-30.09.2025.
5. Grant PID2023-148214OB-C21 funded by MICIU/AEI/10.13039/501100011033 and FEDER/EU (Murcia, Spain), 01.05.2023-31.12.2024.

## 1.7. The Structure of the Doctoral Thesis

The Doctoral Thesis comprises four main chapters. Chapter 1 presents the research context of fine-grained monitoring and classification of interactive media network traffic. It outlines the background, motivation, literature review, objectives, aims, theses, and hypotheses. Chapter 2 describes the methodologies employed to achieve the research goals. It includes five subsections. Chapter 3 provides a detailed analysis of the classification results at the three granularity levels and their corresponding datasets. Chapter 4 concludes the Doctoral Thesis and discusses future research directions.

## 1.8. Publications and Approbation of the Doctoral Thesis

The results of the Doctoral Thesis are presented in **five** scientific articles and publications in conference proceedings, **four** of which are indexed in SCOPUS, Web of Science (WoS), and Institute of Electrical and Electronics Engineers (IEEE) databases. The author has a total of **eight** publications. The main results of the Doctoral Thesis were summarized in **three** scientific journals. The results of the research were presented at **eight** international scientific conferences.

Scientific articles and publications

1. **Chen, T.**, Grabs, E., Ipatovs, A., Cano, M. "*Ensemble Learning Enabled Flow-level Internet Traffic Classification*" submitted to Journal of Information and Telecommunication (Under review).
2. **Chen, T.**, Benkis, R., Bogdanovs, N., Stetjuha, M., Klūga, J., Jeralovičs, V., Rjazanovs, D., Grabs, E., Ipatovs, A. "*Video Traffic Classification in the 5G Open RAN Network*" submitted to 2025 Photonics & Electromagnetics Research Symposium (PIERS 2025) Conference (Accepted), United Arab Emirates, Abu Dhabi, 4–8 May **2025**.
3. **Chen, T.**, Grabs, E., Ipatovs, A., Cano, M. *A Novel Proprietary Internet Video Traffic Dataset Generation Algorithm*. Applied Sciences, Vol. 15, No. 2, Article number 515. e-ISSN 2076-3417, **2025**.
4. **Chen, T.**, Grabs, E., Ipatovs, A., Pētersons, E., Ancāns, A. *Bitrate-based Video Traffic Classification*. In: 2023 PhotonIcs & Electromagnetics Research Symposium (PIERS 2023): Proceedings, Czech Republic, Prague, 3–6 July **2023**.
5. **Chen, T.**, Grabs, E., Tasic, I., Cano, M. *Network Traffic Analysis for eXtended Reality Applications*. In: XVI Telematics Engineering Conference (JITEL 2023), Spain, Barcelona, 8–10 November **2023**.
6. **Chen, T.**, Grabs, E., Pētersons, E., Efrosinin, D., Ipatovs, A., Bogdanovs, N., Rjazanovs, D. *Multiclass Live Streaming Video Quality Classification Based on Convolutional Neural Networks*. Automatic Control and Computer Sciences, Vol. 54, No. 5, pp. 455–466. ISSN 0146-4116. e-ISSN 1558-108X, **2022**.
7. **Chen, T.**, Grabs, E., Pētersons, E., Efrosinin, D., Ipatovs, A., Klūga, J. *Encapsulated and Anonymized Network Video Traffic Classification with Generative Models*. In: 2022

Workshop on Microwave Theory and Techniques in Wireless Communications 2022 (MTTW'22): Proceedings, Latvia, Riga, 5–7 October **2022**.

8. Grabs, E., **Chen, T.**, Pētersons, E., Efrosinin, D., Ipatovs, A., Klūga, J., Čulkovs, D. *Features Extraction for Live Streaming Video Classification with Deep and Convolutional Neural Networks*. In: 2021 IEEE Microwave Theory and Techniques in Wireless Communications (MTTW 2021), Latvia, Riga, 7–8 October, **2021**.

Presentations at **8** international scientific conferences

1. **Chen, T.**, Benkis, R., Bogdanovs, N., Stetjuha, M., Klūga, J., Jeralovičs, V., Rjazanovs, D., Grabs, E., Ipatovs, A. "*Video Traffic Classification in the 5G Open RAN Network*" submitted to 2025 Photonics & Electromagnetics Research Symposium (PIERS 2025) Conference (Accepted), United Arab Emirates, Abu Dhabi, 4–8 May **2025**.
2. **Chen, T.**, Grabs, E., Tasic, I., Cano, M. *Network Traffic Analysis for eXtended Reality Applications*. In: XVI Telematics Engineering Conference (JITEL 2023), Spain, Barcelona, 8–10 November **2023**.
3. **Chen, T.**, Grabs, E., Ipatovs, A. *Artificial Intelligence Application in Network Traffic Engineering*, 63rd International Scientific Conference of RTU, Latvia, Riga,17 October **2023**.
4. **Chen, T.**, Grabs, E., Ipatovs, A., Pētersons, E., Ancāns, A. *Bitrate-based Video Traffic Classification*. In: 2023 PhotonIcs & Electromagnetics Research Symposium (PIERS 2023): Proceedings, Czech Republic, Prague, 3–6 July **2023**.
5. **Chen, T.**, Grabs, E., Ipatovs, A. *Artificial Intelligence Application in Network Traffic Engineering*, 64th International Scientific Conference of RTU, Latvia, Riga, 14 October **2022**.
6. **Chen, T.**, Grabs, E., Pētersons, E., Efrosinin, D., Ipatovs, A., Klūga, J. *Encapsulated and Anonymized Network Video Traffic Classification with Generative Models*. In: 2022 Workshop on Microwave Theory and Techniques in Wireless Communications 2022 (MTTW'22): Proceedings, Latvia, Riga, 5–7 October **2022**.
7. Grabs, E., **Chen, T.**, Ipatovs, A. *Flow-based Video Quality Traffic Classification for Real-Time SDN Applications*, First Workshop for ERI on Telecommunication and Networks, Romania, Cluj-Napoca, 14–15 March **2022**.
8. Grabs, E., **Chen, T.**, Pētersons, E., Efrosinin, D., Ipatovs, A., Klūga, J., Čulkovs, D. *Features Extraction for Live Streaming Video Classification with Deep and Convolutional Neural Networks*. In: 2021 IEEE Microwave Theory and Techniques in Wireless Communications (MTTW 2021), Latvia, Riga, 7–8 October **2021**.

# 2. METHODOLOGY

In this chapter, all employed methods are systematically introduced across five main sections.

## 2.1. Data Acquisition and Preprocessing

Data acquisition and preprocessing are essential steps in constructing standardized training datasets for interactive media applications network traffic analysis. Table 1 summarizes prior knowledge related to network traffic in interactive media applications, consisting of four datasets: Live Streaming, Cloud Gaming, Metaverse, and Streaming VLS Simulation. Each dataset includes multiple sub-datasets. For the Live Streaming dataset, each proprietary sub-dataset corresponds to a specific OTT platform. Traffic data was captured while watching a single video on each platform using *Google Chrome version 100.0.4896.88*. The traffic was recorded as Packet Capture (PCAP) files via *Wireshark 3.4.3* on a *Windows 10 system (Build 18363.1316)*, using an *Intel(R) Wi-Fi 6 AX200 160MHz* adapter in monitor mode. Captured PCAP files were validated using Internet Protocol (IP) and Media Access Control (MAC) addresses, filtered, and truncated to 30 minutes to ensure consistent duration across services. Sub-datasets are categorized by platform name, video resolution, frames per second (FPS), and whether the content was VoD or VLS. Over 84 GB of traffic was collected.

The YouTube sub-dataset exhibited the greatest category diversity. Different OTT platforms recommend various video encoding settings. Most platforms use adaptive bitrate (ABR) streaming to adjust video quality dynamically based on network conditions and device capabilities [109], [20]. During data capture, fixed playback resolutions and latency mode were selected, Wi-Fi connection bandwidth reached over 100 Mbps. However, many platforms, such as YouTube and Vimeo, favor variable bitrate (VBR) encoding, which allows the bitrate to fluctuate for optimal quality [110]–[112]. The usage of VBR encoding is influenced by several factors, including resolution height and width, frame rate, color depth, codec, encoding settings, scene complexity, and motion intensity. As these parameters increase, the required bitrate tends to rise accordingly. In contrast, platforms like Twitch, Facebook, TikTok, and BiliBili often prefer constant bitrate (CBR) encoding [113]–[115]. For VoD applications, downloaded video files can be analyzed to retrieve raw bitrate information. In contrast, for VLS, such data must be inferred indirectly using real-time traffic statistics. This approach is affected by factors such as Content Delivery Network (CDN) cache delays, storage location of video resources, and varying network conditions, including bandwidth, jitter, and packet loss [116].

In the context of 5G networks, traffic characteristics have changed significantly due to different cellular network configuration variances. According to the *5G Coverage Expansion Dataset 1*, *Italian in-lab testbed dataset 1* and *Italian in-lab testbed dataset 2* from the *NANCY* project, video traffic was simulated and captured using `srsRAN` and `PCAPdroid` tools in the *Open RAN* environment. Data was collected across multiple layers of the protocol stack, including MAC, Next Generation Application Protocol (NGAP), Radio Link Control (RLC), and GPRS Tunneling Protocol User Plane (GTP-U), as well as on the edge client side. Different

video files were streamed using various resolutions (480 P, 720 P, 1080 P, 2160 P, and 4320 P), bitrates (6 Mbps, 10 Mbps, 20 Mbps, and 40 Mbps), and bandwidths (10 MHz and 20 MHz) on two frequency bands (*N3* and *N78*) via Real Time Messaging Protocol (RTMP). The resulting constant bitrates list is approximated through rate limiting under various network bandwidth conditions. However, the theoretical maximum transmission rate in 5G networks significantly exceeds the bitrates discussed in this context. Encoding was performed using H.264 and VP8 codecs. These traffic traces are consolidated into the *Big Buck Bunny* sub-dataset under the Live Streaming category [117]–[119].

Besides, even within the same network environment, network traffic distribution varies across different ABR protocols. This variation is influenced by differences in packet buffering policies, latency tolerance, device capabilities, and user behavior. Platforms such as Twitch, Facebook, and TikTok primarily use the HTTP Live Streaming (HLS) protocol. In contrast, BiliBili and Netflix mainly adopt Dynamic Adaptive Streaming over HTTP (DASH), while YouTube and Vimeo support both protocols. HLS was defined by RFC 8216, which enables segmented video transmission. Clients retrieve and play video content through *.m3u8* playlist files and *.ts* segment files [120]. It typically uses the H.264 video codec and supports audio codecs such as AAC, MP3, and AC-3. DASH, standardized by ISO/IEC 23009-1:2022, also supports segmented transmission. Clients use the Media Presentation Description (MPD) playlist file to access *.ts* or *.m4s* segment files. DASH supports a broader range of video codecs, including H.265, H.264, and VP9 [121]. Although many parameters of a video source (e.g., encoding format, resolution, bitrate, framerate, protocol) can be customized before streaming, these details are typically not visible to the client. Each playback instance on an OTT platform exhibits a unique traffic fingerprint based on these hidden parameters. Consequently, traffic characteristics in interactive media applications vary significantly depending on the chosen protocol, platform, and playback scenario.

To eliminate the influence of external environmental parameters on the network traffic distribution of interactive media applications, the characteristics of traffic under different streaming protocols are systematically examined. All servers and clients are configured within the same Local Area Network (LAN) environment to ensure consistent testing conditions. A single Big Buck Bunny video file is used repeatedly in multiple VLS sessions, with different ABR protocols applied. During each session, downstream traffic is captured from the client side, while upstream traffic is recorded from the Simple Realtime Server (`SRS`) using the `Tcpdump` tool [122]. The bidirectional traffic is selected as an individual category. Big Buck Bunny serves as a widely accepted benchmark for video playback testing due to the availability of high-quality raw video formats that support flexible preprocessing for experimental needs [123]. The `SRS` server is deployed in a `Docker` environment and supports several streaming protocols, including RTMP, Web Real-Time Communication (WebRTC), HLS, HTTP FLV live stream (HTTP-FLV), Secure Reliable Transport (SRT), DASH, and GB28181. For this study, five protocols relevant to interactive media applications are selected: RTMP, WebRTC, HLS, HTTP-FLV, and DASH. RTMP is used as the stream pushing protocol, while `FFmpeg` is employed for video editing, transcoding, and segment generation [124]. The duration of each media segment is set to two seconds, and 10 fragments are retained in the sliding window for

each session. All five selected protocols are supported on the client side using playback tools such as `SRS Player`, `VLC`, `Dash JavaScript Player` and `FFplay` [125], [126]. The resulting network traffic from these streaming sessions is compiled into the Streaming VLS Simulation dataset, with a total volume of approximately 20 GB.

The Cloud Gaming and Metaverse datasets exhibit substantial differences in their data characteristics. The Cloud Gaming dataset includes three sub-datasets: *Spitlings* (SP), *Tomb Raider* (TR), and *Thumper* (TH), all of which were captured on *Google's Stadia* platform. Stadia delivers its services through the WebRTC framework, utilizing several protocols including ICE, STUN, TURN, DTLS, RTP, and RTCP [127]. ICE (Interactive Connectivity Establishment) facilitates peer-to-peer communication over UDP in NAT environments by incorporating STUN and TURN. DTLS (Datagram Transport Layer Security) secures datagram-based transmissions. RTP (Real-Time Protocol) is used for media delivery from the server, while RTCP (Real-Time Control Protocol) enables client-side feedback on reception quality. Network traffic was collected under a VBR scenario with no bandwidth limitations. Each sub-dataset is categorized by game title, resolution, codec, game state, and streaming protocol. Due to the unavailability of raw PCAP files, detailed flow-level and payload-level analysis is not feasible [98].

The Metaverse datasets consist of seven distinct services, with network traffic captured using a *virtual desktop streaming* (VDS) configuration. A cloud-based computer served as the rendering platform, hosting the VDS server. An *Oculus Quest 2* device was connected via a VDS client, and traffic was recorded using `Wireshark`. Each session was limited to 15 Mbps bandwidth and had an approximate duration of one minute. Similar to the Cloud Gaming dataset, Metaverse applications also rely on the WebRTC protocol for streaming [108]. In addition to video-related metadata, OTT applications often implement anonymization and encryption techniques. To reflect these characteristics, YouTube and Vimeo traffic samples, which combine Tor anonymization browser and OpenVPN protocol encapsulation encryption, are extracted from the ISCXVPN2016 and ISCXTor2016 public datasets, respectively [29], [30]. The resulting dataset includes four categories; however, it lacks detailed metadata descriptions related to video parameters. To summarize, the network traffic of various interactive media applications has been analyzed based on protocol diversity, video metadata, and network environment characteristics.

Table 1
Summary of Variations in Network Traffic Characteristics of Interactive Media Applications

| Datasets | Dataset | Category | Raw PCAP File size | Duration | Network | Source |
|---|---|---|---|---|---|---|
| Live Streaming | YouTube | 26 | 44.8 GB | 1800 s | Wi-Fi | Proprietary |
| | Big_Buck_Bunny | 13 | 10.4 GB | 600 s | 5G | Public [117], [119] |
| | BiliBili | 15 | 20.2 GB | 1800 s | Wi-Fi | Proprietary |
| | Twitch | 14 | 8.48 GB | 1800 s | Wi-Fi | Proprietary |
| | Facebook | 12 | 3.78 GB | 1800 s | Wi-Fi | Proprietary |
| | TikTok | 6 | 2.98 GB | 1800 s | Wi-Fi | Proprietary |
| | Vimeo | 5 | 1.84 GB | 1800 s | Wi-Fi | Proprietary |
| | Tor&VPN | 4 | 1.95 GB | 945s | Wi-Fi | Public [29], [30] |
| Cloud Gaming | Spitlings | 6 | / | 600 s | Wi-Fi | Public [98] |
| | Tomb Raider | 5 | / | 600 s | Wi-Fi | Public [98] |
| | Thumper | 1 | / | 600 s | Wi-Fi | Public [98] |
| Metaverse | VRChat | 3 | 338 MB | 63 s | Wi-Fi | Public [108] |
| | TheLab | 3 | 348 MB | 63 s | Wi-Fi | Public [108] |
| | SolarSystemAR | 3 | 345 MB | 63 s | Wi-Fi | Public [108] |
| | RealityMixer | 3 | 350 MB | 63 s | Wi-Fi | Public [108] |
| | Hellblade | 3 | 347 MB | 63 s | Wi-Fi | Public [108] |
| | DiRTRally2.0 | 3 | 358 MB | 63 s | Wi-Fi | Public [108] |
| | BigscreenTheatre | 3 | 357 MB | 63 s | Wi-Fi | Public [108] |
| Streaming VLS Simulation | Big_Buck_Bunny | 13 | 4.14 GB | 635 s | Ethernet | Proprietary |
| | Big_Buck_Bunny | 13 | 3.23 GB | 635 s | Ethernet | Proprietary |
| | Big_Buck_Bunny | 13 | 3.17 GB | 635 s | Ethernet | Proprietary |
| | Big_Buck_Bunny | 13 | 3.22 GB | 635 s | Ethernet | Proprietary |
| | Big_Buck_Bunny | 13 | 2.81 GB | 635 s | Ethernet | Proprietary |
| | Big_Buck_Bunny | 13 | 3.23 GB | 635 s | Ethernet | Proprietary |

| Datasets | Dataset | Resolution | Frame rates | Streaming protocol | Bitrate |
|---|---|---|---|---|---|
| Live Streaming | YouTube | 144p–8K | 30; 60 | HLS/DASH | VBR/CBR |
| | Big_Buck_Bunny | 480p–8K | / | HLS/DASH | 6;10;20;40Mbps |
| | BiliBili | 360p–8K | 30; 60; 80; 120 | DASH | VBR/CBR |
| | Twitch | 160p–1080p | 30; 60 | HLS | VBR/CBR |
| | Facebook | 144p–1080p | 30 | HLS | VBR/CBR |
| | TikTok | 360p–1080p | 30; 60 | HLS | VBR/CBR |
| | Vimeo | / | / | HLS/DASH | VBR/CBR |
| | Tor&VPN | / | / | HLS/DASH | VBR/CBR |
| Cloud Gaming | Spitlings | 720p–4K | / | WebRTC | VBR |
| | Tomb Raider | 720p–4K | / | WebRTC | VBR |
| | Thumper | 1080p | / | WebRTC | VBR |
| Metaverse | VRChat | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | TheLab | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | SolarSystemAR | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | RealityMixer | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | Hellblade | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | DiRTRally2.0 | / | 60; 90; 120 | WebRTC | 15 Mbps |
| | BigscreenTheatre | / | 60; 90; 120 | WebRTC | 15 Mbps |
| Streaming VLS Simulation | Big_Buck_Bunny | 144p–4K | 30; 60 | HLS | VBR |
| | Big_Buck_Bunny | 144p–4K | 30; 60 | DASH | VBR |
| | Big_Buck_Bunny | 144p–4K | 30; 60 | RTMP | VBR |
| | Big_Buck_Bunny | 144p–4K | 30; 60 | HTTP_FLV | VBR |
| | Big_Buck_Bunny | 144p–4K | 30; 60 | WebRTC | VBR |
| | Big_Buck_Bunny | 144p–4K | 30; 60 | RTMP_Upload | VBR |

## 2.2. Classification Models

Currently, many state-of-the-art machine learning and deep learning classifiers have been widely adopted for NTC tasks, demonstrating high and robust performance across diverse datasets [13]. Previous studies, including Publications 1, 2, 4, and 7, have employed a variety of models, including Multi-Layer Perceptron (MLP), Convolutional Neural Networks (CNNs), Deep Belief Networks (DBNs), Extra Trees, Graph Neural Networks (GNNs), Graph Convolutional Networks (GCNs), and XGBoost [147]–[149], [105], [57], [51], [49], [52].

These models can be broadly categorized into two groups: deep learning models and ensemble learning models. Among deep learning architectures, CNNs are among the most extensively applied for classification. Publications 7 and 8 introduced the 1D-CNN model comprising three one-dimensional convolutional layers followed by four fully connected layers, each with over 1000 neurons. The final layer uses a SoftMax activation function for multi-class classification. A notable aspect of this architecture is the use of progressively smaller kernel sizes (5, 3, and 1) across the convolutional layers, which facilitates the extraction of fine-grained local patterns in sequential data. This design is particularly effective for time series-level and flow-level classification tasks. The corresponding computational process is presented in Equation (1).

$$y_j[t] = \sum_{i=1}^{C_{\text{input}}} \sum_{k=0}^{K-1} x_i[t \times S + k - P] \times W_{j,i,k} + b_j, \quad (1)$$

where

$y_j[t]$ – output at time step $t$ for output channel $j$;

$x_i[\cdot]$ – input from channel $i$;

$W_{j,i,k}$ – weight connecting input channel $i$ to output channel $j$ at kernel index $k$;

$b_j$ – bias term for output channel $j$;

$C_{\text{input}}$ – the number of input channels;

$K$ – kernel size;

$S$ – stride;

$P$ – padding.

This architecture was further extended into a 2D-CNN variant by replacing the one-dimensional convolutional layers with two-dimensional counterparts while retaining the overall structure. This adaptation enables the model to exploit spatial feature hierarchies through local receptive fields, offering benefits such as parameter efficiency, translation invariance, and weight sharing. The 2D-CNN is particularly well-suited for payload-level graph classification, where input data is represented in spatial or matrix form. The associated convolutional operation is defined in Equation (2) and shares conceptual similarity with the 1D-CNN.

$$y_j(h,w) = \sum_{i=1}^{C_{input}} \sum_{m=0}^{K_h-1} \sum_{n=0}^{K_w-1} x_i\,[h \times S_h + m - P_h, w \times S_w + n - P_w] \times W_{j,i,m,n} + b_j, \quad (2)$$

where

$y_j(h,w)$ – output value at spatial position $(h,w)$ in output channel $j$;

$x_i[\cdot\cdot]$ – input feature map from channel $i$ with spatial dimensions $(h,w)$;

$W_{j,i,m,n}$ – weight of the kernel at position $(m,n)$ for input channel $i$ and output channel $j$;

$K_h, K_w$ – the height and width of the convolution kernel;

$S_h, S_w$ – the stride along the height and width dimensions;

$P_h, P_w$ – padding along the height and width dimensions;

$C_{\text{input}}$ – the number of input channels;

$b_j$ is bias term for output channel $j$.

The 2D-CNN model is modified based on the 1D-CNN model as presented in Table 2.

Table 2
The Architecture Description of the 2D-CNN Model

| Layer | Type | Kernel size | Filters | Stride | Padding | Input size | Output size |
|---|---|---|---|---|---|---|---|
| Input | - | - | - | - | - | (1, 100, 100) | (1, 100, 100) |
| Conv1 | Conv2D | (5, 5) | 32 | 1 | 2 | (1, 100, 100) | (32, 100, 100) |
| Pool1 | MaxPool2D | (2, 2) | - | 2 | 0 | (32, 100, 100) | (32, 50, 50) |
| Conv2 | Conv2D | (3, 3) | 64 | 1 | 1 | (32, 50, 50) | (64, 50, 50) |
| Pool2 | MaxPool2D | (2, 2) | - | 2 | 0 | (64, 50, 50) | (64, 25, 25) |
| Conv3 | Conv2D | (1, 1) | 128 | 1 | 0 | (64, 25, 25) | (128, 25, 25) |
| Flatten | Flatten | - | - | - | - | (128, 25, 25) | (80000) |
| FC1 | Linear | - | 1000 | - | - | (80000) | (1000) |
| FC2 | Linear | - | 500 | - | - | (1000) | (500) |
| FC3 | Linear | - | 250 | - | - | (500) | (250) |
| FC4 | Linear | - | 125 | - | - | (250) | (125) |
| Output | Linear | - | $C$ | - | - | (125) | ($C$) |

In addition, an RNN model was developed using three stacked recurrent layers, which were based on the 1D-CNN model. RNNs are effective for modeling temporal dependencies by maintaining internal states over time. Although less complex than architectures such as LSTMs or Transformers, RNNs remain efficient for capturing short-term dependencies in sequential data. The underlying computations are described in Equation (3).

$$h_t = \mathrm{Tanh}(W_{ih} \times x_t + b_{ih} + W_{hh} \times h_{t-1} + b_{hh}), \quad (3)$$

where

$h_t$ in $R^H$ – the hidden state at time step $t$;

$\boldsymbol{x}_t$ in $R^D$ – the input vector at time step $t$;

$W_{ih}$ in $R^{H \times D}$ – the input to the hidden weight matrix;

$W_{hh}$ in $R^{H \times H}$ – the hidden to the hidden weight matrix;

$b_{ih}$ and $b_{hh}$ in $R^H$ – the respective bias vectors;

$\mathrm{Tanh}(\cdot)$ – element-wise hyperbolic tangent function.

In the flow-level classification task, a single flow-level individual feature has relation connections to other statistical or contextual features derived; it could be constructed as graph-structured data, and a GCN model was used. In this approach, a feature dimension from the dataset is used to construct a graph via the k-Nearest Neighbors (k-NN) algorithm, where each node represents a sample and edges reflect neighborhood relationships. This transforms the classification task into a node-level problem. GCNs offer advantages such as the ability to learn from non-Euclidean structures, aggregate information from local neighborhoods, and deliver strong performance in both node and graph classification tasks. Computational formulation is provided in Equation (4).

$$H^{(l+1)} = \mathrm{ReLU}\left(\widetilde{D}^{-\frac{1}{2}}\tilde{A}\widetilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)}\right), \quad (4)$$

where $H^{(l)}$ in $R^{N\times F^{(l)}}$ represents node feature matrix at layer $l$; $W^{(l)}$ in $R^{F^{(l)}\times F^{(l+1)}}$ represents the trainable weight matrix; $\tilde{A} = A + I$ represents adjacency matrix with self-loops; $\widetilde{D}$ is the diagonal degree matrix of $\tilde{A}$; and $ReLU(\cdot)$ represents rectifier activation function.

For classification performance comparison, three benchmark models were selected: Deep Packet, FlowPic, and FS-Net [150]–[152]. Both Deep Packet and FlowPic are designed based on 1D-CNN architectures, while FS-Net employs the Gated Recurrent Unit (GRU), an enhanced recurrent model that incorporates update and reset gates to manage information flow and address the vanishing gradient problem. To further enhance generalization capability, ensemble learning methods were employed. The stacking model includes eleven heterogeneous base learners comprising various boosting and bagging techniques. The final classification output is generated using logistic regression, leveraging the diverse strengths of individual classifiers to improve overall performance as introduced in Publication 1.

Table 4 summarizes the computational complexity of all classification models evaluated. The Stacking model has the highest complexity due to the combined effect of multiple base learners, each involving tree-based computations, along with the overhead introduced by parallel training across all sub-models. The 2D-CNN has relatively high computational and memory costs per batch, particularly when processing high-resolution input and using deep fully connected layers. Both FS-Net and RNN models operate bidirectionally over sequences. Their computational load increases significantly with the number of layers, which makes them relatively demanding. The 1D-CNN model also presents a high computational cost because it uses more than 1000 neurons in the first fully connected layer. In contrast, the computational complexity of the GCN model is primarily determined by the number of graph edges. Since only three neighbors are used to construct each graph, the complexity remains comparatively low. Among all models, Deep Packet and FlowPic exhibit the lowest computational complexity. This is mainly due to their streamlined 1D-CNN architectures and efficient feature extraction mechanisms.

Table 3
Summary of Computational Complexity of All Classification Models

| **Models** | **Computational complexity** |
|---|---|
| 1D-CNN [52] | $\mathcal{O}(B \times L \times F_{1D})$ |
| RNN | $\mathcal{O}\left(B \times L \times n_{\text{layers}} \times (D \times H + H^2)\right)$ |
| GCN [57] | $\mathcal{O}\left(B \times \left(N \times {F_{gcn}}^2 + \lvert E\rvert \times F_{gcn}\right)\right)$ |
| Stacking | $\mathcal{O}(M \times T \times N \times \log N) + \mathcal{O}\left((M + D) \times N\right)$ |
| 2D-CNN | $\mathcal{O}(B \times H_s \times W \times F_{2D})$ |
| Deep Packet [150] | $\mathcal{O}(B \times D \times H^2)$ |
| FlowPic [151] | $\mathcal{O}(B \times D \times H^2)$ |
| FS-Net [152] | $\mathcal{O}\left(B \times L \times H^2 \times n_{\text{layers}}\right)$ |

Note:
$B$ − batch size;
$C$ − number of classes;
$D$ − input feature size;
$F$ − number of output channels;
$F_{1D}$ − total filter operations in 1D convolutional layers;
$F_{2D}$ − total filter operations in 2D convolutional layers;
$L$ − length of 1D sequence in 1D-CNN/RNN;
$n_{\text{layers}}$ − number of RNN/GRU layers;
$F_{gcn}$ − feature dimension per node;
$H$ − number of hidden units in RNN/GRU layers;
$H_s$ − spatial height of 2D input;
$W$ − spatial width of 2D input;
$|E|$ − number of edges;
$N$ − number of training samples;
$M$ − number of meta learners;
$T$ − number of estimators.

# 3. MAIN RESULTS

## 3.1. Time Series Level Classification Performance

In the time series-level evaluation of network traffic classification performance for interactive media applications, both packet-based and byte-based time series were utilized. Seven classification models were applied, resulting in a total of 28 classification experiments per dataset. These datasets included Live Streaming, Cloud Gaming, and Streaming VLS Simulation. The Metaverse dataset was excluded from time series-level classification due to the short duration of raw traffic data, which prevented the construction of appropriate time series. As summarized in Table 6, the 1D-CNN model achieved the highest classification accuracy on the Live Streaming dataset, reaching 0.65 on both packet and byte time series. Without SMOTEENN resampling, the Stacking model's performance was over 15 % lower than that of the 1D-CNN model. After applying SMOTEENN, classification accuracy for the 1D-CNN model improved by more than 30 %, reaching 0.95 on the packet time series with 95 categories. Similar improvements were observed for all models at the byte time series level.

Notably, the performance gap between the Stacking and 1D-CNN models narrowed significantly after resampling, with the Stacking model achieving classification accuracy within 5 % of that of the 1D-CNN model on both packet and byte time series. Given the large number of categories and imbalanced sample distribution, the macro F1-score was adopted as a more suitable evaluation metric. It treats all categories equally and avoids favoring categories with large sample volumes. The highest macro F1-score, 0.93, was recorded by the 1D-CNN model on the packet time series after SMOTEENN resampling. Table 7 provides detailed per-category classification results. Most categories achieved F1-scores above 0.8. However, some categories

still suffered from low precision and recall. For instance, the 480P_3Mbps_N3_10MHz category, despite its large sample size, had a recall of only 0.68, with many samples misclassified as 1080P_60FPS_VLS_YouTube. These two categories exhibit low traffic similarity, where 480P_3Mbps_N3_10MHz averages around 50 packets per 100 ms, while the 1080P_60FPS_VLS_YouTube category averages around 200 packets per 100 ms. The poor performance is likely due to SMOTE generating interpolated samples in noisy or overlapping regions without considering inter-class distances, leading to misclassifications. Additionally, categories such as 1440P_60FPS_VLS_BiliBili and 1080P_60FPS_VoD_YouTube were also frequently misclassified as 1080P_60FPS_VLS_YouTube due to their high traffic similarity, with average packet counts around 300 packets per 100 ms. Therefore, low classification performance is not solely caused by small sample sizes, and noise amplification during resampling also plays a significant role.

In the Live Streaming dataset, after applying the SMOTEENN technique, the total number of samples increased by over 25 percent. However, for the originally small-sized and category-balanced Cloud Gaming dataset, the sample count decreased by more than 30 percent, with each category containing fewer than 50 samples. This reduction is attributed to the ENN component of SMOTEENN, which aggressively removes samples, particularly in datasets that are balanced but noisy or exhibit overlapping categories. Although the highest classification accuracy reached 0.94 using the 1D-CNN model at the packet time series level, the limited sample size undermines the reliability of this performance. As shown in Table 9, three categories, namely SP_720P_VP9_WebRTC, TR_2160P_VP9_WebRTC, and TR_1080P_VP9_Play_State, achieved zero correct classifications. This is due to the absence of corresponding category samples in the testing dataset, which significantly reduced both precision and recall. Upon reviewing the raw data files, it is evident that although more than 600 seconds of traffic duration was available to construct the time series level dataset, this duration was still insufficient to support the development of a high-quality and well-balanced dataset.

For the Streaming VLS Simulation dataset, the raw traffic duration was also approximately 600 seconds. However, due to the total sample size exceeding 10,000, the large dataset allowed SMOTE to generate more reliable synthetic samples. Additionally, ENN benefited from the dense neighborhood structure, which reduced the likelihood of false misclassifications. After applying SMOTEENN, all categories were correctly oversampled, and the sample volume for each category more than doubled. According to the classification results presented in Table 10, all models initially achieved classification accuracies below 0.5 at both the packet and byte time series levels without resampling. This highlights the difficulty of accurate classification when the number of samples is insufficient. After applying SMOTEENN, the best performance was achieved by the 1D-CNN model at the byte time series level, which reached a classification accuracy of 0.91 across 78 categories and a macro F1-score of 0.90. Compared to the packet time series level, this result shows over a 7 % improvement in classification performance under the same conditions. This improvement is understandable, as certain categories, such as WebRTC, generate large volumes of packets, contributing to data noise and making classification more challenging at the packet level. The Stacking model also performed well,

achieving over 0.8 accuracy, though about 8 % lower than the 1D-CNN. A detailed review of Table 11 shows that the 360P30FPS_HLS_VLS_Download category had a precision of only 0.47, primarily due to frequent misclassification into the 240P30FPS_HLS_VLS_Download category, which had a recall score of 0.59. Similarly, the 480P30FPS_HLS_VLS_Download and 720P60FPS_RTMP_VLS_Upload categories exhibited misclassifications that contributed to lower precision. The 2160P30FPS_HLS_VLS_Download and 2160P60FPS_HLS_VLS_Download categories were also difficult to distinguish due to their high similarity.

Overall, both the 1D-CNN and Stacking models demonstrated strong classification performance at the time series level, with the 1D-CNN model showing the best and most robust results. Most categories achieved high precision and recall scores, although a few categories had low accuracy due to either model limitations or excessive data noise. However, it is important to note that this high classification performance was largely dependent on the resampling techniques. The effectiveness of these techniques is significantly influenced by several key factors: dataset size, feature variance, class imbalance, and class separability. Larger datasets provide a denser and more representative feature space, allowing SMOTE to generate more meaningful and diverse synthetic samples. High feature variance enables interpolation between distinct samples, reducing the likelihood of ambiguous or noisy points. When genuine class imbalance exists, SMOTE effectively improves minority class representation. However, in balanced or low-variance datasets, SMOTE may generate low-quality or redundant samples. Additionally, when class separability is poor, synthetic samples may cross decision boundaries, especially when combined with aggressive noise cleaning methods like ENN, which may further degrade data quality by removing valid but overlapping instances. Therefore, the success of synthetic sampling methods is highly dependent on the underlying data characteristics. If the focus shifts to using only raw, real samples without synthetic augmentation, the importance of high-quality data acquisition increases significantly. Active traffic sniffing over extended durations of over 30 minutes may increase collection complexity, but is necessary for sufficient data volume. It is also crucial to consider the quality of packet or byte-level data. For example, traffic from low-resolution and low-framerate sources may consist of a large number of small packets with low payloads, increasing noise and making classification more difficult. Such noise must be carefully cleaned to improve model performance.

Table 4

Classification Performance Summary of Live Streaming Datasets at the Time Series Level

| Time series | $T_{smp}$ | $T_{wnd}$ | Resampling | Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Packet | 100 ms | 5s | / | 1D-CNN | 95 | 0.72 | 0.65 | 0.66 | 0.65 | 0.65 | 0.75 | 0.75 |
| Packet | 100 ms | 5s | / | RNN | 95 | 0.43 | 0.42 | 0.38 | 0.42 | 0.41 | 0.42 | 0.43 |
| Packet | 100 ms | 5s | / | GCN | 95 | 0.49 | 0.43 | 0.43 | 0.43 | 0.43 | 0.50 | 0.42 |
| Packet | 100 ms | 5s | / | Stacking | 95 | 0.60 | 0.58 | 0.59 | 0.58 | 0.58 | 0.58 | 0.58 |
| Packet | 100 ms | 5s | / | Deep Packet | 95 | 0.57 | 0.54 | 0.54 | 0.54 | 0.55 | 0.58 | 0.57 |
| Packet | 100 ms | 5s | / | FlowPic | 95 | 0.47 | 0.46 | 0.43 | 0.46 | 0.46 | 0.47 | 0.48 |
| Packet | 100 ms | 5s | / | FS-Net | 95 | 0.52 | 0.51 | 0.50 | 0.51 | 0.51 | 0.54 | 0.53 |
| Byte | 100 ms | 5s | / | 1D-CNN | 95 | 0.72 | 0.65 | 0.66 | 0.65 | 0.65 | 0.77 | 0.76 |
| Byte | 100 ms | 5s | / | RNN | 95 | 0.43 | 0.42 | 0.39 | 0.42 | 0.42 | 0.42 | 0.42 |

Table 4 (continued)

| | | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Byte | 100 ms | 5s | / | GCN | 95 | 0.46 | 0.39 | 0.41 | 0.39 | 0.39 | 0.52 | 0.42 |
| Byte | 100 ms | 5s | / | Stacking | 95 | 0.65 | 0.63 | 0.63 | 0.63 | 0.63 | 0.82 | 0.62 |
| Byte | 100 ms | 5s | / | Deep Packet | 95 | 0.57 | 0.55 | 0.55 | 0.55 | 0.55 | 0.59 | 0.58 |
| Byte | 100 ms | 5s | / | FlowPic | 95 | 0.47 | 0.47 | 0.44 | 0.47 | 0.46 | 0.48 | 0.46 |
| Byte | 100 ms | 5s | / | FS-Net | 95 | 0.55 | 0.51 | 0.50 | 0.51 | 0.50 | 0.54 | 0.54 |
| Packet | 100 ms | 5s | SMOTEENN | 1D-CNN | 95 | 0.96 | 0.95 | 0.93 | 0.95 | 0.95 | 0.98 | 0.97 |
| Packet | 100 ms | 5s | SMOTEENN | RNN | 95 | 0.75 | 0.75 | 0.66 | 0.75 | 0.75 | 0.76 | 0.76 |
| Packet | 100 ms | 5s | SMOTEENN | GCN | 95 | 0.84 | 0.82 | 0.75 | 0.82 | 0.82 | 0.82 | 0.79 |
| Packet | 100 ms | 5s | SMOTEENN | Stacking | 95 | 0.90 | 0.90 | 0.82 | 0.90 | 0.90 | 0.99 | 0.90 |
| Packet | 100 ms | 5s | SMOTEENN | Deep Packet | 95 | 0.83 | 0.83 | 0.75 | 0.83 | 0.82 | 0.84 | 0.84 |
| Packet | 100 ms | 5s | SMOTEENN | FlowPic | 95 | 0.73 | 0.73 | 0.64 | 0.73 | 0.74 | 0.74 | 0.74 |
| Packet | 100 ms | 5s | SMOTEENN | FS-Net | 95 | 0.83 | 0.82 | 0.76 | 0.82 | 0.82 | 0.86 | 0.86 |
| Byte | 100 ms | 5s | SMOTEENN | 1D-CNN | 95 | 0.96 | 0.94 | 0.91 | 0.94 | 0.94 | 0.97 | 0.97 |
| Byte | 100 ms | 5s | SMOTEENN | RNN | 95 | 0.73 | 0.73 | 0.66 | 0.73 | 0.73 | 0.75 | 0.77 |
| Byte | 100 ms | 5s | SMOTEENN | GCN | 95 | 0.83 | 0.81 | 0.76 | 0.81 | 0.81 | 0.82 | 0.78 |
| Byte | 100 ms | 5s | SMOTEENN | Stacking | 95 | 0.91 | 0.91 | 0.84 | 0.91 | 0.91 | 0.99 | 0.91 |
| Byte | 100 ms | 5s | SMOTEENN | Deep Packet | 95 | 0.83 | 0.82 | 0.76 | 0.82 | 0.82 | 0.84 | 0.84 |
| Byte | 100 ms | 5s | SMOTEENN | FlowPic | 95 | 0.72 | 0.72 | 0.63 | 0.72 | 0.72 | 0.72 | 0.73 |
| Byte | 100 ms | 5s | SMOTEENN | FS-Net | 95 | 0.86 | 0.85 | 0.79 | 0.85 | 0.85 | 0.87 | 0.87 |

Table 5

Classification Performance in Each Category of Live Streaming Datasets under the 1D-CNN Model after Resampling at the Packet Time Series Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 480P_3Mbps_N3_10Mhz | 0.98 | 0.68 | 0.8 | VPN_Vimeo | 1 | 1 | 1 |
| 480P_3Mbps_N78_10Mhz | 0.96 | 0.91 | 0.93 | VPN_YouTube | 0.98 | 0.78 | 0.87 |
| 480P_3Mbps_N78_20Mhz | 0.94 | 0.92 | 0.93 | 160P_30FPS_VoD_Twitch | 0.99 | 0.91 | 0.95 |
| 720P_6Mbps_N3_10Mhz | 0.99 | 0.96 | 0.97 | 160P_30FPS_VLS_Twitch | 1 | 0.99 | 1 |
| 720P_6Mbps_N78_10Mhz | 0.99 | 1 | 1 | 360P_30FPS_VoD_Twitch | 1 | 0.9 | 0.95 |
| 720P_6Mbps_N78_20Mhz | 0.96 | 0.98 | 0.97 | 360P_30FPS_VLS_Twitch | 0.97 | 0.99 | 0.98 |
| 1080P_10Mbps_N3_10Mhz | 0.99 | 1 | 1 | 480P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 1080P_10Mbps_N78_10Mhz | 0.98 | 1 | 0.99 | 480P_30FPS_VLS_Twitch | 0.97 | 0.98 | 0.97 |
| 1080P_10Mbps_N78_20Mhz | 0.97 | 0.93 | 0.95 | 720P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 2160P_20Mbps_N3_10Mhz | 0.99 | 0.96 | 0.98 | 720P_60FPS_VoD_Twitch | 1 | 1 | 1 |
| 2160P_20Mbps_N78_20Mhz | 1 | 0.99 | 0.99 | 720P_30FPS_VLS_Twitch | 0.95 | 0.89 | 0.92 |
| 4320P_40Mbps_N3_10Mhz | 0.95 | 0.97 | 0.96 | 720P_60FPS_VLS_Twitch | 0.96 | 0.94 | 0.95 |
| 4320P_40Mbps_N78_20Mhz | 0.97 | 0.99 | 0.98 | 1080P_30FPS_VoD_Twitch | 0.97 | 0.96 | 0.97 |
| 2160P_30FPS_VoD_BiliBili | 0.99 | 1 | 1 | 1080P_60FPS_VoD_Twitch | 0.97 | 0.98 | 0.97 |
| 2160P_60FPS_VoD_BiliBili | 1 | 0.95 | 0.98 | 1080P_30FPS_VLS_Twitch | 0.99 | 1 | 1 |
| 2160P_120FPS_VoD_BiliBili | 0.99 | 1 | 1 | 1080P_60FPS_VLS_Twitch | 0.95 | 0.95 | 0.95 |
| 2160P_60FPS_VLS_BiliBili | 0.99 | 0.99 | 0.99 | 240P_30FPS_VoD_Vimeo | 0.98 | 0.94 | 0.96 |
| 2160P_80FPS_VLS_BiliBili | 0.83 | 1 | 0.91 | 360P_30FPS_VoD_Vimeo | 1 | 0.94 | 0.97 |
| 4320P_30FPS_VoD_BiliBili | 1 | 0.93 | 0.96 | 540P_30FPS_VoD_Vimeo | 0.77 | 0.89 | 0.83 |
| 360P_30FPS_VoD_BiliBili | 0.93 | 0.97 | 0.95 | 720P_30FPS_VoD_Vimeo | 0.95 | 0.97 | 0.96 |
| 480P_30FPS_VoD_BiliBili | 0.98 | 0.97 | 0.98 | 1080P_30FPS_VoD_Vimeo | 0.82 | 0.87 | 0.84 |
| 480P_30FPS_VLS_BiliBili | 0.98 | 0.98 | 0.98 | 1440P_30FPS_VoD_YouTube | 0.96 | 0.97 | 0.96 |
| 720P_30FPS_VoD_BiliBili | 0.97 | 1 | 0.99 | 1440P_60FPS_VoD_YouTube | 0.91 | 0.82 | 0.86 |
| 720P_30FPS_VLS_BiliBili | 0.95 | 0.92 | 0.94 | 1440P_30FPS_VLS_YouTube | 0.98 | 1 | 0.99 |
| 1080P_30FPS_VoD_BiliBili | 1 | 0.9 | 0.95 | 1440P_60FPS_VLS_YouTube | 0.93 | 0.97 | 0.95 |
| 1080P_60FPS_VoD_BiliBili | 0.96 | 0.96 | 0.96 | 2160P_30FPS_VoD_YouTube | 0.9 | 0.91 | 0.9 |
| 1080P_60FPS_VLS_BiliBili | 0.64 | 0.78 | 0.7 | 2160P_60FPS_VoD_YouTube | 0.96 | 1 | 0.98 |
| 1440P_60FPS_VLS_BiliBili | 0.75 | 0.75 | 0.75 | 2160P_30FPS_VLS_YouTube | 0.99 | 1 | 0.99 |
| 144P_30FPS_VOD_Facebook | 0.87 | 0.85 | 0.86 | 2160P_60FPS_VLS_YouTube | 1 | 0.98 | 0.99 |
| 144P_30FPS_VLS_Facebook | 0.8 | 0.8 | 0.8 | 4320P_30FPS_VoD_YouTube | 0.99 | 0.89 | 0.93 |
| 240P_30FPS_VLS_Facebook | 0.98 | 0.96 | 0.97 | 4320P_60FPS_VoD_YouTube | 0.99 | 1 | 1 |
| 360P_30FPS_VOD_Facebook | 0.79 | 0.9 | 0.84 | 144P_30FPS_VoD_YouTube | 0.97 | 1 | 0.98 |
| 360P_30FPS_VLS_Facebook | 0.92 | 0.63 | 0.75 | 144P_30FPS_VLS_YouTube | 0.96 | 0.95 | 0.95 |
| 480P_30FPS_VOD_Facebook | 0.97 | 0.99 | 0.98 | 240P_30FPS_VoD_YouTube | 0.98 | 0.97 | 0.98 |
| 480P_30FPS_VLS_Facebook | 1 | 0.8 | 0.89 | 240P_30FPS_VLS_YouTube | 1 | 1 | 1 |
| 540P_30FPS_VOD_Facebook | 0.92 | 0.8 | 0.86 | 360P_30FPS_VoD_YouTube | 1 | 0.99 | 1 |
| 720P_30FPS_VOD_Facebook | 0.95 | 0.97 | 0.96 | 360P_30FPS_VLS_YouTube | 0.97 | 0.97 | 0.97 |
| 720P_30FPS_VLS_Facebook | 0.88 | 0.95 | 0.91 | 480P_30FPS_VoD_YouTube | 0.98 | 0.93 | 0.96 |

Table 5 (continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 1080P_30FPS_VOD_Facebook | 0.98 | 0.93 | 0.96 | 480P_30FPS_VLS_YouTube | 0.92 | 1 | 0.96 |
| 1080P_30FPS_VLS_Facebook | 0.87 | 0.99 | 0.93 | 720P_30FPS_VoD_YouTube | 0.89 | 0.73 | 0.8 |
| 360P_30FPS_VLS_TikTok | 1 | 1 | 1 | 720P_60FPS_VoD_YouTube | 0.95 | 0.91 | 0.93 |
| 540P_30FPS_VLS_TikTok | 1 | 0.99 | 1 | 720P_30FPS_VLS_YouTube | 0.94 | 0.96 | 0.95 |
| 720P_30FPS_VLS_TikTok | 0.97 | 0.98 | 0.98 | 720P_60FPS_VLS_YouTube | 0.99 | 0.99 | 0.99 |
| 720P_60FPS_VLS_TikTok | 0.99 | 1 | 1 | 1080P_30FPS_VoD_YouTube | 0.77 | 0.83 | 0.8 |
| 1080P_30FPS_VLS_TikTok | 1 | 1 | 1 | 1080P_60FPS_VoD_YouTube | 1 | 0.11 | 0.2 |
| 1080P_60FPS_VLS_TikTok | 1 | 0.99 | 1 | 1080P_30FPS_VLS_YouTube | 0.97 | 0.99 | 0.98 |
| Tor_Vimeo | 0.95 | 0.98 | 0.97 | 1080P_60FPS_VLS_YouTube | 0.38 | 0.94 | 0.54 |
| Tor_YouTube | 0.97 | 0.95 | 0.96 | | | | |

Table 6

Classification Performance Summary of Cloud Gaming Datasets at the Time Series Level

| **Time series** | $T_{smp}$ | $T_{wnd}$ | **Resampling** | **Model** | **Category** | **P.W** | **R.W** | **F.A** | **F.I** | **A.Test** | **A.Train** | **A.Validate** |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Packet | 100 ms | 5 s | / | 1D-CNN | 12 | 0.58 | 0.63 | 0.55 | 0.63 | 0.63 | 0.61 | 0.59 |
| Packet | 100 ms | 5 s | / | RNN | 12 | 0.49 | 0.53 | 0.47 | 0.52 | 0.50 | 0.51 | 0.51 |
| Packet | 100 ms | 5 s | / | GCN | 12 | 0.30 | 0.38 | 0.29 | 0.38 | 0.38 | 0.46 | 0.37 |
| Packet | 100 ms | 5 s | / | Stacking | 12 | 0.65 | 0.64 | 0.64 | 0.64 | 0.64 | 0.98 | 0.61 |
| Packet | 100 ms | 5 s | / | Deep Packet | 12 | 0.49 | 0.49 | 0.45 | 0.49 | 0.46 | 0.49 | 0.47 |
| Packet | 100 ms | 5 s | / | FlowPic | 12 | 0.45 | 0.47 | 0.41 | 0.47 | 0.43 | 0.43 | 0.44 |
| Packet | 100 ms | 5 s | / | FS-Net | 12 | 0.56 | 0.53 | 0.51 | 0.52 | 0.52 | 0.51 | 0.51 |
| Byte | 100 ms | 5 s | / | 1D-CNN | 12 | 0.57 | 0.65 | 0.55 | 0.58 | 0.52 | 0.53 | 0.52 |
| Byte | 100 ms | 5 s | / | RNN | 12 | 0.56 | 0.58 | 0.52 | 0.53 | 0.47 | 0.51 | 0.48 |
| Byte | 100 ms | 5 s | / | GCN | 12 | 0.38 | 0.37 | 0.31 | 0.37 | 0.37 | 0.48 | 0.39 |
| Byte | 100 ms | 5 s | / | Stacking | 12 | 0.50 | 0.48 | 0.48 | 0.48 | 0.48 | 0.98 | 0.48 |
| Byte | 100 ms | 5 s | / | Deep Packet | 12 | 0.43 | 0.45 | 0.41 | 0.45 | 0.48 | 0.48 | 0.48 |
| Byte | 100 ms | 5 s | / | FlowPic | 12 | 0.41 | 0.43 | 0.38 | 0.43 | 0.42 | 0.45 | 0.42 |
| Byte | 100 ms | 5 s | / | FS-Net | 12 | 0.57 | 0.59 | 0.51 | 0.53 | 0.48 | 0.49 | 0.48 |
| Packet | 100 ms | 5 s | SMOTEENN | 1D-CNN | 12 | 0.95 | 0.97 | 0.95 | 0.96 | 0.94 | 0.96 | 0.89 |
| Packet | 100 ms | 5 s | SMOTEENN | RNN | 12 | 0.76 | 0.77 | 0.65 | 0.74 | 0.69 | 0.76 | 0.65 |
| Packet | 100 ms | 5 s | SMOTEENN | GCN | 12 | 0.54 | 0.64 | 0.33 | 0.64 | 0.64 | 0.76 | 0.64 |
| Packet | 100 ms | 5 s | SMOTEENN | Stacking | 12 | 0.74 | 0.74 | 0.72 | 0.73 | 0.73 | 0.97 | 0.73 |
| Packet | 100 ms | 5 s | SMOTEENN | Deep Packet | 12 | 0.79 | 0.81 | 0.71 | 0.79 | 0.73 | 0.76 | 0.74 |
| Packet | 100 ms | 5 s | SMOTEENN | FlowPic | 12 | 0.64 | 0.70 | 0.59 | 0.64 | 0.59 | 0.61 | 0.51 |
| Packet | 100 ms | 5 s | SMOTEENN | FS-Net | 12 | 0.82 | 0.88 | 0.80 | 0.79 | 0.74 | 0.75 | 0.79 |
| Byte | 100 ms | 5 s | SMOTEENN | 1D-CNN | 12 | 0.93 | 0.90 | 0.89 | 0.89 | 0.88 | 0.86 | 0.96 |
| Byte | 100 ms | 5 s | SMOTEENN | RNN | 12 | 0.87 | 0.87 | 0.82 | 0.85 | 0.82 | 0.79 | 0.79 |
| Byte | 100 ms | 5 s | SMOTEENN | GCN | 12 | 0.68 | 0.65 | 0.42 | 0.65 | 0.65 | 0.81 | 0.71 |
| Byte | 100 ms | 5 s | SMOTEENN | Stacking | 12 | 0.63 | 0.63 | 0.59 | 0.63 | 0.62 | 0.97 | 0.61 |
| Byte | 100 ms | 5 s | SMOTEENN | Deep Packet | 12 | 0.78 | 0.79 | 0.67 | 0.78 | 0.82 | 0.78 | 0.83 |
| Byte | 100 ms | 5 s | SMOTEENN | FlowPic | 12 | 0.73 | 0.79 | 0.67 | 0.76 | 0.69 | 0.66 | 0.74 |
| Byte | 100 ms | 5 s | SMOTEENN | FS-Net | 12 | 0.84 | 0.80 | 0.72 | 0.79 | 0.78 | 0.77 | 0.79 |

Table 7
Classification Performance in Each Category of Cloud Gaming Datasets under the 1D-CNN Model after Resampling at the Packet Time Series Level

| Category | P. | R. | F. |
|---|---|---|---|
| SP_2160P_VP9_WebRTC | 0.67 | 0.67 | 0.67 |
| SP_720P_VP9_WebRTC | 0 | 0 | 0 |
| SP_1080P_H264_WebRTC | 1 | 1 | 1 |
| SP_1080P_VP9_WebRTC | 0.88 | 1 | 0.94 |
| SP_1080P_VP9_RTP | 1 | 1 | 1 |
| SP_1080P_VP9_Play_State | 0.89 | 1 | 0.94 |
| TH_1080P_VP9_RTP | 1 | 1 | 1 |
| TR_2160P_VP9_WebRTC | 0 | 0 | 0 |
| TR_720P_VP9_WebRTC | 1 | 0.92 | 0.96 |
| TR_1080P_VP9_WebRTC | 1 | 1 | 1 |
| TR_1080P_VP9_RTP | 1 | 1 | 1 |
| TR_1080P_VP9_ Play_State | 0 | 0 | 0 |

Table 8
Classification Performance Summary of Streaming VLS Simulation Datasets at the Time Series Level

| Time series | $T_{smp}$ | $T_{wnd}$ | Resampling | Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Packet | 100 ms | 5 s | / | 1D-CNN | 78 | 0.45 | 0.44 | 0.39 | 0.44 | 0.43 | 0.56 | 0.55 |
| Packet | 100 ms | 5 s | / | RNN | 78 | 0.30 | 0.33 | 0.26 | 0.31 | 0.30 | 0.34 | 0.34 |
| Packet | 100 ms | 5 s | / | GCN | 78 | 0.34 | 0.35 | 0.31 | 0.35 | 0.35 | 0.51 | 0.35 |
| Packet | 100 ms | 5 s | / | Stacking | 78 | 0.34 | 0.34 | 0.30 | 0.34 | 0.34 | 0.91 | 0.34 |
| Packet | 100 ms | 5 s | / | Deep Packet | 78 | 0.34 | 0.34 | 0.30 | 0.34 | 0.35 | 0.40 | 0.38 |
| Packet | 100 ms | 5 s | / | FlowPic | 78 | 0.29 | 0.31 | 0.27 | 0.31 | 0.29 | 0.33 | 0.34 |
| Packet | 100 ms | 5 s | / | FS-Net | 78 | 0.33 | 0.33 | 0.29 | 0.33 | 0.34 | 0.39 | 0.39 |
| Byte | 100 ms | 5 s | / | 1D-CNN | 78 | 0.45 | 0.45 | 0.41 | 0.45 | 0.45 | 0.69 | 0.67 |
| Byte | 100 ms | 5 s | / | RNN | 78 | 0.28 | 0.28 | 0.23 | 0.28 | 0.27 | 0.31 | 0.30 |
| Byte | 100 ms | 5 s | / | GCN | 78 | 0.27 | 0.25 | 0.22 | 0.25 | 0.25 | 0.46 | 0.29 |
| Byte | 100 ms | 5 s | / | Stacking | 78 | 0.33 | 0.33 | 0.29 | 0.33 | 0.33 | 0.93 | 0.32 |
| Byte | 100 ms | 5 s | / | Deep Packet | 78 | 0.34 | 0.34 | 0.31 | 0.34 | 0.32 | 0.40 | 0.39 |
| Byte | 100 ms | 5 s | / | FlowPic | 78 | 0.26 | 0.26 | 0.22 | 0.26 | 0.27 | 0.30 | 0.30 |
| Byte | 100 ms | 5 s | / | FS-Net | 78 | 0.30 | 0.31 | 0.25 | 0.30 | 0.30 | 0.34 | 0.34 |
| Packet | 100 ms | 5 s | SMOTEENN | 1D-CNN | 78 | 0.85 | 0.84 | 0.83 | 0.84 | 0.84 | 0.89 | 0.91 |
| Packet | 100 ms | 5 s | SMOTEENN | RNN | 78 | 0.41 | 0.42 | 0.37 | 0.41 | 0.40 | 0.41 | 0.40 |
| Packet | 100 ms | 5 s | SMOTEENN | GCN | 78 | 0.77 | 0.76 | 0.74 | 0.76 | 0.76 | 0.78 | 0.70 |
| Packet | 100 ms | 5 s | SMOTEENN | Stacking | 78 | 0.83 | 0.83 | 0.82 | 0.83 | 0.83 | 0.99 | 0.84 |
| Packet | 100 ms | 5 s | SMOTEENN | Deep Packet | 78 | 0.56 | 0.57 | 0.54 | 0.57 | 0.57 | 0.60 | 0.59 |
| Packet | 100 ms | 5 s | SMOTEENN | FlowPic | 78 | 0.44 | 0.45 | 0.41 | 0.45 | 0.45 | 0.45 | 0.45 |
| Packet | 100 ms | 5 s | SMOTEENN | FS-Net | 78 | 0.55 | 0.54 | 0.51 | 0.54 | 0.55 | 0.56 | 0.57 |
| Byte | 100 ms | 5 s | SMOTEENN | 1D-CNN | 78 | 0.91 | 0.91 | 0.90 | 0.91 | 0.91 | 0.96 | 0.96 |
| Byte | 100 ms | 5 s | SMOTEENN | RNN | 78 | 0.42 | 0.42 | 0.41 | 0.42 | 0.42 | 0.45 | 0.43 |
| Byte | 100 ms | 5 s | SMOTEENN | GCN | 78 | 0.78 | 0.76 | 0.76 | 0.76 | 0.76 | 0.80 | 0.70 |
| Byte | 100 ms | 5 s | SMOTEENN | Stacking | 78 | 0.82 | 0.82 | 0.79 | 0.82 | 0.82 | 0.99 | 0.82 |
| Byte | 100 ms | 5 s | SMOTEENN | Deep Packet | 78 | 0.62 | 0.61 | 0.59 | 0.61 | 0.60 | 0.65 | 0.65 |
| Byte | 100 ms | 5 s | SMOTEENN | FlowPic | 78 | 0.43 | 0.44 | 0.42 | 0.44 | 0.45 | 0.47 | 0.46 |
| Byte | 100 ms | 5 s | SMOTEENN | FS-Net | 78 | 0.66 | 0.65 | 0.64 | 0.65 | 0.66 | 0.70 | 0.72 |

Table 9
Classification Performance in Each Category of Streaming VLS Simulation Datasets under the 1D-CNN Model after Resampling at the Byte Time Series Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 144P30FPS_DASH_VLS_Download | 0.8 | 0.74 | 0.77 | 720P60FPS_RTMP_VLS_Download | 0.93 | 0.93 | 0.93 |
| 144P30FPS_HLS_VLS_Download | 0.97 | 1 | 0.98 | 720P60FPS_RTMP_VLS_Upload | 1 | 0.67 | 0.8 |
| 144P30FPS_HTTP_FLV_VLS_Download | 0.93 | 0.97 | 0.95 | 720P60FPS_WebRTC_VLS_Download | 0.96 | 0.88 | 0.92 |
| 144P30FPS_RTMP_VLS_Download | 1 | 0.93 | 0.96 | 1080P30FPS_DASH_VLS_Download | 0.96 | 1 | 0.98 |
| 144P30FPS_RTMP_VLS_Upload | 0.93 | 0.93 | 0.93 | 1080P30FPS_HLS_VLS_Download | 0.96 | 0.99 | 0.98 |
| 144P30FPS_WebRTC_VLS_Download | 0.9 | 0.99 | 0.94 | 1080P30FPS_HTTP_FLV_VLS_Download | 0.87 | 0.93 | 0.9 |
| 240P30FPS_DASH_VLS_Download | 0.88 | 0.95 | 0.91 | 1080P30FPS_RTMP_VLS_Download | 0.96 | 0.94 | 0.95 |
| 240P30FPS_HLS_VLS_Download | 0.59 | 0.83 | 0.69 | 1080P30FPS_RTMP_VLS_Upload | 0.92 | 0.92 | 0.92 |
| 240P30FPS_HTTP_FLV_VLS_Download | 0.88 | 0.89 | 0.88 | 1080P30FPS_WebRTC_VLS_Download | 0.91 | 0.78 | 0.84 |
| 240P30FPS_RTMP_VLS_Download | 0.95 | 0.9 | 0.93 | 1080P60FPS_DASH_VLS_Download | 0.88 | 0.96 | 0.92 |
| 240P30FPS_RTMP_VLS_Upload | 0.97 | 0.9 | 0.94 | 1080P60FPS_HLS_VLS_Download | 1 | 0.85 | 0.92 |
| 240P30FPS_WebRTC_VLS_Download | 0.91 | 0.91 | 0.91 | 1080P60FPS_HTTP_FLV_VLS_Download | 0.97 | 0.9 | 0.94 |
| 360P30FPS_DASH_VLS_Download | 0.96 | 0.62 | 0.75 | 1080P60FPS_RTMP_VLS_Download | 0.94 | 0.96 | 0.95 |
| 360P30FPS_HLS_VLS_Download | 0.44 | 0.47 | 0.46 | 1080P60FPS_RTMP_VLS_Upload | 0.76 | 0.96 | 0.85 |
| 360P30FPS_HTTP_FLV_VLS_Download | 0.85 | 0.91 | 0.88 | 1080P60FPS_WebRTC_VLS_Download | 0.87 | 0.87 | 0.87 |
| 360P30FPS_RTMP_VLS_Download | 0.96 | 0.96 | 0.96 | 1440P30FPS_DASH_VLS_Download | 0.95 | 0.88 | 0.91 |
| 360P30FPS_RTMP_VLS_Upload | 0.97 | 0.93 | 0.95 | 1440P30FPS_HLS_VLS_Download | 0.78 | 0.82 | 0.8 |
| 360P30FPS_WebRTC_VLS_Download | 0.82 | 0.94 | 0.88 | 1440P30FPS_HTTP_FLV_VLS_Download | 0.82 | 0.97 | 0.89 |
| 480P30FPS_DASH_VLS_Download | 0.82 | 0.82 | 0.82 | 1440P30FPS_RTMP_VLS_Download | 0.85 | 0.93 | 0.89 |
| 480P30FPS_HLS_VLS_Download | 0.79 | 0.6 | 0.68 | 1440P30FPS_RTMP_VLS_Upload | 0.97 | 0.79 | 0.87 |
| 480P30FPS_HTTP_FLV_VLS_Download | 0.96 | 0.73 | 0.83 | 1440P30FPS_WebRTC_VLS_Download | 0.95 | 0.88 | 0.91 |
| 480P30FPS_RTMP_VLS_Download | 0.97 | 0.92 | 0.95 | 1440P60FPS_DASH_VLS_Download | 0.95 | 0.81 | 0.88 |
| 480P30FPS_RTMP_VLS_Upload | 0.99 | 0.9 | 0.94 | 1440P60FPS_HLS_VLS_Download | 0.98 | 1 | 0.99 |
| 480P30FPS_WebRTC_VLS_Download | 0.89 | 0.96 | 0.92 | 1440P60FPS_HTTP_FLV_VLS_Download | 0.96 | 0.93 | 0.95 |
| 540P30FPS_DASH_VLS_Download | 0.96 | 0.95 | 0.95 | 1440P60FPS_RTMP_VLS_Download | 0.96 | 0.9 | 0.93 |
| 540P30FPS_HLS_VLS_Download | 1 | 1 | 1 | 1440P60FPS_RTMP_VLS_Upload | 0.94 | 0.94 | 0.94 |
| 540P30FPS_HTTP_FLV_VLS_Download | 0.94 | 0.99 | 0.96 | 1440P60FPS_WebRTC_VLS_Download | 0.94 | 0.91 | 0.93 |
| 540P30FPS_RTMP_VLS_Download | 0.98 | 1 | 0.99 | 2160P30FPS_DASH_VLS_Download | 0.89 | 1 | 0.94 |
| 540P30FPS_RTMP_VLS_Upload | 0.89 | 0.92 | 0.91 | 2160P30FPS_HLS_VLS_Download | 0.67 | 0.66 | 0.67 |
| 540P30FPS_WebRTC_VLS_Download | 0.91 | 0.98 | 0.94 | 2160P30FPS_HTTP_FLV_VLS_Download | 0.99 | 0.84 | 0.91 |
| 720P30FPS_DASH_VLS_Download | 0.85 | 0.98 | 0.91 | 2160P30FPS_RTMP_VLS_Download | 0.88 | 0.92 | 0.9 |
| 720P30FPS_HLS_VLS_Download | 0.95 | 0.96 | 0.96 | 2160P30FPS_RTMP_VLS_Upload | 0.86 | 0.96 | 0.91 |
| 720P30FPS_HTTP_FLV_VLS_Download | 0.78 | 0.99 | 0.87 | 2160P30FPS_WebRTC_VLS_Download | 0.91 | 0.88 | 0.89 |
| 720P30FPS_RTMP_VLS_Download | 0.92 | 0.92 | 0.92 | 2160P60FPS_DASH_VLS_Download | 0.9 | 0.93 | 0.92 |
| 720P30FPS_RTMP_VLS_Upload | 0.78 | 0.96 | 0.86 | 2160P60FPS_HLS_VLS_Download | 0.68 | 0.78 | 0.73 |
| 720P30FPS_WebRTC_VLS_Download | 0.84 | 0.89 | 0.87 | 2160P60FPS_HTTP_FLV_VLS_Download | 0.95 | 0.99 | 0.97 |
| 720P60FPS_DASH_VLS_Download | 0.97 | 0.78 | 0.87 | 2160P60FPS_RTMP_VLS_Download | 0.98 | 0.89 | 0.93 |
| 720P60FPS_HLS_VLS_Download | 0.99 | 1 | 0.99 | 2160P60FPS_RTMP_VLS_Upload | 0.93 | 0.94 | 0.93 |
| 720P60FPS_HTTP-FLV_VLS_Download | 0.96 | 0.84 | 0.9 | 2160P60FPS_WebRTC_VLS_Download | 0.94 | 1 | 0.97 |

## 3.2. Flow-Level Classification Performance

At the flow-level classification task, Table 12 summarizes the classification results for the Live Streaming dataset. The overall classification performance on the raw dataset was poor, with the highest accuracy reaching only 0.4 using the 1D-CNN model. This was largely due to severe class imbalance among categories. Upon investigating individual categories, the 360P_30FPS_VoD_YouTube traffic in the raw PCAP file is primarily transmitted via the QUIC protocol over UDP. It contains four conversation streams corresponding to four dynamic transport layer ports, with each stream lasting over 200 s and one reaching more than 800 s. Similarly, the 480P_30FPS_VoD_YouTube traffic also uses UDP but includes only two conversation streams, meaning the media sender changed ports just twice during the 30-minute session. Using the `NFStream` tool, a new flow sample is triggered either by changes in any of the five-tuple attributes or by a one-minute timer. If a flow lasts longer than one minute, it gets split into a new sample. Due to this, both 360P_30FPS_VoD_YouTube and 480P_30FPS_VoD_YouTube categories generate only 81 flow samples, most of which are

created by the time-based truncation. Since the five-tuple attributes rarely change dynamically, this limits the effectiveness of flow-level sample generation. In contrast, the 720P_60FPS_VLS_Twitch category relies on TCP, includes over 12 TCP conversations, and produces more than 2000 flow-level samples. All three categories span approximately 30 min of traffic, but factors such as IP address changes, port switching, or CDN-based load balancing significantly affect the distribution of flow samples.

Moreover, the Tor_Vimeo raw PCAP file has a duration exceeding 3500 s, which is more than double that of the others. Although it includes only one conversation stream, it still generates over 3000 flow-level samples. The primary reason is that Tor aggregates all streams into a single encrypted tunnel directed to the guard relay, rather than establishing separate sockets for each stream. It also reflects high-quality video playback with short inter-arrival times (IATs), meaning that one-second flow containers cannot hold many packets or payloads, thus requiring more containers and resulting in a larger number of flow samples. In summary, the flow sample volume is heavily influenced by the transmission protocol and the duration of the raw PCAP file. After applying SMOTEENN, the 360P_30FPS_VoD_YouTube and 480P_30FPS_VoD_YouTube categories were oversampled more than tenfold, reaching over 1000 samples each, while the original large-volume categories remained unchanged. This significantly reduced class imbalance and improved classification performance. Notably, the Stacking model achieved a macro F1-score exceeding 0.99, outperforming the 1D-CNN by 2 %. According to Table 13, most category samples were correctly classified, with many achieving 100 % accuracy. Even the lowest F1-score was as high as 0.94, demonstrating the robustness and strong performance of the Stacking model.

For the Metaverse dataset, the overall classification performance is observed to be relatively poor, both in the raw dataset and after applying SMOTEENN. According to the results presented in Table 14, the maximum classification accuracy reaches 0.85, and the macro F1-score peaks at 0.73 after resampling. Each category's raw PCAP traffic duration is limited to only one minute. Since Metaverse applications are based on WebRTC and transmitted over TCP, each PCAP file contains three conversation streams at the transport layer. This results in approximately 180 flow-level samples per category. Due to the small total number of samples, SMOTE struggles to generate meaningful synthetic samples, and ENN tends to aggressively remove data. This is especially problematic in datasets that are balanced but contain noise or exhibit category overlap. As a result, the number of samples per category after SMOTEENN remains under ten, which is insufficient to support effective classification. According to the per-category results in Table 15, more than five categories are entirely missing from the classified results. This is due to the small test set size, which causes some categories to be unrepresented during testing, while misclassifications also remain high. Most categories exhibit F1-scores below 0.8. These findings confirm that small-scale acquisition datasets are not suitable for reliable classification, whether at the time series level or the flow level. Sufficient data volume is essential to enable meaningful classification performance.

According to the classification performance summary of the Streaming VLS Simulation dataset in Table 16, the overall classification results show significant improvement. The Stacking model achieved the highest classification accuracy, reaching 0.9 across 78 categories

without applying any resampling techniques. This strong performance is attributed to the uniform duration of approximately 600 s for all raw PCAP traffic files and a relatively balanced category distribution. However, differences in streaming protocols result in varying numbers of flow samples. For instance, using 1080P30FPS video as a reference, the DASH protocol, which is based on TCP and HTTP, generates over 25 TCP conversation streams. The HLS protocol, also built on TCP and HTTP, results in over 642 TCP streams. In contrast, the HTTP-FLV and RTMP protocols, which use persistent session-based TCP connections, each produce only one TCP stream. WebRTC, which is real-time and based on UDP, generates a single UDP stream. DASH and HLS are segmented and pull-based protocols where the client repeatedly downloads small media chunks and playlist files, leading to a high number of TCP connections. DASH initiates a new HTTP request for each 2–10-second video segment, while HLS frequently polls for playlist files and segment files. In comparison, HTTP-FLV and RTMP maintain long-lived TCP sessions without frequent reconnections. WebRTC uses a persistent UDP stream for continuous communication. With all categories having approximately 600 s of traffic, most yield around 600 flow samples. However, DASH and HLS generate more flow samples than other protocols, with increases exceeding 10 percent. Notably, both 2160P30FPS_HLS_VLS_Download and 2160P60FPS_HLS_VLS_Download categories have over 1500 TCP conversation streams, resulting in more than 2000 flow-level samples. In the 2160P60FPS_HLS_VLS_Download case, latency in video decoding extends the playback duration to over 1500 s. This increases inter-arrival times between packets, which causes the flow segmentation process to generate more flow samples even though the actual number of packets remains nearly the same. This kind of time shift introduces noise into the data, which negatively impacts classification performance at the time series level.

After applying SMOTEENN, all category sample counts approximately doubled. With this enhanced dataset, the Stacking model achieved a classification accuracy of 0.98 and a macro F1-score of 0.97. The increased volume and diversity of flow samples also enabled strong performance in other models. For example, the 1D-CNN model achieved a macro F1-score of 0.87, which is about 10 percent lower than the Stacking model. Models such as RNN, Deep Packet, FlowPic, and FS-Net also performed well, although each lagged behind the Stacking model by more than 20 percent. The RNN model had the weakest results, with a classification accuracy of around 0.64 and a macro F1-score of 0.57. As presented in Table 17, the Stacking model maintained F1-scores close to 0.9 for all categories, demonstrating excellent robustness and effectiveness in flow-level classification. In summary, the overall classification performance across the three datasets is excellent after applying resampling and using 86 flow-level training features. However, the small sample volume remains a persistent issue with `NFStream` exported flow data, and alternative flow management tools require further evaluation.

Table 10

Classification Performance Summary of Live Streaming Datasets at the Flow Level

| Resampling | Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|---|
| / | 1D-CNN | 86 | 0.45 | 0.44 | 0.39 | 0.44 | 0.43 | 0.56 | 0.55 |
| / | RNN | 86 | 0.30 | 0.33 | 0.26 | 0.31 | 0.30 | 0.34 | 0.34 |
| / | GCN | 86 | 0.34 | 0.35 | 0.31 | 0.35 | 0.35 | 0.51 | 0.35 |
| / | Stacking | 86 | 0.34 | 0.34 | 0.30 | 0.34 | 0.34 | 0.91 | 0.34 |
| / | Deep Packet | 86 | 0.34 | 0.34 | 0.30 | 0.34 | 0.35 | 0.40 | 0.38 |
| / | FlowPic | 86 | 0.29 | 0.31 | 0.27 | 0.31 | 0.29 | 0.33 | 0.34 |
| / | FS-Net | 86 | 0.33 | 0.33 | 0.29 | 0.33 | 0.34 | 0.39 | 0.39 |
| SMOTEENN | 1D-CNN | 86 | 0.98 | 0.98 | 0.97 | 0.98 | 0.98 | 0.99 | 0.99 |
| SMOTEENN | RNN | 86 | 0.95 | 0.94 | 0.93 | 0.94 | 0.94 | 0.96 | 0.96 |
| SMOTEENN | GCN | 86 | 0.89 | 0.89 | 0.89 | 0.89 | 0.89 | 0.87 | 0.87 |
| SMOTEENN | Stacking | 86 | 0.99 | 0.99 | 0.99 | 0.99 | 0.99 | 1.00 | 0.99 |
| SMOTEENN | Deep Packet | 86 | 0.93 | 0.93 | 0.92 | 0.93 | 0.93 | 0.94 | 0.93 |
| SMOTEENN | FlowPic | 86 | 0.87 | 0.87 | 0.84 | 0.87 | 0.87 | 0.87 | 0.87 |
| SMOTEENN | FS-Net | 86 | 0.96 | 0.96 | 0.95 | 0.96 | 0.96 | 0.96 | 0.97 |

Table 11

Classification Performance in Each Category of Live Streaming Datasets under the Stacking Model after Resampling at the Flow Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 480P_3Mbps_N3_10Mhz | 1 | 1 | 1 | 360P_30FPS_VoD_Twitch | 0.99 | 1 | 1 |
| 480P_3Mbps_N78_10Mhz | 0.99 | 0.99 | 0.99 | 360P_30FPS_VLS_Twitch | 1 | 1 | 1 |
| 720P_6Mbps_N3_10Mhz | 1 | 0.99 | 1 | 480P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 1080P_10Mbps_N78_10Mhz | 0.99 | 1 | 1 | 480P_30FPS_VLS_Twitch | 1 | 1 | 1 |
| 2160P_30FPS_VoD_BiliBili | 1 | 1 | 1 | 720P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 2160P_60FPS_VoD_BiliBili | 0.99 | 1 | 1 | 720P_60FPS_VoD_Twitch | 1 | 1 | 1 |
| 2160P_120FPS_VoD_BiliBili | 1 | 1 | 1 | 720P_30FPS_VLS_Twitch | 1 | 1 | 1 |
| 2160P_60FPS_VLS_BiliBili | 1 | 1 | 1 | 720P_60FPS_VLS_Twitch | 1 | 1 | 1 |
| 2160P_80FPS_VLS_BiliBili | 1 | 1 | 1 | 1080P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 4320P_30FPS_VoD_BiliBili | 1 | 0.99 | 1 | 1080P_60FPS_VoD_Twitch | 1 | 1 | 1 |
| 360P_30FPS_VoD_BiliBili | 0.99 | 0.99 | 0.99 | 1080P_30FPS_VLS_Twitch | 1 | 1 | 1 |
| 480P_30FPS_VoD_BiliBili | 0.99 | 1 | 0.99 | 1080P_60FPS_VLS_Twitch | 1 | 1 | 1 |
| 480P_30FPS_VLS_BiliBili | 1 | 1 | 1 | 240P_30FPS_VoD_Vimeo | 0.98 | 0.99 | 0.98 |
| 720P_30FPS_VoD_BiliBili | 0.99 | 0.99 | 0.99 | 360P_30FPS_VoD_Vimeo | 0.96 | 0.97 | 0.96 |
| 720P_30FPS_VLS_BiliBili | 1 | 0.99 | 1 | 540P_30FPS_VoD_Vimeo | 0.97 | 0.97 | 0.97 |
| 1080P_30FPS_VoD_BiliBili | 0.99 | 0.98 | 0.99 | 720P_30FPS_VoD_Vimeo | 0.99 | 0.96 | 0.97 |
| 1080P_60FPS_VoD_BiliBili | 1 | 0.99 | 1 | 1080P_30FPS_VoD_Vimeo | 1 | 1 | 1 |
| 1080P_60FPS_VLS_BiliBili | 0.99 | 1 | 1 | 1440P_30FPS_VoD_YouTube | 0.99 | 0.99 | 0.99 |
| 1440P_60FPS_VLS_BiliBili | 1 | 1 | 1 | 1440P_60FPS_VoD_YouTube | 0.92 | 0.96 | 0.94 |
| 144P_30FPS_VOD_Facebook | 1 | 0.99 | 0.99 | 1440P_30FPS_VLS_YouTube | 1 | 0.99 | 0.99 |
| 144P_30FPS_VLS_Facebook | 0.99 | 0.98 | 0.98 | 1440P_60FPS_VLS_YouTube | 1 | 1 | 1 |
| 240P_30FPS_VLS_Facebook | 0.96 | 0.99 | 0.97 | 2160P_30FPS_VoD_YouTube | 1 | 0.99 | 1 |
| 360P_30FPS_VOD_Facebook | 0.98 | 0.99 | 0.99 | 2160P_60FPS_VoD_YouTube | 0.97 | 0.91 | 0.94 |
| 360P_30FPS_VLS_Facebook | 0.95 | 0.95 | 0.95 | 2160P_30FPS_VLS_YouTube | 0.99 | 1 | 0.99 |
| 480P_30FPS_VOD_Facebook | 0.99 | 1 | 1 | 2160P_60FPS_VLS_YouTube | 1 | 1 | 1 |
| 480P_30FPS_VLS_Facebook | 0.97 | 0.96 | 0.97 | 4320P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| 540P_30FPS_VOD_Facebook | 1 | 1 | 1 | 4320P_60FPS_VoD_YouTube | 1 | 1 | 1 |
| 720P_30FPS_VOD_Facebook | 1 | 1 | 1 | 144P_30FPS_VoD_YouTube | 0.99 | 1 | 1 |
| 720P_30FPS_VLS_Facebook | 1 | 1 | 1 | 144P_30FPS_VLS_YouTube | 0.99 | 1 | 1 |
| 1080P_30FPS_VOD_Facebook | 1 | 1 | 1 | 240P_30FPS_VoD_YouTube | 0.99 | 0.99 | 0.99 |
| 1080P_30FPS_VLS_Facebook | 1 | 1 | 1 | 240P_30FPS_VLS_YouTube | 1 | 0.99 | 1 |
| 360P_30FPS_VLS_TikTok | 1 | 1 | 1 | 360P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| 540P_30FPS_VLS_TikTok | 1 | 1 | 1 | 360P_30FPS_VLS_YouTube | 1 | 1 | 1 |
| 720P_30FPS_VLS_TikTok | 1 | 1 | 1 | 480P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| 720P_60FPS_VLS_TikTok | 1 | 1 | 1 | 480P_30FPS_VLS_YouTube | 1 | 1 | 1 |
| 1080P_30FPS_VLS_TikTok | 1 | 1 | 1 | 720P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| 1080P_60FPS_VLS_TikTok | 1 | 1 | 1 | 720P_60FPS_VoD_YouTube | 1 | 1 | 1 |
| Tor_Vimeo | 1 | 1 | 1 | 720P_30FPS_VLS_YouTube | 0.94 | 0.96 | 0.95 |
| Tor_YouTube | 1 | 1 | 1 | 720P_60FPS_VLS_YouTube | 0.99 | 0.99 | 0.99 |
| VPN_Vimeo | 1 | 0.99 | 0.99 | 1080P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| VPN_YouTube | 0.99 | 1 | 0.99 | 1080P_60FPS_VoD_YouTube | 0.99 | 0.99 | 0.99 |
| 160P_30FPS_VoD_Twitch | 1 | 1 | 1 | 1080P_30FPS_VLS_YouTube | 0.97 | 0.95 | 0.96 |
| 160P_30FPS_VLS_Twitch | 1 | 1 | 1 | 1080P_60FPS_VLS_YouTube | 0.99 | 1 | 1 |

Table 12
Classification Performance Summary of Metaverse Datasets at the Flow Level

| Resampling | Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|---|
| / | 1D-CNN | 21 | 0.49 | 0.45 | 0.45 | 0.45 | 0.45 | 0.56 | 0.57 |
| / | RNN | 21 | 0.31 | 0.39 | 0.34 | 0.34 | 0.29 | 0.39 | 0.33 |
| / | GCN | 21 | 0.38 | 0.35 | 0.35 | 0.35 | 0.35 | 0.43 | 0.36 |
| / | Stacking | 21 | 0.47 | 0.46 | 0.45 | 0.46 | 0.46 | 1.00 | 0.48 |
| / | Deep Packet | 21 | 0.37 | 0.36 | 0.37 | 0.36 | 0.37 | 0.40 | 0.40 |
| / | FlowPic | 21 | 0.32 | 0.30 | 0.31 | 0.30 | 0.30 | 0.35 | 0.32 |
| / | FS-Net | 21 | 0.40 | 0.37 | 0.36 | 0.37 | 0.36 | 0.43 | 0.40 |
| SMOTEENN | 1D-CNN | 21 | 0.87 | 0.87 | 0.73 | 0.85 | 0.84 | 0.87 | 0.89 |
| SMOTEENN | RNN | 21 | 0.63 | 0.67 | 0.40 | 0.63 | 0.60 | 0.68 | 0.71 |
| SMOTEENN | GCN | 21 | 0.80 | 0.77 | 0.66 | 0.77 | 0.77 | 0.94 | 0.80 |
| SMOTEENN | Stacking | 21 | 0.54 | 0.53 | 0.51 | 0.53 | 0.53 | 1.00 | 0.51 |
| SMOTEENN | Deep Packet | 21 | 0.64 | 0.66 | 0.39 | 0.60 | 0.61 | 0.62 | 0.58 |
| SMOTEENN | FlowPic | 21 | 0.49 | 0.66 | 0.33 | 0.58 | 0.46 | 0.51 | 0.47 |
| SMOTEENN | FS-Net | 21 | 0.79 | 0.80 | 0.53 | 0.76 | 0.72 | 0.80 | 0.79 |

Table 13
Classification Performance in Each Category of Metaverse Datasets under the 1D-CNN Model after Resampling at the Flow Level

| Category | P. | R. | F. |
|---|---|---|---|
| BigscreenTheatre_60FPS | 0.75 | 0.6 | 0.67 |
| BigscreenTheatre_90FPS | 0.75 | 0.9 | 0.82 |
| BigscreenTheatre_120FPS | 0.5 | 1 | 0.67 |
| DiRTRally2.0_60FPS | 1 | 0.5 | 0.67 |
| DiRTRally2.0_90FPS | 0.6 | 1 | 0.75 |
| DiRTRally2.0_120FPS | 0 | 0 | 0 |
| Hellblade_60FPS | 1 | 0.5 | 0.67 |
| Hellblade_90FPS | 0 | 0 | 0 |
| Hellblade_120FPS | 0.67 | 1 | 0.8 |
| RealityMixer+SteamVRHome_60FPS | 0.67 | 1 | 0.8 |
| RealityMixer+SteamVRHome_90FPS | 0.89 | 0.73 | 0.8 |
| RealityMixer+SteamVRHome_120FPS | 0.75 | 0.6 | 0.67 |
| SolarSystemAR_60FPS | 0.2 | 0.33 | 0.25 |
| SolarSystemAR_90FPS | 0 | 0 | 0 |
| SolarSystemAR_120FPS | 1 | 1 | 1 |
| TheLab_60FPS | 0.73 | 0.73 | 0.73 |
| TheLab_90FPS | 0.67 | 1 | 0.8 |
| TheLab_120FPS | 0.67 | 0.8 | 0.73 |
| VRChat_60FPS | 1 | 0.75 | 0.86 |
| VRChat_90FPS | 0 | 0 | 0 |
| VRChat_120FPS | 0 | 0 | 0 |

Table 14
Classification Performance Summary of Streaming VLS Simulation Datasets at the Flow Level

| Resampling | Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|---|
| / | 1D-CNN | 78 | 0.74 | 0.73 | 0.73 | 0.73 | 0.73 | 0.78 | 0.77 |
| / | RNN | 78 | 0.68 | 0.67 | 0.66 | 0.67 | 0.67 | 0.69 | 0.69 |
| / | GCN | 78 | 0.59 | 0.58 | 0.55 | 0.58 | 0.58 | 0.58 | 0.58 |
| / | Stacking | 78 | 0.90 | 0.90 | 0.89 | 0.90 | 0.90 | 0.98 | 0.90 |
| / | Deep Packet | 78 | 0.68 | 0.68 | 0.67 | 0.68 | 0.67 | 0.68 | 0.68 |
| / | FlowPic | 78 | 0.65 | 0.64 | 0.63 | 0.64 | 0.64 | 0.65 | 0.65 |
| / | FS-Net | 78 | 0.72 | 0.71 | 0.71 | 0.71 | 0.71 | 0.72 | 0.71 |
| SMOTEENN | 1D-CNN | 78 | 0.91 | 0.90 | 0.87 | 0.90 | 0.90 | 0.92 | 0.92 |

Table 14 (continued)

| | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| SMOTEENN | RNN | 78 | 0.84 | 0.84 | 0.78 | 0.84 | 0.84 | 0.84 | 0.84 |
| SMOTEENN | GCN | 78 | 0.70 | 0.64 | 0.57 | 0.64 | 0.64 | 0.69 | 0.69 |
| SMOTEENN | Stacking | 78 | 0.98 | 0.98 | 0.97 | 0.98 | 0.98 | 1.00 | 0.98 |
| SMOTEENN | Deep Packet | 78 | 0.82 | 0.82 | 0.75 | 0.82 | 0.82 | 0.82 | 0.83 |
| SMOTEENN | FlowPic | 78 | 0.79 | 0.78 | 0.71 | 0.78 | 0.78 | 0.78 | 0.79 |
| SMOTEENN | FS-Net | 78 | 0.87 | 0.86 | 0.81 | 0.86 | 0.86 | 0.86 | 0.86 |

Table 15

Classification Performance in Each Category of Streaming VLS Simulation Datasets under the Stacking Model after Resampling at the Flow Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 144P30FPS_DASH_VLS_Download | 0.98 | 0.97 | 0.98 | 720P60FPS_RTMP_VLS_Download | 1 | 1 | 1 |
| 144P30FPS_HLS_VLS_Download | 0.9 | 0.9 | 0.9 | 720P60FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 144P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 720P60FPS_WebRTC_VLS_Download | 0.99 | 1 | 1 |
| 144P30FPS_RTMP_VLS_Download | 0.99 | 1 | 1 | 1080P30FPS_DASH_VLS_Download | 0.97 | 0.95 | 0.96 |
| 144P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 1080P30FPS_HLS_VLS_Download | 0.95 | 0.98 | 0.97 |
| 144P30FPS_WebRTC_VLS_Download | 1 | 1 | 1 | 1080P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 240P30FPS_DASH_VLS_Download | 0.97 | 0.97 | 0.97 | 1080P30FPS_RTMP_VLS_Download | 0.99 | 1 | 1 |
| 240P30FPS_HLS_VLS_Download | 0.89 | 0.85 | 0.87 | 1080P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 240P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 1080P30FPS_WebRTC_VLS_Download | 0.91 | 0.92 | 0.91 |
| 240P30FPS_RTMP_VLS_Download | 1 | 0.99 | 1 | 1080P60FPS_DASH_VLS_Download | 0.92 | 0.95 | 0.93 |
| 240P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 1080P60FPS_HLS_VLS_Download | 0.88 | 0.84 | 0.86 |
| 240P30FPS_WebRTC_VLS_Download | 1 | 0.99 | 0.99 | 1080P60FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 360P30FPS_DASH_VLS_Download | 1 | 0.98 | 0.99 | 1080P60FPS_RTMP_VLS_Download | 1 | 0.99 | 0.99 |
| 360P30FPS_HLS_VLS_Download | 0.85 | 0.88 | 0.87 | 1080P60FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 360P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 1080P60FPS_WebRTC_VLS_Download | 0.92 | 0.85 | 0.88 |
| 360P30FPS_RTMP_VLS_Download | 1 | 1 | 1 | 1440P30FPS_DASH_VLS_Download | 0.95 | 0.93 | 0.94 |
| 360P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 1440P30FPS_HLS_VLS_Download | 0.81 | 0.86 | 0.84 |
| 360P30FPS_WebRTC_VLS_Download | 0.98 | 0.99 | 0.98 | 1440P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 480P30FPS_DASH_VLS_Download | 0.98 | 0.97 | 0.98 | 1440P30FPS_RTMP_VLS_Download | 1 | 1 | 1 |
| 480P30FPS_HLS_VLS_Download | 0.89 | 0.89 | 0.89 | 1440P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 480P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 1440P30FPS_WebRTC_VLS_Download | 0.93 | 0.93 | 0.93 |
| 480P30FPS_RTMP_VLS_Download | 1 | 0.99 | 0.99 | 1440P60FPS_DASH_VLS_Download | 0.94 | 0.95 | 0.94 |
| 480P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 1440P60FPS_HLS_VLS_Download | 0.82 | 0.88 | 0.85 |
| 480P30FPS_WebRTC_VLS_Download | 1 | 1 | 1 | 1440P60FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 540P30FPS_DASH_VLS_Download | 0.96 | 0.98 | 0.97 | 1440P60FPS_RTMP_VLS_Download | 1 | 1 | 1 |
| 540P30FPS_HLS_VLS_Download | 0.98 | 0.96 | 0.97 | 1440P60FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 540P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 1440P60FPS_WebRTC_VLS_Download | 0.9 | 0.93 | 0.91 |
| 540P30FPS_RTMP_VLS_Download | 1 | 1 | 1 | 2160P30FPS_DASH_VLS_Download | 0.96 | 0.98 | 0.97 |
| 540P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 2160P30FPS_HLS_VLS_Download | 0.88 | 0.82 | 0.85 |
| 540P30FPS_WebRTC_VLS_Download | 0.92 | 0.97 | 0.94 | 2160P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 720P30FPS_DASH_VLS_Download | 0.98 | 0.96 | 0.97 | 2160P30FPS_RTMP_VLS_Download | 1 | 0.99 | 1 |
| 720P30FPS_HLS_VLS_Download | 1 | 1 | 1 | 2160P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 720P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 2160P30FPS_WebRTC_VLS_Download | 0.93 | 0.9 | 0.91 |
| 720P30FPS_RTMP_VLS_Download | 0.99 | 1 | 1 | 2160P60FPS_DASH_VLS_Download | 0.98 | 0.99 | 0.99 |
| 720P30FPS_RTMP_VLS_Upload | 1 | 1 | 1 | 2160P60FPS_HLS_VLS_Download | 0.9 | 0.87 | 0.89 |
| 720P30FPS_WebRTC_VLS_Download | 0.99 | 0.98 | 0.98 | 2160P60FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 |
| 720P60FPS_DASH_VLS_Download | 0.96 | 0.97 | 0.97 | 2160P60FPS_RTMP_VLS_Download | 1 | 1 | 1 |
| 720P60FPS_HLS_VLS_Download | 1 | 1 | 1 | 2160P60FPS_RTMP_VLS_Upload | 1 | 1 | 1 |
| 720P60FPS_HTTP-FLV_VLS_Download | 1 | 1 | 1 | 2160P60FPS_WebRTC_VLS_Download | 1 | 1 | 1 |

## 3.3. Payload-Level Classification Performance

In the payload-level classification task, all byte payload samples have been converted into graph-based representations. As a result, SMOTEENN resampling cannot be applied because both SMOTE and ENN are designed for tabular datasets. These methods assume that each sample is a fixed-length vector and that distances between samples are meaningful, typically using Euclidean distance. In graph data, where each sample has a unique structure composed of nodes and edges, interpolation between samples is not straightforward. Therefore, techniques

like SMOTE and ENN are not applicable to graph-based datasets. In all interactive media network traffic datasets at the payload level, even the 144P_30FPS_DASH_VLS_Download category with the smallest number of PCAP packets, still contains more than 5000 packets. This ensures that there is a sufficient volume of payload-level samples; synthetic augmentation is not required. All eight classification models were evaluated on the payload-level task. With the exception of the 2D-CNN model, which operates directly on the original graph structures, all other models convert the graphs into one-dimensional feature vectors and apply PCA for dimensionality reduction. As shown in Table 18, the 2D-CNN model achieved the highest classification accuracy of 0.81 on the test set. However, both the training and validation accuracies were 0.98, suggesting overfitting. The model memorizes the training and validation data instead of learning patterns that generalize well. The issue is more evident in the 1D-CNN model, where the gap between training and test accuracy exceeds 28 percent.

The other models show similar macro F1-scores, generally around 0.7. The 2D-CNN model had a macro F1-score of approximately 0.82, which is about 15 percent higher than the others. Although the dataset does not suffer from class imbalance, a major challenge is the selection of appropriate feature fusion techniques. Ineffective fusion strategies prevent the model from learning useful classification patterns or defining clear decision boundaries. Table 19 provides a breakdown of performance by category. For example, the 480P_30FPS_VLS_Facebook category had an F1-score of only 0.26, and the 720P_30FPS_VOD_Facebook category had an even lower score of 0.14. In both cases, the feature fusion methods included random pixel scrambling and random pixel shifting, which did not improve performance. Similarly, the 360P_30FPS_VLS_TikTok and 540P_30FPS_VLS_TikTok categories had F1-scores below 0.4. These results were obtained using fusion techniques such as applying random Gaussian blur, alternating small-kernel sharpening, two-dimensional image filtering, and adding random tile patterns. More than 20 categories had F1-scores below 0.80. This indicates that using stochastic noise or small graph alterations through permutations and combinations provides only limited benefit for improving classification performance.

For the Metaverse dataset, the overall classification performance at the payload level is significantly better than the flow-level results, as shown in Table 20 when compared to Table 14. The 2D-CNN model achieved a classification accuracy as high as 0.81, and the macro F1-score reached 0.82. However, there are signs of overfitting, as the training and validation accuracy are notably higher than the test performance. Despite this, the 2D-CNN still outperformed all other models by more than 10 percent in overall classification accuracy. According to Table 21, which presents classification performance by category, only four categories, including RealityMixer+SteamVRHome_90FPS, SolarSystemAR_60FPS, TheLab_60FPS, and VRChat_60FPS, had F1-scores below 0.6. The remaining categories all achieved F1-scores above 0.8, indicating relatively strong performance across most of the dataset. Several factors contribute to the classification challenges in the lower-performing categories. The original traffic capture was limited to a bandwidth of 15 Mbps and used the WebRTC streaming protocol over TCP. Due to TSO or GSO being enabled, many TCP segments stored in the PCAP files were unfragmented and large, with some payloads exceeding 10,000 bytes. However, only the first 1250 bytes of each packet were used to generate graph samples, and the remaining payload

data was omitted. Additionally, a significant number of packets had payloads smaller than 100 bytes. Although the dataset contained a sufficient number of samples, the individual graph samples often lacked informative or effective features, which made accurate classification more difficult. Moreover, many categories relied on similar feature fusion methods, such as random rotation, vertical flipping, Gaussian noise, and Gaussian blur with a random kernel size. These standard augmentation techniques were not able to clearly enhance classification-relevant features, primarily because the original features themselves were limited in quality and informativeness.

The Streaming VLS Simulation dataset shows relatively low classification performance at the payload level. As reported in Table 22, the highest overall classification accuracy reaches only 0.75 under the 2D-CNN model across 78 categories. This is over 20 % lower than the time-series classification accuracy, where the 1D-CNN model achieves 0.91, and the Stacking model reaches 0.98 at the flow level. Analysis of the raw PCAP files highlights protocol-related differences that affect classification performance. WebRTC, which is based on UDP, contains over 80 percent of packets with payloads smaller than 500 bytes. This results in a lack of effective, high-quality samples for graph construction. By contrast, protocols such as DASH, HLS, HTTP-FLV, and RTMP operate over TCP. These protocols often include very large payloads, with some segments exceeding 10,000 bytes and reaching up to 65,549 bytes in the DASH protocol. However, due to storage limitations (1250 bytes) during graph conversion, only approximately 2 percent of the payload bytes are retained, eliminating most of the useful information. Additionally, more than 0.5 percent of packets have payloads smaller than 100 bytes. As shown in Table 23, most DASH-related categories have F1-scores below 0.5. These issues stem from TCP offloading mechanisms, where the operating system and network interface card aggregate large TCP segments for efficiency before splitting them into smaller packets during transmission. When captured on the sender side, especially on loopback interfaces, these large segments appear in the PCAP files even though they are not transmitted as such on the network. In contrast, UDP-based protocols such as WebRTC do not support these offload mechanisms and therefore do not exhibit similar behavior. In the Metaverse dataset, WebRTC is sometimes configured to fall back to TCP due to network restrictions. This fallback leads to similar large TCP segments when TSO or GSO is enabled. Although WebRTC is fundamentally UDP-based, real-world deployments often rely on TCP-based fallback paths, which result in the same offloading effects observed in DASH and HLS traffic. When using 2000 randomly selected samples, the limited payload information in small packets continues to obstruct correct classification. Even after applying graph-based feature fusion techniques, the results remain poor due to the insufficient quality of the original graph samples. Compared to the Live Streaming dataset, the VLS Simulation dataset includes more TCP-based traffic and a higher proportion of low-quality byte payload samples.

In summary, the classification performance at the payload level varies significantly across interactive media applications. The lack of consistent structural characteristics between traffic types makes it difficult to generalize models. Although a variety of feature fusion techniques were applied to extract common patterns, the limited information in many graph samples reduced their effectiveness. During dataset generation, automatic zero-padding was used to fill

insufficient payloads, which contributed little useful information to the graphs. As a result, standard augmentation methods such as shape rotation and the addition of random noise were not effective in improving classification accuracy. These methods are not well-suited to the structure of the data, and more advanced feature fusion strategies need to be explored.

Table 16
Classification Performance Summary of Live Streaming Datasets at the Payload Level

| Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|
| 2D-CNN | 95 | 0.82 | 0.81 | 0.82 | 0.81 | 0.81 | 0.98 | 0.98 |
| 1D-CNN | 95 | 0.71 | 0.71 | 0.71 | 0.71 | 0.71 | 0.99 | 0.99 |
| RNN | 95 | 0.73 | 0.73 | 0.72 | 0.73 | 0.73 | 0.77 | 0.78 |
| GCN | 95 | 0.70 | 0.69 | 0.68 | 0.69 | 0.69 | 0.70 | 0.68 |
| Stacking | 95 | 0.74 | 0.74 | 0.74 | 0.74 | 0.74 | 0.99 | 0.74 |
| Deep Packet | 95 | 0.73 | 0.72 | 0.72 | 0.72 | 0.72 | 0.79 | 0.80 |
| FlowPic | 95 | 0.72 | 0.71 | 0.70 | 0.71 | 0.71 | 0.73 | 0.73 |
| FS-Net | 95 | 0.75 | 0.74 | 0.74 | 0.74 | 0.74 | 0.77 | 0.77 |

Table 17
Classification Performance in Each Category of Live Streaming Datasets under the 2D-CNN Model at the Payload Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 480P_3Mbps_N3_10Mhz | 0.83 | 0.69 | 0.75 | VPN_Vimeo | 0.99 | 0.98 | 0.98 |
| 480P_3Mbps_N78_10Mhz | 1 | 1 | 1 | VPN_YouTube | 0.52 | 0.52 | 0.52 |
| 480P_3Mbps_N78_20Mhz | 1 | 1 | 1 | 160P_30FPS_VoD_Twitch | 0.99 | 0.98 | 0.99 |
| 720P_6Mbps_N3_10Mhz | 1 | 1 | 1 | 160P_30FPS_VLS_Twitch | 1 | 1 | 1 |
| 720P_6Mbps_N78_10Mhz | 1 | 1 | 1 | 360P_30FPS_VoD_Twitch | 0.48 | 0.47 | 0.48 |
| 720P_6Mbps_N78_20Mhz | 1 | 1 | 1 | 360P_30FPS_VLS_Twitch | 0.99 | 1 | 0.99 |
| 1080P_10Mbps_N3_10Mhz | 1 | 0.98 | 0.99 | 480P_30FPS_VoD_Twitch | 0.98 | 0.96 | 0.97 |
| 1080P_10Mbps_N78_10Mhz | 1 | 1 | 1 | 480P_30FPS_VLS_Twitch | 0.83 | 0.94 | 0.88 |
| 1080P_10Mbps_N78_20Mhz | 0.95 | 0.99 | 0.97 | 720P_30FPS_VoD_Twitch | 1 | 1 | 1 |
| 2160P_20Mbps_N3_10Mhz | 0.79 | 0.92 | 0.85 | 720P_60FPS_VoD_Twitch | 1 | 0.97 | 0.98 |
| 2160P_20Mbps_N78_20Mhz | 1 | 0.99 | 0.99 | 720P_30FPS_VLS_Twitch | 0.67 | 0.84 | 0.74 |
| 4320P_40Mbps_N3_10Mhz | 0.99 | 0.99 | 0.99 | 720P_60FPS_VLS_Twitch | 0.72 | 0.66 | 0.69 |
| 4320P_40Mbps_N78_20Mhz | 0.99 | 0.99 | 0.99 | 1080P_30FPS_VoD_Twitch | 0.95 | 1 | 0.98 |
| 2160P_30FPS_VoD_BiliBili | 1 | 0.99 | 0.99 | 1080P_60FPS_VoD_Twitch | 1 | 0.99 | 1 |
| 2160P_60FPS_VoD_BiliBili | 1 | 1 | 1 | 1080P_30FPS_VLS_Twitch | 0.92 | 0.79 | 0.85 |
| 2160P_120FPS_VoD_BiliBili | 0.85 | 0.85 | 0.85 | 1080P_60FPS_VLS_Twitch | 0.99 | 0.96 | 0.97 |
| 2160P_60FPS_VLS_BiliBili | 0.69 | 0.67 | 0.68 | 240P_30FPS_VoD_Vimeo | 0.74 | 0.8 | 0.77 |
| 2160P_80FPS_VLS_BiliBili | 0.95 | 0.96 | 0.96 | 360P_30FPS_VoD_Vimeo | 1 | 1 | 1 |
| 4320P_30FPS_VoD_BiliBili | 0.87 | 0.87 | 0.87 | 540P_30FPS_VoD_Vimeo | 0.49 | 0.49 | 0.49 |
| 360P_30FPS_VoD_BiliBili | 0.9 | 0.89 | 0.89 | 720P_30FPS_VoD_Vimeo | 0.34 | 0.38 | 0.36 |
| 480P_30FPS_VoD_BiliBili | 0.95 | 0.87 | 0.91 | 1080P_30FPS_VoD_Vimeo | 0.65 | 0.66 | 0.66 |
| 480P_30FPS_VLS_BiliBili | 1 | 1 | 1 | 1440P_30FPS_VoD_YouTube | 0.5 | 0.48 | 0.49 |
| 720P_30FPS_VoD_BiliBili | 0.55 | 0.47 | 0.51 | 1440P_60FPS_VoD_YouTube | 0.41 | 0.31 | 0.35 |
| 720P_30FPS_VLS_BiliBili | 1 | 0.97 | 0.98 | 1440P_30FPS_VLS_YouTube | 0.99 | 1 | 0.99 |
| 1080P_30FPS_VoD_BiliBili | 1 | 1 | 1 | 1440P_60FPS_VLS_YouTube | 0.26 | 0.29 | 0.27 |
| 1080P_60FPS_VoD_BiliBili | 0.99 | 1 | 1 | 2160P_30FPS_VoD_YouTube | 0.27 | 0.43 | 0.33 |
| 1080P_60FPS_VLS_BiliBili | 0.89 | 0.94 | 0.92 | 2160P_60FPS_VoD_YouTube | 0.81 | 0.68 | 0.74 |
| 1440P_60FPS_VLS_BiliBili | 1 | 0.99 | 0.99 | 2160P_30FPS_VLS_YouTube | 0.97 | 0.99 | 0.98 |
| 144P_30FPS_VOD_Facebook | 0.98 | 1 | 0.99 | 2160P_60FPS_VLS_YouTube | 0.99 | 0.99 | 0.99 |
| 144P_30FPS_VLS_Facebook | 0.8 | 0.72 | 0.76 | 4320P_30FPS_VoD_YouTube | 0.99 | 1 | 0.99 |
| 240P_30FPS_VLS_Facebook | 0.97 | 0.99 | 0.98 | 4320P_60FPS_VoD_YouTube | 0.96 | 0.94 | 0.95 |
| 360P_30FPS_VOD_Facebook | 0.98 | 0.91 | 0.94 | 144P_30FPS_VoD_YouTube | 0.78 | 1 | 0.88 |
| 360P_30FPS_VLS_Facebook | 1 | 1 | 1 | 144P_30FPS_VLS_YouTube | 0.97 | 1 | 0.98 |
| 480P_30FPS_VOD_Facebook | 1 | 1 | 1 | 240P_30FPS_VoD_YouTube | 1 | 1 | 1 |
| 480P_30FPS_VLS_Facebook | 0.27 | 0.26 | 0.26 | 240P_30FPS_VLS_YouTube | 1 | 1 | 1 |
| 540P_30FPS_VOD_Facebook | 0.62 | 0.65 | 0.64 | 360P_30FPS_VoD_YouTube | 1 | 0.89 | 0.94 |
| 720P_30FPS_VOD_Facebook | 0.17 | 0.12 | 0.14 | 360P_30FPS_VLS_YouTube | 0.96 | 0.86 | 0.91 |
| 720P_30FPS_VLS_Facebook | 0.9 | 0.76 | 0.82 | 480P_30FPS_VoD_YouTube | 0.83 | 0.75 | 0.79 |
| 1080P_30FPS_VOD_Facebook | 0.77 | 0.87 | 0.82 | 480P_30FPS_VLS_YouTube | 0.51 | 0.46 | 0.48 |
| 1080P_30FPS_VLS_Facebook | 0.99 | 0.99 | 0.99 | 720P_30FPS_VoD_YouTube | 0.34 | 0.4 | 0.37 |
| 360P_30FPS_VLS_TikTok | 0.3 | 0.39 | 0.34 | 720P_60FPS_VoD_YouTube | 1 | 0.99 | 1 |
| 540P_30FPS_VLS_TikTok | 0.4 | 0.37 | 0.38 | 720P_30FPS_VLS_YouTube | 0.99 | 0.99 | 0.99 |

Table 17 (continued)

| | | | | | | | |
|---|---|---|---|---|---|---|---|
| 720P_30FPS_VLS_TikTok | 0.85 | 1 | 0.92 | 720P_60FPS_VLS_YouTube | 0.72 | 0.63 | 0.67 |
| 720P_60FPS_VLS_TikTok | 0.56 | 0.64 | 0.6 | 1080P_30FPS_VoD_YouTube | 0.51 | 0.47 | 0.49 |
| 1080P_30FPS_VLS_TikTok | 0.31 | 0.2 | 0.24 | 1080P_60FPS_VoD_YouTube | 1 | 1 | 1 |
| 1080P_60FPS_VLS_TikTok | 0.62 | 0.58 | 0.6 | 1080P_30FPS_VLS_YouTube | 1 | 1 | 1 |
| Tor_Vimeo | 0.62 | 0.6 | 0.61 | 1080P_60FPS_VLS_YouTube | 0.23 | 0.3 | 0.26 |
| Tor_YouTube | 1 | 0.9 | 0.95 | | | | |

Table 18

Classification Performance Summary of Metaverse Datasets at the Payload Level

| Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|
| 2D-CNN | 21 | 0.84 | 0.84 | 0.85 | 0.84 | 0.84 | 1.00 | 1.00 |
| 1D-CNN | 21 | 0.79 | 0.79 | 0.79 | 0.79 | 0.79 | 1.00 | 0.99 |
| RNN | 21 | 0.79 | 0.78 | 0.79 | 0.78 | 0.78 | 0.89 | 0.89 |
| GCN | 21 | 0.73 | 0.73 | 0.73 | 0.73 | 0.73 | 0.78 | 0.72 |
| Stacking | 21 | 0.82 | 0.82 | 0.82 | 0.82 | 0.82 | 1.00 | 0.82 |
| Deep Packet | 21 | 0.79 | 0.79 | 0.80 | 0.79 | 0.79 | 0.95 | 0.95 |
| FlowPic | 21 | 0.78 | 0.78 | 0.78 | 0.78 | 0.78 | 0.81 | 0.81 |
| FS-Net | 21 | 0.81 | 0.81 | 0.81 | 0.81 | 0.80 | 0.87 | 0.87 |

Table 19

Classification Performance in Each Category of Metaverse Datasets under the 2D-CNN Model at the Payload Level

| Category | P. | R. | F. |
|---|---|---|---|
| BigscreenTheatre_60FPS | 0.77 | 0.84 | 0.8 |
| BigscreenTheatre_90FPS | 1 | 1 | 1 |
| BigscreenTheatre_120FPS | 1 | 1 | 1 |
| DiRTRally2.0_60FPS | 0.98 | 0.99 | 0.98 |
| DiRTRally2.0_90FPS | 0.81 | 0.82 | 0.82 |
| DiRTRally2.0_120FPS | 0.93 | 0.94 | 0.93 |
| Hellblade_60FPS | 0.81 | 0.87 | 0.84 |
| Hellblade_90FPS | 0.99 | 1 | 0.99 |
| Hellblade_120FPS | 0.94 | 0.9 | 0.92 |
| RealityMixer+SteamVRHome_60FPS | 0.84 | 0.93 | 0.88 |
| RealityMixer+SteamVRHome_90FPS | 0.54 | 0.55 | 0.54 |
| RealityMixer+SteamVRHome_120FPS | 0.89 | 0.85 | 0.87 |
| SolarSystemAR_60FPS | 0.52 | 0.6 | 0.56 |
| SolarSystemAR_90FPS | 0.96 | 0.92 | 0.94 |
| SolarSystemAR_120FPS | 0.9 | 0.85 | 0.87 |
| TheLab_60FPS | 0.66 | 0.63 | 0.64 |
| TheLab_90FPS | 0.92 | 0.8 | 0.86 |
| TheLab_120FPS | 0.82 | 0.8 | 0.81 |
| VRChat_60FPS | 0.61 | 0.64 | 0.62 |
| VRChat_90FPS | 0.88 | 0.74 | 0.8 |
| VRChat_120FPS | 1 | 0.99 | 1 |

Table 20
Classification Performance Summary of Streaming VLS Simulation Datasets at the Payload Level

| Model | Category | P.W | R.W | F.A | F.I | A.Test | A.Train | A.Validate |
|---|---|---|---|---|---|---|---|---|
| 2D-CNN | 78 | 0.76 | 0.75 | 0.75 | 0.75 | 0.75 | 0.99 | 0.99 |
| 1D-CNN | 78 | 0.68 | 0.67 | 0.67 | 0.67 | 0.67 | 0.98 | 0.98 |
| RNN | 78 | 0.68 | 0.65 | 0.65 | 0.65 | 0.65 | 0.71 | 0.71 |
| GCN | 78 | 0.60 | 0.57 | 0.57 | 0.57 | 0.57 | 0.61 | 0.56 |
| Stacking | 78 | 0.65 | 0.65 | 0.65 | 0.65 | 0.65 | 0.99 | 0.65 |
| Deep Packet | 78 | 0.64 | 0.63 | 0.63 | 0.63 | 0.64 | 0.77 | 0.77 |
| FlowPic | 78 | 0.65 | 0.63 | 0.62 | 0.63 | 0.63 | 0.65 | 0.65 |
| FS-Net | 78 | 0.67 | 0.66 | 0.66 | 0.66 | 0.66 | 0.73 | 0.73 |

Table 21
Classification Performance in Each Category of Streaming VLS Simulation Datasets under the 2D-CNN Model at the Payload Level

| Category | P. | R. | F. | Category | P. | R. | F. |
|---|---|---|---|---|---|---|---|
| 144P30FPS_DASH_VLS_Download | 0.99 | 0.94 | 0.96 | 720P60FPS_RTMP_VLS_Download | 0.55 | 0.63 | 0.59 |
| 144P30FPS_HLS_VLS_Download | 1 | 1 | 1 | 720P60FPS_RTMP_VLS_Upload | 0.66 | 0.64 | 0.65 |
| 144P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 720P60FPS_WebRTC_VLS_Download | 0.87 | 0.95 | 0.91 |
| 144P30FPS_RTMP_VLS_Download | 0.99 | 1 | 1 | 1080P30FPS_DASH_VLS_Download | 0.11 | 0.15 | 0.13 |
| 144P30FPS_RTMP_VLS_Upload | 0.94 | 0.88 | 0.91 | 1080P30FPS_HLS_VLS_Download | 0.76 | 0.63 | 0.69 |
| 144P30FPS_WebRTC_VLS_Download | 1 | 1 | 1 | 1080P30FPS_HTTP_FLV_VLS_Download | 0.61 | 0.73 | 0.67 |
| 240P30FPS_DASH_VLS_Download | 0.96 | 0.9 | 0.93 | 1080P30FPS_RTMP_VLS_Download | 0.78 | 0.73 | 0.76 |
| 240P30FPS_HLS_VLS_Download | 0.99 | 0.96 | 0.98 | 1080P30FPS_RTMP_VLS_Upload | 0.53 | 0.57 | 0.55 |
| 240P30FPS_HTTP_FLV_VLS_Download | 0.71 | 0.77 | 0.74 | 1080P30FPS_WebRTC_VLS_Download | 0.99 | 0.99 | 0.99 |
| 240P30FPS_RTMP_VLS_Download | 0.91 | 0.83 | 0.87 | 1080P60FPS_DASH_VLS_Download | 0.41 | 0.33 | 0.37 |
| 240P30FPS_RTMP_VLS_Upload | 0.61 | 0.67 | 0.64 | 1080P60FPS_HLS_VLS_Download | 0.85 | 0.65 | 0.74 |
| 240P30FPS_WebRTC_VLS_Download | 0.99 | 1 | 1 | 1080P60FPS_HTTP_FLV_VLS_Download | 1 | 0.88 | 0.94 |
| 360P30FPS_DASH_VLS_Download | 0.92 | 0.81 | 0.86 | 1080P60FPS_RTMP_VLS_Download | 1 | 1 | 1 |
| 360P30FPS_HLS_VLS_Download | 0.99 | 0.91 | 0.95 | 1080P60FPS_RTMP_VLS_Upload | 0.76 | 0.77 | 0.76 |
| 360P30FPS_HTTP_FLV_VLS_Download | 0.52 | 0.59 | 0.55 | 1080P60FPS_WebRTC_VLS_Download | 1 | 0.99 | 0.99 |
| 360P30FPS_RTMP_VLS_Download | 0.52 | 0.61 | 0.56 | 1440P30FPS_DASH_VLS_Download | 0.83 | 0.66 | 0.74 |
| 360P30FPS_RTMP_VLS_Upload | 0.56 | 0.53 | 0.54 | 1440P30FPS_HLS_VLS_Download | 0.91 | 0.89 | 0.9 |
| 360P30FPS_WebRTC_VLS_Download | 0.96 | 0.94 | 0.95 | 1440P30FPS_HTTP_FLV_VLS_Download | 0.94 | 0.99 | 0.96 |
| 480P30FPS_DASH_VLS_Download | 0.87 | 0.82 | 0.84 | 1440P30FPS_RTMP_VLS_Download | 0.18 | 0.34 | 0.24 |
| 480P30FPS_HLS_VLS_Download | 0.85 | 0.75 | 0.8 | 1440P30FPS_RTMP_VLS_Upload | 0.33 | 0.45 | 0.38 |
| 480P30FPS_HTTP_FLV_VLS_Download | 0.86 | 0.91 | 0.89 | 1440P30FPS_WebRTC_VLS_Download | 1 | 0.99 | 0.99 |
| 480P30FPS_RTMP_VLS_Download | 1 | 0.97 | 0.98 | 1440P60FPS_DASH_VLS_Download | 0.23 | 0.19 | 0.21 |
| 480P30FPS_RTMP_VLS_Upload | 0.88 | 0.54 | 0.67 | 1440P60FPS_HLS_VLS_Download | 0.68 | 0.69 | 0.69 |
| 480P30FPS_WebRTC_VLS_Download | 1 | 1 | 1 | 1440P60FPS_HTTP_FLV_VLS_Download | 0.17 | 0.17 | 0.17 |
| 540P30FPS_DASH_VLS_Download | 0.99 | 1 | 0.99 | 1440P60FPS_RTMP_VLS_Download | 0.85 | 0.89 | 0.87 |
| 540P30FPS_HLS_VLS_Download | 0.85 | 0.71 | 0.78 | 1440P60FPS_RTMP_VLS_Upload | 0.81 | 0.78 | 0.8 |
| 540P30FPS_HTTP_FLV_VLS_Download | 0.8 | 0.76 | 0.78 | 1440P60FPS_WebRTC_VLS_Download | 1 | 1 | 1 |
| 540P30FPS_RTMP_VLS_Download | 0.77 | 0.86 | 0.81 | 2160P30FPS_DASH_VLS_Download | 0.56 | 0.66 | 0.61 |
| 540P30FPS_RTMP_VLS_Upload | 0.98 | 0.9 | 0.94 | 2160P30FPS_HLS_VLS_Download | 0.72 | 0.62 | 0.67 |
| 540P30FPS_WebRTC_VLS_Download | 0.92 | 0.89 | 0.9 | 2160P30FPS_HTTP_FLV_VLS_Download | 0.76 | 0.82 | 0.79 |
| 720P30FPS_DASH_VLS_Download | 0.65 | 0.99 | 0.79 | 2160P30FPS_RTMP_VLS_Download | 0.8 | 0.79 | 0.8 |
| 720P30FPS_HLS_VLS_Download | 0.37 | 0.45 | 0.41 | 2160P30FPS_RTMP_VLS_Upload | 0.59 | 0.53 | 0.56 |
| 720P30FPS_HTTP_FLV_VLS_Download | 1 | 1 | 1 | 2160P30FPS_WebRTC_VLS_Download | 1 | 1 | 1 |
| 720P30FPS_RTMP_VLS_Download | 0.99 | 0.99 | 0.99 | 2160P60FPS_DASH_VLS_Download | 0.14 | 0.16 | 0.15 |
| 720P30FPS_RTMP_VLS_Upload | 0.36 | 0.29 | 0.32 | 2160P60FPS_HLS_VLS_Download | 0.79 | 0.69 | 0.74 |
| 720P30FPS_WebRTC_VLS_Download | 0.97 | 0.93 | 0.95 | 2160P60FPS_HTTP_FLV_VLS_Download | 0.19 | 0.16 | 0.17 |
| 720P60FPS_DASH_VLS_Download | 0.41 | 0.36 | 0.39 | 2160P60FPS_RTMP_VLS_Download | 0.11 | 0.1 | 0.11 |
| 720P60FPS_HLS_VLS_Download | 0.85 | 0.72 | 0.78 | 2160P60FPS_RTMP_VLS_Upload | 0.82 | 0.73 | 0.77 |
| 720P60FPS_HTTP-FLV_VLS_Download | 1 | 0.95 | 0.97 | 2160P60FPS_WebRTC_VLS_Download | 1 | 1 | 1 |

# 4. CONCLUSIONS

The Doctoral Thesis presents a comprehensive approach to achieve fine-grained classification of interactive media applications network traffic across multiple granularities, including the time series, flow, and payload levels. The main achievements are summarized as follows.

1. The scope of traditional video category network traffic, which is typically limited to general Internet applications and protocol-specific datasets, has been significantly expanded. This study redefines the classification scope to encompass three representative and widely used types of interactive media applications: OTT Live Streaming, Cloud Gaming, and Metaverse applications. These categories share common features, such as high traffic volume between clients and servers, and the presence of continuous bidirectional interaction over diverse and complex network environments. The resulting traffic exhibits unique and distinguishable patterns not captured by traditional coarse-grained video classification approaches. Unlike conventional classification tasks that rely solely on simple application names, service identifiers, or protocol labels as ground-truth, this study emphasizes the need for deeper context. Conventional methods often lack clarity in the relationship between the ground-truth labels and actual traffic characteristics. Furthermore, existing research rarely provides a comprehensive analysis of the factors influencing video traffic patterns.

This study systematically investigates the impact of various video-related factors on interactive media network traffic. These include video quality parameters such as resolution, frame rate, compression format, container type, bitrate, motion complexity, content origin, and hardware capabilities of both client and server devices. In addition, it considers transmission parameters like segment duration, GOP structure, codec type, encoding preset, maximum bitrate, buffer size, and transport protocols, including TCP, UDP, and QUIC. Application-level streaming protocols examined include DASH, HLS, RTMP, RTCP, RTP, SRT, HTTP-FLV, and WebRTC. Network condition variables such as 4G, 5G, Wi-Fi, available bandwidth, QoS, QoE, and the usage of VPNs or Tor for encryption and anonymization are also considered. These factors were analyzed to determine their influence on the characteristics of interactive media network traffic. Based on the findings, traffic categories in the classification tasks were defined using combinations of the most impactful factors, such as platform, application name, resolution, frame rate, and traffic direction (uplink and downlink). This approach enabled the construction of highly detailed, fine-grained interactive media network traffic datasets. A total of over 84 GB of traffic data was collected, covering both real-world applications and idealized testing environments. The datasets include both public and proprietary sources, ensuring a diverse and representative sample of interactive media network traffic.

2. The fine-grained interactive media network traffic datasets developed in the Doctoral Thesis are characterized not only by a wide range of category diversity but also by multiple levels of classification granularity, including time series level, flow level, and payload level. Each level addresses specific use-case scenarios. When raw traffic PCAP files are unavailable,

time series datasets constructed from packets, bytes and timestamps provide valuable alternatives. These temporal representations enable the rapid identification of encrypted traffic based on timing patterns, without requiring access to raw PCAP contents. At the flow level, samples capture general traffic patterns, which may manifest as elephant or mouse-specific flows, which are particularly useful for identifying distinct behavioral characteristics of interactive media applications. These flow-level features, combined with partial DPI, contribute to more accurate classification of application types and protocols. At the payload level, classification is especially advantageous in situations with limited PCAP data capture durations or sparse session streams. Due to the bursty nature of interactive media network traffic, even brief capture periods typically yield sufficient byte-level samples. These payload samples, when transformed into graph representations, enable the extraction of both spatial and temporal features, offering a richer structure than one-dimensional data formats. The datasets constructed at all three granularities exhibit transparent, explainable preprocessing procedures and ensure reliability and reproducibility of the samples.

3. This study introduces a multi-layer 1D-CNN tailored for fine-grained traffic classification. Building upon this architecture, a modified 2D-CNN model was developed specifically for graph-based classification, along with an RNN and a GCN designed for flow-level tasks. Across all three granularities, the proposed 1D-CNN consistently outperforms existing state-of-the-art models, including Deep Packet, FlowPic, and FS-Net, demonstrating strong adaptability and robustness across diverse traffic scenarios. A Stacking model, composed of multiple meta-learners, was also implemented and found to be particularly effective at the flow level. Its ability to handle high-variance features and large sample volumes resulted in classification accuracy exceeding 1D-CNN by more than 10 % under the same conditions. At the payload level, the 2D-CNN model achieved the best performance for graph-based classification. Its advantage stems from its capacity to leverage two-dimensional spatiotemporal features, surpassing one-dimensional models by at least 10 % in overall classification accuracy. In summary, model selection is highly dependent on dataset characteristics, feature construction, and scenario-specific requirements. For the fine-grained classification of interactive media network traffic, all proposed models achieved strong and consistent performance across their respective domains.

Future research can be summarized as follows.

1. At the time series-level and flow-level classification stages, a persistent issue is the imbalance in the volume of samples across different traffic categories. While this can be mitigated by actively capturing more raw traffic data or passively collecting longer flow data, synthetic data generation remains a common approach. However, the SMOTEENN method used in this work shows limitations when applied to small datasets, as it may only execute the ENN component without generating new synthetic samples. Therefore, more advanced data augmentation techniques need to be explored for better handling of imbalanced datasets.

2. In the time series-level classification task, only packet and byte vectors were used to construct the samples through the proposed universal method in Algorithm 4. Further research includes examining how applying other elements in Algorithm 4 and the corresponding time series distributions affect the classification results. Notably, not all categories demonstrate a clear increase in traffic volume with higher video resolution or frame rate. Some categories show irregular packet bursts and high noise levels in their time series, which negatively impact classification. Applying appropriate filtering methods may help reduce the influence of these noise patterns.

3. For payload-level classification, the current method pads packet payloads smaller than 1250 bytes with zeros and truncates those larger than the threshold. This results in considerable data loss, especially for large payloads associated with TCP-based streaming protocols using features like TSO. A potential improvement involves reducing the grayscale graph size from $100 \times 100$ to $32 \times 32$, corresponding to 128 bytes. For payloads exceeding this size, additional graphs can be generated to retain all relevant information. Furthermore, existing graph feature fusion methods have shown limited effectiveness in improving fine-grained classification accuracy. More advanced fusion strategies are needed to better leverage structural and contextual information in payload samples.

4. Both one-dimensional temporal and two-dimensional spatiotemporal features used in training can be standardized and stored as pre-trained representations to enable online, real-time classification of interactive media traffic. Beyond classification, these features may also be valuable for tasks such as traffic prediction, anomaly detection, and identifying malicious activity disguised as normal traffic. These directions offer practical value and merit further investigation. In addition, interactive media network traffic patterns after training can be used to identify anonymized and encrypted traffic. This is particularly relevant as generative artificial intelligence (AI) applications such as text-to-video generation, GPU-based deep learning super sampling (DLSS), and motion interpolation for gaming or metaverse environments become more prevalent. These emerging technologies not only improve users' QoE but also contribute to significant increases in traffic volume. In future heterogeneous networks, such traffic patterns may provide valuable prior knowledge to support further research.

# BIBLIOGRAPHY


1. A., P. *IDC: Expect 175 Zettabytes of Data Worldwide by 2025*, *Network World*; 2018; Available online: https://www.networkworld.com/article/3325397/idc-expect-175-zettabytes-of-data-worldwide-by-2025.html
2. Abbasi, M.; Shahraki, A.; Taherkordi, A. Deep Learning for Network Traffic Monitoring and Analysis (NTMA): A Survey. *Computer Communications* **2021**, *170*, 19–41, doi:10.1016/j.comcom.2021.01.021.
3. Fowdur, T. P.; Babooram, L. *Machine Learning for Network Traffic and Video Quality Analysis: Develop and Deploy Applications Using JavaScript and Node. Js*; Apress: Berkeley, CA, 2024; ISBN 979-8-8688-0353-6.
4. Nguyen, T. T. T.; Armitage, G. A Survey of Techniques for Internet Traffic Classification Using Machine Learning. *IEEE Commun. Surv. Tutorials* **2008**, *10*, 56–76, doi:10.1109/SURV.2008.080406.
5. Yan, J.; Yuan, J. A Survey of Traffic Classification in Software-Defined Networks. In Proceedings of the 2018 1st IEEE International Conference on Hot Information-Centric Networking (HotICN); IEEE: Shenzhen, August 2018; pp. 200–206.
6. Wang, P.; Chen, X.; Ye, F.; Sun, Z. A Survey of Techniques for Mobile Service Encrypted Traffic Classification Using Deep Learning. *IEEE Access* **2019**, *7*, 54024–54033, doi:10.1109/ACCESS.2019.2912896.
7. Tahaei, H.; Afifi, F.; Asemi, A.; Zaki, F.; Anuar, N. B. The Rise of Traffic Classification in IoT Networks: A Survey. *Journal of Network and Computer Applications* **2020**, *154*, 102538, doi:10.1016/j.jnca.2020.102538.
8. Gunavathie; UmaMaheswariS. A Survey on Traffic Prediction and Classification in SDN. In *Advances in Parallel Computing*; Jude Hemanth, D., Ambeth Kumar, V. D., Malathi, S., Eds.; IOS Press, 2020 ISBN 978-1-64368-102-3.
9. Zhao, J.; Jing, X.; Yan, Z.; Pedrycz, W. Network Traffic Classification for Data Fusion: A Survey. *Information Fusion* **2021**, *72*, 22–47, doi:10.1016/j.inffus.2021.02.009.
10. Murshed, M. G. S.; Murphy, C.; Hou, D.; Khan, N.; Ananthanarayanan, G.; Hussain, F. Machine Learning at the Network Edge: A Survey. *ACM Comput. Surv.* **2022**, *54*, 1–37, doi:10.1145/3469029.
11. Adeleke, O. A.; Bastin, N.; Gurkan, D. Network Traffic Generation: A Survey and Methodology. *ACM Comput. Surv.* **2023**, *55*, 1–23, doi:10.1145/3488375.
12. Azab, A.; Khasawneh, M.; Alrabaee, S.; Choo, K.-K. R.; Sarsour, M. Network Traffic Classification: Techniques, Datasets, and Challenges. *Digital Communications and Networks* **2024**, *10*, 676–692, doi:10.1016/j.dcan.2022.09.009.
13. Boutaba, R.; Salahuddin, M. A.; Limam, N.; Ayoubi, S.; Shahriar, N.; Estrada-Solano, F.; Caicedo, O. M. A Comprehensive Survey on Machine Learning for Networking: Evolution, Applications and Research Opportunities. *J Internet Serv Appl* **2018**, *9*, 16, doi:10.1186/s13174-018-0087-2.
14. *VNI Complete Forecast Highlights Global Internet Users: % of Population Devices and Connections per Capita Average Speeds Average Traffic per Capita per Month Global: 2021 Forecast Highlights IP Traffic*; 2016; Available online: https://www.cisco.com/c/dam/m/en_us/solutions/service-provider/vni-forecast-highlights/pdf/Global_2021_Forecast_Highlights.pdf
15. Sandvine *How 'App QoE' Can Increase Profitability While Improving Subscriber Satisfaction*; 2023; Available online: https://www.sandvine.com/hubfs/Sandvine_Redesign_2019/Downloads/2023/PDF/BRO-Sandvine-FEB2023.pdf

16. Kwak, K. T.; Lee, S. Y.; Ham, M.; Lee, S. W. The Effects of Internet Proliferation on Search Engine and Over-the-Top Service Markets. *Telecommunications Policy* **2021**, *45*, 102146, doi:10.1016/j.telpol.2021.102146.
17. Che, X.; Ip, B.; Lin, L. A Survey of Current YouTube Video Characteristics. *IEEE MultiMedia* **2015**, *22*, 56–63, doi:10.1109/MMUL.2015.34.
18. Bilal, K.; Erbad, A. Impact of Multiple Video Representations in Live Streaming: A Cost, Bandwidth, and QoE Analysis. In Proceedings of the 2017 IEEE International Conference on Cloud Engineering (IC2E); IEEE: Vancouver, BC, Canada, April 2017; pp. 88–94.
19. Pan, W.; Cheng, G. QoE Assessment of Encrypted YouTube Adaptive Streaming for Energy Saving in Smart Cities. *IEEE Access* **2018**, *6*, 25142–25156, doi:10.1109/ACCESS.2018.2811416.
20. Bentaleb, A.; Taani, B.; Begen, A. C.; Timmerer, C.; Zimmermann, R. A Survey on Bitrate Adaptation Schemes for Streaming Media Over HTTP. *IEEE Commun. Surv. Tutorials* **2019**, *21*, 562–585, doi:10.1109/COMST.2018.2862938.
21. Bermudez, H.-F.; Martinez-Caro, J.-M.; Sanchez-Iborra, R.; Arciniegas, J. L.; Cano, M.-D. Live Video-Streaming Evaluation Using the ITU-T P.1203 QoE Model in LTE Networks. *Computer Networks* **2019**, *165*, 106967, doi:10.1016/j.comnet.2019.106967.
22. Jimenez, L. R.; Solera, M.; Toril, M. A Network-Layer QoE Model for YouTube Live in Wireless Networks. *IEEE Access* **2019**, *7*, 70237–70252, doi:10.1109/ACCESS.2019.2918433.
23. Lee, R.; Venieris, S. I.; Lane, N. D. Deep Neural Network-Based Enhancement for Image and Video Streaming Systems: A Survey and Future Directions. *ACM Comput. Surv.* **2022**, *54*, 1–30, doi:10.1145/3469094.
24. Sultan, M. T.; El Sayed, H. QoE-Aware Analysis and Management of Multimedia Services in 5G and Beyond Heterogeneous Networks. *IEEE Access* **2023**, *11*, 77679–77688, doi:10.1109/ACCESS.2023.3298556.
25. Zhang, Z.; Liu, L.; Lu, X.; Yan, Z.; Li, H. Encrypted Network Traffic Classification: A Data Driven Approach. In Proceedings of the 2020 IEEE Intl Conf on Parallel & Distributed Processing with Applications, Big Data & Cloud Computing, Sustainable Computing & Communications, Social Computing & Networking (ISPA/BDCloud/SocialCom/SustainCom); IEEE: Exeter, United Kingdom, October 2020; p. 7.
26. Zhao, Y.; Yang, Y.; Tian, B.; Yang, J.; Zhang, T.; Hu, N. Edge Intelligence-Based Identification and Classification of Encrypted Traffic of Internet of Things. *IEEE Access* **2021**, *9*, 21895–21903, doi:10.1109/ACCESS.2021.3056216.
27. Canavese, D.; Regano, L.; Basile, C.; Ciravegna, G.; Lioy, A. Data Set and Machine Learning Models for the Classification of Network Traffic Originators. *Data in Brief* **2022**, *41*, 107968, doi:10.1016/j.dib.2022.107968.
28. Bader, O.; Lichy, A.; Dvir, A.; Dubin, R.; Hajaj, C. OSF-EIMTC: An Open-Source Framework for Standardized Encrypted Internet Traffic Classification. *Computer Communications* **2024**, *213*, 271–284, doi:10.1016/j.comcom.2023.10.011.
29. Draper-Gil, G.; Lashkari, A. H.; Mamun, M. S. I.; A. Ghorbani, A. Characterization of Encrypted and VPN Traffic Using Time-Related Features. In Proceedings of the 2nd International Conference on Information Systems Security and Privacy; SCITEPRESS – Science and Technology Publications: Rome, Italy, 2016; pp. 407–414.
30. Habibi Lashkari, A.; Draper Gil, G.; Mamun, M. S. I.; Ghorbani, A. A. Characterization of Tor Traffic Using Time-Based Features. In Proceedings of the 3rd International Conference on Information Systems Security and Privacy; SCITEPRESS – Science and Technology Publications: Porto, Portugal, 2017; pp. 253–262.

31. Aceto, G.; Ciuonzo, D.; Montieri, A.; Persico, V.; Pescape, A. MIRAGE: Mobile-App Traffic Capture and Ground-Truth Creation. In Proceedings of the 2019 4th International Conference on Computing, Communications and Security (ICCCS); IEEE: Rome, Italy, October 2019; pp. 1–8.
32. Habibi Lashkari, A.; Kaur, G.; Rahali, A. DIDarknet: A Contemporary Approach to Detect and Characterize the Darknet Traffic Using Deep Image Learning. In Proceedings of the 2020 10th International Conference on Communication and Network Security; ACM: Tokyo, Japan, November 27, 2020; pp. 1–13.
33. Pham, T.-D.; Ho, T.-L.; Truong-Huu, T.; Cao, T.-D.; Truong, H.-L. MAppGraph: Mobile-App Classification on Encrypted Network Traffic Using Deep Graph Convolution Neural Networks. In Proceedings of the Annual Computer Security Applications Conference; ACM: Virtual Event USA, December 6, 2021; pp. 1025–1038.
34. Dener, M.; Al, S.; Ok, G. RFSE-GRU: Data Balanced Classification Model for Mobile Encrypted Traffic in Big Data Environment. *IEEE Access* **2023**, *11*, 21831–21847, doi:10.1109/ACCESS.2023.3251745.
35. Lan, Y.; Sun, Y.; Liu, S.-P.; Ma, Z.-Z. A Real-Time Network Traffic Analysis and QoS Management Platform. In Proceedings of the 2017 IEEE 9th International Conference on Communication Software and Networks (ICCSN); IEEE: Guangzhou, May 2017; pp. 266–270.
36. Orsolic, I.; Suznjevic, M.; Skorin-Kapov, L. YouTube QoE Estimation from Encrypted Traffic: Comparison of Test Methodologies and Machine Learning Based Models. In Proceedings of the 2018 Tenth International Conference on Quality of Multimedia Experience (QoMEX); IEEE: Cagliari, May 2018; pp. 1–6.
37. Zhang, H.; Dong, L.; Gao, G.; Hu, H.; Wen, Y.; Guan, K. DeepQoE: A Multimodal Learning Framework for Video Quality of Experience (QoE) Prediction. *IEEE Trans. Multimedia* **2020**, *22*, 3210–3223, doi:10.1109/TMM.2020.2973828.
38. Wassermann, S.; Seufert, M.; Casas, P.; Gang, L.; Li, K. ViCrypt to the Rescue: Real-Time, Machine-Learning-Driven Video-QoE Monitoring for Encrypted Streaming Traffic. *IEEE Trans. Netw. Serv. Manage.* **2020**, *17*, 2007–2023, doi:10.1109/TNSM.2020.3036497.
39. Huang, Y.-F.; Lin, C.-B.; Chung, C.-M.; Chen, C.-M. Research on QoS Classification of Network Encrypted Traffic Behavior-Based on Machine Learning. *Electronics* **2021**, *10*, 1376, doi:10.3390/electronics10121376.
40. Wu, Z.; Dong, Y.; Qiu, X.; Jin, J. Online Multimedia Traffic Classification from the QoS Perspective Using Deep Learning. *Computer Networks* **2022**, *204*, 108716, doi:10.1016/j.comnet.2021.108716.
41. Al-Fayoumi, M.; Al-Fawa'reh, M.; Nashwan, S. VPN and Non-VPN Network Traffic Classification Using Time-Related Features. *Computers, Materials & Continua* **2022**, *72*, 3091–3111, doi:10.32604/cmc.2022.025103.
42. Li, W.; Canini, M.; Moore, A. W.; Bolla, R. Efficient Application Identification and the Temporal and Spatial Stability of Classification Schema. *Computer Networks* **2009**, *53*, 790–809, doi:10.1016/j.comnet.2008.11.016.
43. Muehlstein, J.; Zion, Y.; Bahumi, M.; Kirshenboim, I.; Dubin, R.; Dvir, A.; Pele, O. Analyzing HTTPS Encrypted Traffic to Identify User's Operating System, Browser and Application. In Proceedings of the 2017 14th IEEE Annual Consumer Communications & Networking Conference (CCNC); IEEE: Las Vegas, NV, USA, January 2017; pp. 1–6.
44. Wei Wang; Ming Zhu; Xuewen Zeng; Xiaozhou Ye; Yiqiang Sheng Malware Traffic Classification Using Convolutional Neural Network for Representation Learning. In

Proceedings of the 2017 International Conference on Information Networking (ICOIN); IEEE: Da Nang, Vietnam, 2017; pp. 712–717.
45. Arestrom, E.; Carlsson, N. Early Online Classification of Encrypted Traffic Streams Using Multi-Fractal Features. In Proceedings of the IEEE INFOCOM 2019 – IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS); IEEE: Paris, France, April 2019; pp. 84–89.
46. Xiao, X.; Xiao, W.; Li, R.; Luo, X.; Zheng, H.-T.; Xia, S.-T. EBSNN: Extended Byte Segment Neural Network for Network Traffic Classification. *IEEE Trans. Dependable and Secure Comput.* **2021**, 1–1, doi:10.1109/TDSC.2021.3101311.
47. Wu, H.; Wang, L.; Cheng, G.; Hu, X. Mobile Application Encryption Traffic Classification Based On TLS Flow Sequence Network. In Proceedings of the 2021 IEEE International Conference on Communications Workshops (ICC Workshops); IEEE: Montreal, QC, Canada, June 2021; pp. 1–6.
48. Shahraki, A.; Abbasi, M.; Taherkordi, A.; Kaosar, M. Internet Traffic Classification Using an Ensemble of Deep Convolutional Neural Networks. In Proceedings of the Proceedings of the 4th FlexNets Workshop on Flexible Networks Artificial Intelligence Supported Network Flexibility and Agility; ACM: Virtual Event USA, August 23, 2021; pp. 38–43.
49. Grabs, E.; Chen, T.; Petersons, E.; Efrosinin, D.; Ipatovs, A.; Kluga, J.; Culkovs, D. Features Extraction for Live Streaming Video Classification with Deep and Convolutional Neural Networks. In Proceedings of the 2021 IEEE Microwave Theory and Techniques in Wireless Communications (MTTW); IEEE: Riga, Latvia, October 7, 2021; pp. 58–63.
50. Ren, X.; Gu, H.; Wei, W. Tree-RNN: Tree Structural Recurrent Neural Network for Network Traffic Classification. *Expert Systems with Applications* **2021**, *167*, 114363, doi:10.1016/j.eswa.2020.114363.
51. Chen, T.; Grabs, E.; Petersons, E.; Efrosinin, D.; Ipatovs, A.; Kluga, J. Encapsulated and Anonymized Network Video Traffic Classification With Generative Models. In Proceedings of the 2022 Workshop on Microwave Theory and Techniques in Wireless Communications (MTTW); IEEE: Riga, Latvia, October 5, 2022; pp. 13–18.
52. Chen, T.; Grabs, E.; Petersons, E.; Efrosinin, D.; Ipatovs, A.; Bogdanovs, N.; Rjazanovs, D. Multiclass Live Streaming Video Quality Classification Based on Convolutional Neural Networks. *Aut. Control Comp. Sci.* **2022**, *56*, 455–466, doi:10.3103/S0146411622050029.
53. Salman, O.; Elhajj, I. H.; Chehab, A.; Kayssi, A. Towards Efficient Real-Time Traffic Classifier: A Confidence Measure with Ensemble Deep Learning. *Computer Networks* **2022**, *204*, 108684, doi:10.1016/j.comnet.2021.108684.
54. Gabilondo, Á.; Fernández, Z.; Viola, R.; Martín, Á.; Zorrilla, M.; Angueira, P.; Montalbán, J. Traffic Classification for Network Slicing in Mobile Networks. *Electronics* **2022**, *11*, 1097, doi:10.3390/electronics11071097.
55. Luxemburk, J.; Čejka, T. Fine-Grained TLS Services Classification with Reject Option. *Computer Networks* **2023**, *220*, 109467, doi:10.1016/j.comnet.2022.109467.
56. Qin, T.; Cheng, G.; Wei, Y.; Yao, Z. Hier-SFL: Client-Edge-Cloud Collaborative Traffic Classification Framework Based on Hierarchical Federated Split Learning. *Future Generation Computer Systems* **2023**, *149*, 12–24, doi:10.1016/j.future.2023.07.001.
57. Chen, T.; Grabs, E.; Ipatovs, A.; Pētersons, E.; Ancāns, A. Bitrate-Based Video Traffic Classification. In Proceedings of the 2023 Photonics & Electromagnetics Research Symposium (PIERS); IEEE: Prague, Czech Republic, July 3, 2023; pp. 509–514.

58. Zhu, Y.; Tao, J.; Wang, H.; Yu, L.; Luo, Y.; Qi, T.; Wang, Z.; Xu, Y. DGNN: Accurate Darknet Application Classification Adopting Attention Graph Neural Network. *IEEE Trans. Netw. Serv. Manage.* **2024**, *21*, 1660–1671, doi:10.1109/TNSM.2023.3344580.
59. Abolfathi, M.; Inturi, S.; Banaei-Kashani, F.; Jafarian, J. H. Toward Enhancing Web Privacy on HTTPS Traffic: A Novel SuperLearner Attack Model and an Efficient Defense Approach with Adversarial Examples. *Computers & Security* **2024**, *139*, 103673, doi:10.1016/j.cose.2023.103673.
60. Liu, L.; Yu, Y.; Wu, Y.; Hui, Z.; Lin, J.; Hu, J. Method for Multi-Task Learning Fusion Network Traffic Classification to Address Small Sample Labels. *Sci. Rep.* **2024**, *14*, 2518, doi:10.1038/s41598-024-51933-8.
61. Li, Z.; Xu, X. L2-BiTCN-CNN: Spatio-Temporal Features Fusion-Based Multi-Classification Model for Various Internet Applications Identification. *Computer Networks* **2024**, *243*, 110298, doi:10.1016/j.comnet.2024.110298.
62. Wang, C.; Zhang, W.; Hao, H.; Shi, H. Network Traffic Classification Model Based on Spatio-Temporal Feature Extraction. *Electronics* **2024**, *13*, 1236, doi:10.3390/electronics13071236.
63. Park, J.-T.; Shin, C.-Y.; Baek, U.-J.; Kim, M.-S. Fast and Accurate Multi-Task Learning for Encrypted Network Traffic Classification. *Applied Sciences* **2024**, *14*, 3073, doi:10.3390/app14073073.
64. Chen, Z.; Cheng, G.; Wei, Z.; Niu, D.; Fu, N. Classify Traffic Rather Than Flow: Versatile Multi-Flow Encrypted Traffic Classification With Flow Clustering. *IEEE Trans. Netw. Serv. Manage.* **2024**, *21*, 1446–1466, doi:10.1109/TNSM.2023.3322861.
65. Alkadi, S.; Al-Ahmadi, S.; Ben Ismail, M. M. RobEns: Robust Ensemble Adversarial Machine Learning Framework for Securing IoT Traffic. *Sensors* **2024**, *24*, 2626, doi:10.3390/s24082626.
66. Wang, C.; Finamore, A.; Michiardi, P.; Gallo, M.; Rossi, D. Data Augmentation for Traffic Classification. In *Passive and Active Measurement*; Richter, P., Bajpai, V., Carisimo, E., Eds.; Lecture Notes in Computer Science; Springer Nature Switzerland: Cham, 2024; Vol. 14537, pp. 159–186 ISBN 978-3-031-56248-8.
67. Song, Q.; Liu, J.; Zhang, C.; Wang, Z.; Wei, Z.; Ma, P. Classification of IPSec VPN Encrypted Traffic Based on a Hybrid Method. In Proceedings of the 2024 2nd International Conference on Electronics, Computers and Communication Technology; ACM: Chengdu, China, October 25, 2024; pp. 204–209.
68. Gudla, R.; Vollala, S.; Srinivasa, K.G.; Amin, R. A Novel Approach for Classification of Tor and Non-Tor Traffic Using Efficient Feature Selection Methods. *Expert Systems with Applications* **2024**, *249*, 123544, doi:10.1016/j.eswa.2024.123544.
69. Chong, K. Spatiotemporal Influence Analysis Through Traffic Speed Pattern Analysis Using Spatial Classification. *Applied Sciences* **2024**, *15*, 196, doi:10.3390/app15010196.
70. Krupski, J.; Iwanowski, M.; Graniszewski, W. On the Right Choice of Data from Popular Datasets for Internet Traffic Classification. *Computer Communications* **2025**, *233*, 108068, doi:10.1016/j.comcom.2025.108068.
71. Xie, J.; Li, S.; Yun, X.; Si, C.; Yin, T. Sample Analysis and Multi-Label Classification for Malicious Sample Datasets. *Computer Networks* **2025**, *258*, 110999, doi:10.1016/j.comnet.2024.110999.
72. Zhan, M.; Yang, J.; Jia, D.; Fu, G. EAPT: An Encrypted Traffic Classification Model via Adversarial Pre-Trained Transformers. *Computer Networks* **2025**, *257*, 110973, doi:10.1016/j.comnet.2024.110973.

73. Li, Z.; Bu, L.; Wang, Y.; Ma, Q.; Tan, L.; Bu, F. Hierarchical Perception for Encrypted Traffic Classification via Class Incremental Learning. *Computers & Security* **2025**, *149*, 104195, doi:10.1016/j.cose.2024.104195.
74. Xu, S.-J.; Kong, K.-C.; Jin, X.-B.; Geng, G.-G. Unveiling Traffic Paths: Explainable Path Signature Feature-Based Encrypted Traffic Classification. *Computers & Security* **2025**, *150*, 104283, doi:10.1016/j.cose.2024.104283.
75. Han, Y.; Huang, Z.; Xu, P. Field-Road Trajectory Classification for Agricultural Machinery by Integrating Spatio-Temporal Clustering and Semantic Segmentation. *Computers and Electronics in Agriculture* **2025**, *233*, 110139, doi:10.1016/j.compag.2025.110139.
76. Liu, Z.; Wei, Q.; Song, Q.; Duan, C. Fine-Grained Encrypted Traffic Classification Using Dual Embedding and Graph Neural Networks. *Electronics* **2025**, *14*, 778, doi:10.3390/electronics14040778.
77. Zhu, L.; Zhang, Q.; Jian, X.; Yang, Y. Graph Convolutional Network for Traffic Incidents Duration Classification. *Engineering Applications of Artificial Intelligence* **2025**, *151*, 110570, doi:10.1016/j.engappai.2025.110570.
78. Abbasi, M.; López Flórez, S.; Shahraki, A.; Taherkordi, A.; Prieto, J.; Corchado, J. M. Class Imbalance in Network Traffic Classification: An Adaptive Weight Ensemble-of-Ensemble Learning Method. *IEEE Access* **2025**, *13*, 26171–26192, doi:10.1109/ACCESS.2025.3538170.
79. Lee, S.-H.; Lee, S.; Yun, I. Mosaic-Mixed Attention-Based Unexpected Traffic Scene Classification. *IEEE Access* **2025**, *13*, 15712–15722, doi:10.1109/ACCESS.2025.3531121.
80. Mezina, A.; Nurmi, J.; Ometov, A. Novel Hybrid UNet++ and LSTM Model for Enhanced Attack Detection and Classification in IoMT Traffic. *IEEE Access* **2025**, 1–1, doi:10.1109/ACCESS.2025.3553966.
81. Ouyang, G.; Guo, Y.; Lu, Y.; He, F. Incremental Learning for Network Traffic Classification Using Generative Adversarial Networks. *IEICE Trans. Inf. & Syst.* **2025**, *E108.D*, 124–136, doi:10.1587/transinf.2024EDP7129.
82. Wang, Y.; Song, L. Application and Optimization of Convolutional Neural Networks Based on Deep Learning in Network Traffic Classification and Anomaly Detection. *IJCAI* **2025**, *49*, doi:10.31449/inf.v49i14.7602.
83. Guan, X.; Zhu, S.; Wan, X.; Wu, Y. STARNeT: Multidimensional Spatial–Temporal Attention Recall Network for Accurate Encrypted Traffic Classification. *Journal of Network and Computer Applications* **2025**, *236*, 104109, doi:10.1016/j.jnca.2025.104109.
84. Li, Z.; Zhao, H.; Zhao, J.; Jiang, Y.; Bu, F. SAT-Net: A Staggered Attention Network Using Graph Neural Networks for Encrypted Traffic Classification. *Journal of Network and Computer Applications* **2025**, *233*, 104069, doi:10.1016/j.jnca.2024.104069.
85. Wang, Z.; Lin, W.; Chen, Y.; Vai, M. I. Robust Classification of Encrypted Network Services Using Convolutional Neural Networks Optimized by Information Bottleneck Method. *IEEE Access* **2025**, *13*, 36995–37005, doi:10.1109/ACCESS.2025.3545476.
86. Xu, S.; Han, J.; Liu, Y.; Liu, H.; Bai, Y. Few-Shot Traffic Classification Based on Autoencoder and Deep Graph Convolutional Networks. *Sci. Rep.* **2025**, *15*, 8995, doi:10.1038/s41598-025-94240-6.
87. Zai-jian, W.; Dong, Y.; Shi, H.; Lingyun, Y.; Pingping, T. Internet Video Traffic Classification Using QoS Features. In Proceedings of the 2016 International Conference on Computing, Networking and Communications (ICNC); IEEE: Kauai, HI, USA, February 2016; pp. 1–5.

88. Dong, Y.; Zhao, J.; Jin, J. Novel Feature Selection and Classification of Internet Video Traffic Based on a Hierarchical Scheme. *Computer Networks* **2017**, *119*, 102–111, doi:10.1016/j.comnet.2017.03.019.
89. Canovas, A.; Jimenez, J. M.; Romero, O.; Lloret, J. Multimedia Data Flow Traffic Classification Using Intelligent Models Based on Traffic Patterns. *IEEE Network* **2018**, *32*, 100–107, doi:10.1109/MNET.2018.1800121.
90. Hayashi, K.; Ooka, R.; Miyoshi, T.; Yamazaki, T. P2PTV Traffic Classification and Its Characteristic Analysis Using Machine Learning. In Proceedings of the 2019 20th Asia-Pacific Network Operations and Management Symposium (APNOMS); IEEE: Matsue, Japan, September 2019; pp. 1–6.
91. Wu, H.; Li, X.; Cheng, G.; Hu, X. Monitoring Video Resolution of Adaptive Encrypted Video Traffic Based on HTTP/2 Features. In Proceedings of the IEEE INFOCOM 2021 – IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS); IEEE: Vancouver, BC, Canada, May 10, 2021; pp. 1–6.
92. Ozkan, H.; Temelli, R.; Gurbuz, O.; Koksal, O. K.; Ipekoren, A. K.; Canbal, F.; Karahan, B. D.; Kuran, M. Ş. Multimedia Traffic Classification with Mixture of Markov Components. *Ad Hoc Networks* **2021**, *121*, 102608, doi:10.1016/j.adhoc.2021.102608.
93. Wu, Z.; Dong, Y.; Jin, J.; Wei, H.-L.; Xie, G. Multimedia Traffic Classification for Imbalanced Environment. *IEEE Trans. Netw. Sci. Eng.* **2022**, *9*, 1838–1852, doi:10.1109/TNSE.2022.3153925.
94. Bukhari, S. M. A. H.; Afaq, M.; Song, W.-C. PPS: A Packets Pattern-Based Video Identification in Encrypted Network Traffic. In Proceedings of the IEEE/ACM 16th International Conference on Utility and Cloud Computing; ACM: Taormina (Messina), Italy, December 4, 2023; pp. 1–6.
95. Barredo Arrieta, A.; Díaz-Rodríguez, N.; Del Ser, J.; Bennetot, A.; Tabik, S.; Barbado, A.; Garcia, S.; Gil-Lopez, S.; Molina, D.; Benjamins, R.; et al. Explainable Artificial Intelligence (XAI): Concepts, Taxonomies, Opportunities and Challenges toward Responsible AI. *Information Fusion* **2020**, *58*, 82–115, doi:10.1016/j.inffus.2019.12.012.
96. Khani, P.; Moeinaddini, E.; Abnavi, N. D.; Shahraki, A. Explainable Artificial Intelligence for Feature Selection in Network Traffic Classification: A Comparative Study. *Trans. Emerging Tel. Tech.* **2024**, *35*, e4970, doi:10.1002/ett.4970.
97. Dong, H.; Lee, J. S. A. The Metaverse From a Multimedia Communications Perspective. *IEEE MultiMedia* **2022**, *29*, 123–127, doi:10.1109/MMUL.2022.3217627.
98. Carrascosa, M.; Bellalta, B. Cloud-Gaming: Analysis of Google Stadia Traffic. *Computer Communications* **2022**, *188*, 99–116, doi:10.1016/j.comcom.2022.03.006.
99. Wang, Y.; Su, Z.; Zhang, N.; Xing, R.; Liu, D.; Luan, T. H.; Shen, X. A Survey on Metaverse: Fundamentals, Security, and Privacy. *IEEE Commun. Surv. Tutorials* **2023**, *25*, 319–352, doi:10.1109/COMST.2022.3202047.
100. Martin-Perez, J.; Cominardi, L.; Bernardos, C. J.; De La Oliva, A.; Azcorra, A. Modeling Mobile Edge Computing Deployments for Low Latency Multimedia Services. *IEEE Trans. on Broadcast.* **2019**, *65*, 464–474, doi:10.1109/TBC.2019.2901406.
101. Chen, M.; Saad, W.; Yin, C.; Debbah, M. Data Correlation-Aware Resource Management in Wireless Virtual Reality (VR): An Echo State Transfer Learning Approach. *IEEE Trans. Commun.* **2019**, *67*, 4267–4280, doi:10.1109/TCOMM.2019.2900624.

102. Perfecto, C.; Elbamby, M. S.; Ser, J. D.; Bennis, M. Taming the Latency in Multi-User VR 360°: A QoE-Aware Deep Learning-Aided Multicast Framework. *IEEE Trans. Commun.* **2020**, *68*, 2491–2508, doi:10.1109/TCOMM.2020.2965527.
103. Lecci, M.; Drago, M.; Zanella, A.; Zorzi, M. An Open Framework for Analyzing and Modeling XR Network Traffic. *IEEE Access* **2021**, *9*, 129782–129795, doi:10.1109/ACCESS.2021.3113162.
104. Alencar, D.; Both, C.; Antunes, R.; Oliveira, H.; Cerqueira, E.; Rosario, D. Dynamic Microservice Allocation for Virtual Reality Distribution With QoE Support. *IEEE Trans. Netw. Serv. Manage.* **2022**, *19*, 729–740, doi:10.1109/TNSM.2021.3076922.
105. Chen, T.; Grabs, E.; Tasic, I.; Cano, M.-D. Network Traffic Analysis for eXtended Reality Applications. In Proceedings of the XVI Jornadas de Ingenieria Telematica (JITEL 2023); 2023.
106. Chiariotti, F.; Drago, M.; Testolina, P.; Lecci, M.; Zanella, A.; Zorzi, M. Temporal Characterization and Prediction of VR Traffic: A Network Slicing Use Case. *IEEE Trans. on Mobile Comput.* **2023**, 1–18, doi:10.1109/TMC.2023.3282689.
107. Balasubramanian, V.; Bouachir, O.; Al-Habashnat, A. BOLT: A Bayesian Online Learning Framework for Time Sensitive Networks in Metaverse. In Proceedings of the 2023 International Conference on Intelligent Metaverse Technologies & Applications (iMETA); IEEE: Tartu, Estonia, September 18, 2023; pp. 1–7.
108. Manjunath, Y. S. K.; Szymanowski, M.; Wissborn, A.; Li, M.; Zhao, L.; Zhang, X.-P. ResLearn: Transformer-Based Residual Learning for Metaverse Network Traffic Prediction 2024.
109. Sani, Y.; Mauthe, A.; Edwards, C. Adaptive Bitrate Selection: A Survey. *IEEE Commun. Surv. Tutorials* **2017**, *19*, 2985–3014, doi:10.1109/COMST.2017.2725241.
110. Van Der Auwera, G.; David, P.; Reisslein, M. Traffic Characteristics of H.264/AVC Variable Bit Rate Video. *IEEE Commun. Mag.* **2008**, *46*, 164–174, doi:10.1109/MCOM.2008.4689260.
111. *Livestreaming Setup and Tools*; Available online: https://help.vimeo.com/hc/en-us/sections/12397317675665-Livestreaming-setup-and-tools
112. *Live Stream on YouTube*; Available online: https://support.google.com/youtube/topic/9257891
113. *Broadcasting Guidelines*; Available online: https://help.twitch.tv/s/article/broadcasting-guidelines
114. *Media Transfer Guide*; Available online: https://developers.tiktok.com/doc/content-posting-api-media-transfer-guide
115. *Requirements for Facebook Live Videos*; Available online: https://www.facebook.com/business/help/162540111070395
116. *Content Delivery Networks*; Buyya, R., Pathan, M., Vakali, A., Eds.; Lecture Notes Electrical Engineering; Springer Berlin Heidelberg: Berlin, Heidelberg, 2008; Vol. 9; ISBN 978-3-540-77886-8.
117. Liatifis, A. L.; Tegos, S. A. T.; Mitsiou, N. A. M.; Makris, I. M.; Kyranou, K. K.; Pliatsios, D. P.; Lagkas, T. L.; Sarigiannidis, P. S. NANCY SNS JU Project “Greek In-Lab Testbed Dataset 1.”
118. Beccari, M. B.; Clavenna, A. C.; Albanese, A. A. NANCY SNS-JU Project “Italian in-Lab Testbed Dataset 1” D6.6.
119. Beccari, M. B.; Clavenna, A. C.; Albanese, A. A. NANCY SNS-JU Project “Italtel Italian in-Lab Testbed Dataset 2.”
120. Pantos, R.; May, W. *HTTP Live Streaming*; RFC Editor, 2017; p. RFC8216;.

121. Reznik, Y.; Cabrera, G. Multi-Regional, Multi-CDN Delivery Optimizations Using HLS/DASH Content Steering Standard. In Proceedings of the Proceedings of the 4th Mile-High Video Conference; ACM: Denver, CO, USA, February 18, 2025; pp. 7–12.
122. *Simple Realtime Server (SRS)*; Available online: https://github.com/ossrs/srs
123. *Big Buck Bunny*; Available online: https://peach.blender.org
124. Tomar, S. Converting Video Formats with FFmpeg. *Linux J.* **2006**, *2006*, 10.
125. Müller, C.; Timmerer, C. A VLC Media Player Plugin Enabling Dynamic Adaptive Streaming over HTTP. In Proceedings of the 19th ACM International Conference on Multimedia; ACM: Scottsdale, Arizona, USA, November 28, 2011; pp. 723–726.
126. O'Hanlon, P.; Aslam, A. Latency Target-Based Analysis of the DASH.Js Player. In Proceedings of the 14th ACM Multimedia Systems Conference; ACM: Vancouver, BC, Canada, June 7, 2023; pp. 153–160.
127. Mahmoud, H.; Abozariba, R. A Systematic Review on WebRTC for Potential Applications and Challenges beyond Audio Video Streaming. *Multimed. Tools Appl.* **2024**, *84*, 2909–2946, doi:10.1007/s11042-024-20448-9.
128. Chakraborty, T.; Chattopadhyay, S.; Das, S.; Kumar, U.; Senthilnath, J. Searching for Heavy-Tailed Probability Distributions for Modeling Real-World Complex Networks. *IEEE Access* **2022**, *10*, 115092–115107, doi:10.1109/ACCESS.2022.3218631.
129. Grab, E.; Petersons, E. Hurst Parameter Estimation by Wavelet Transformation and a Filter Bank for Self-Similar Traffic. *Aut. Control Comp. Sci.* **2015**, *49*, 286–292, doi:10.3103/S0146411615050041.
130. Franzl, G. Queueing Models for Multi-Service Networks. PhD Thesis, Technische Universität Wien, 2015.
131. Lin, K.; Xu, X.; Xiao, F. MFFusion: A Multi-Level Features Fusion Model for Malicious Traffic Detection Based on Deep Learning. *Computer Networks* **2022**, *202*, 108658, doi:10.1016/j.comnet.2021.108658.
132. Song, M.; Ran, J.; Li, S. Encrypted Traffic Classification Based on Text Convolution Neural Networks. In Proceedings of the 2019 IEEE 7th International Conference on Computer Science and Network Technology (ICCSNT); IEEE: Dalian, China, October 2019; pp. 432–436.
133. Li, F.; Ye, F. Simplifying Data Traffic Classification with Byte Importance Distillation. In Proceedings of the ICC 2021 – IEEE International Conference on Communications; IEEE: Montreal, QC, Canada, June 2021; pp. 1–6.
134. Roy, S.; Shapira, T.; Shavitt, Y. Fast and Lean Encrypted Internet Traffic Classification. *Computer Communications* **2022**, *186*, 166–173, doi:10.1016/j.comcom.2022.02.003.
135. Yang, Y.; Yan, Y.; Gao, Z.; Rui, L.; Lyu, R.; Gao, B.; Yu, P. A Network Traffic Classification Method Based on Dual-Mode Feature Extraction and Hybrid Neural Networks. *IEEE Trans. Netw. Serv. Manage.* **2023**, *20*, 4073–4084, doi:10.1109/TNSM.2023.3262246.
136. Batista, G. E. A. P. A.; Prati, R. C.; Monard, M. C. A Study of the Behavior of Several Methods for Balancing Machine Learning Training Data. *SIGKDD Explor. Newsl.* **2004**, *6*, 20–29, doi:10.1145/1007730.1007735.
137. Lemaître, G.; Nogueira, F.; Aridas, C. K. Imbalanced-Learn: A Python Toolbox to Tackle the Curse of Imbalanced Datasets in Machine Learning. *Journal of Machine Learning Research* **2017**, *18*, 1–5.
138. Hofstede, R.; Celeda, P.; Trammell, B.; Drago, I.; Sadre, R.; Sperotto, A.; Pras, A. Flow Monitoring Explained: From Packet Capture to Data Analysis With NetFlow and IPFIX. *IEEE Commun. Surv. Tutorials* **2014**, *16*, 2037–2064, doi:10.1109/COMST.2014.2321898.

139. Brownlee, N.; Mills, C.; Ruth, G. *Traffic Flow Measurement: Architecture*; RFC Editor, 1999; p. RFC2722.
140. Yu, L.; Tao, J.; Xu, Y.; Sun, W.; Wang, Z. TLS Fingerprint for Encrypted Malicious Traffic Detection with Attributed Graph Kernel. *Computer Networks* **2024**, *247*, 110475, doi:10.1016/j.comnet.2024.110475.
141. Aouini, Z.; Pekar, A. NFStream: A Flexible Network Data Analysis Framework. *Computer Networks* **2022**, *204*, 108719, doi:10.1016/j.comnet.2021.108719.
142. Chen, T.; Grabs, E.; Ipatovs, A.; Cano, M.-D. A Novel Proprietary Internet Video Traffic Dataset Generation Algorithm. *Applied Sciences* **2025**, *15*, 515, doi:10.3390/app15020515.
143. Wang, W.; Zhu, M.; Wang, J.; Zeng, X.; Yang, Z. End-to-End Encrypted Traffic Classification with One-Dimensional Convolution Neural Networks. In Proceedings of the 2017 IEEE International Conference on Intelligence and Security Informatics (ISI); IEEE: Beijing, China, July 2017; pp. 43–48.
144. Buitinck, L.; Louppe, G.; Blondel, M.; Pedregosa, F.; Mueller, A.; Grisel, O.; Niculae, V.; Prettenhofer, P.; Gramfort, A.; Grobler, J.; et al. API Design for Machine Learning Software: Experiences from the Scikit-Learn Project. **2013**, doi:10.48550/ARXIV.1309.0238.
145. Van Der Walt, S.; Schönberger, J. L.; Nunez-Iglesias, J.; Boulogne, F.; Warner, J. D.; Yager, N.; Gouillart, E.; Yu, T.; the scikit-image contributors Scikit-Image: Image Processing in Python 2014.
146. Bradski, G. The OpenCV Library. *Dr. Dobb's Journal of Software Tools* **2000**.
147. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. **2016**, doi:10.48550/ARXIV.1603.02754.
148. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library 2019.
149. Fey, M.; Lenssen, J. E. Fast Graph Representation Learning with PyTorch Geometric 2019.
150. Lotfollahi, M.; Jafari Siavoshani, M.; Shirali Hossein Zade, R.; Saberian, M. Deep Packet: A Novel Approach for Encrypted Traffic Classification Using Deep Learning. *Soft Comput.* **2020**, *24*, 1999–2012, doi:10.1007/s00500-019-04030-2.
151. Shapira, T.; Shavitt, Y. FlowPic: A Generic Representation for Encrypted Traffic Classification and Applications Identification. *IEEE Trans. Netw. Serv. Manage.* **2021**, *18*, 1218–1232, doi:10.1109/TNSM.2021.3071441.
152. Liu, C.; He, L.; Xiong, G.; Cao, Z.; Li, Z. FS-Net: A Flow Sequence Network For Encrypted Traffic Classification. In Proceedings of the IEEE INFOCOM 2019 – IEEE Conference on Computer Communications; IEEE: Paris, France, April 2019; pp. 1171–1179.
153. Martinsson, P.-G.; Rokhlin, V.; Tygert, M. A Randomized Algorithm for the Decomposition of Matrices. *Applied and Computational Harmonic Analysis* **2011**, *30*, 47–68, doi:10.1016/j.acha.2010.02.003.
154. Kingma, D. P.; Ba, J. Adam: A Method for Stochastic Optimization 2014.
155. Mao, A.; Mohri, M.; Zhong, Y. Cross-Entropy Loss Functions: Theoretical Analysis and Applications 2023.
156. Dorogush, A. V.; Ershov, V.; Gulin, A. CatBoost: Gradient Boosting with Categorical Features Support 2018.
157. James, G.; Witten, D.; Hastie, T.; Tibshirani, R. *An Introduction to Statistical Learning*; Springer Texts in Statistics; Springer New York: New York, NY, 2013; Vol. 103; ISBN 978-1-4614-7137-0.

**Tianhua Chen** was born in Nanjing, China, in 1995. He received his Master's degree in Engineering in Telecommunications (2021) from Riga Technical University. He has worked as a network engineer at multiple Internet Service Providers. Since 2023, he has been a researcher at Riga Technical University. His research interests are in multimedia, XR application network traffic analysis, as well as 5G Open RAN and SDN application development.