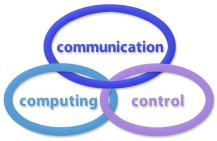
INTERNATIONAL JOURNAL OF COMPUTERS COMMUNICATIONS & CONTROL
Online ISSN 1841-9844, ISSN-L 1841-9836, Volume: 19, Issue: 5, Month: October, Year: 2024
Article Number: 6636, https://doi.org/10.15837/ijccc.2024.5.6636



CCC Publications

Automatic Detection of Stalling Events using Machine Learning Algorithms

A. F. Celis Velez, L. M. Castañeda Herrera, J. L. Arciniegas Herrera, H. F. Bermúdez Orozco

Andres Fernando Celis Velez, Luis Miguel Castañeda Herrera*, José Luis Arciniegas Herrera
Department of Telematics
University of Cauca, Colombia
Cra. 2 no 4N-140, Popayan, Cauca, Colombia
*Corresponding author: luiscastaneda@unicauca.edu.co
cvandres@unicauca.edu.co
jlarci@unicauca.edu.co

Héctor Fabio Bermúdez Orozco
Department of Electronics
University of Quindio, Colombia
Cra 15 #15-12, Armenia, Quindío, Colombia
hfbermudez@uniquindio.edu.co

Abstract

With the proliferation of new video streaming OTT service applications, content providers must guarantee an efficient, effective and satisfactory interaction between the user and the service application, to measure this they use the quality of experience (QoE). QoE in the context of telecommunications can be understood as the acceptability of a service perceived by end users. However, given the variability of the network and the different factors that can intervene during the service, this perception can be altered and one of these causes is stalling events. It is necessary to have an accurate method to detect stalling events, allowing content providers to make better decisions in the design of their OTT applications and thus reduce customer desertion and the economic losses of service or content providers. For this reason, in this paper firstly we show a comparative analysis of supervised algorithms showing its invalidity given the imbalance of the data despite the high levels of accuracy. Therfore, we propose as a contribution a novel method based on algorithms provided by SSAD (Semi Supervised Anomaly Detection) for the detection of stalling events. Finally, we observe that in our method the Isolation Forest model has the best performance, closer to 1 and we will show more detail about its performance in future works.

Keywords: DASH, Machine Learning, OTT, QoE, Stalling Events

1 Introduction

The growing demand for multimedia services has generated a considerable increase in traffic on networks, 70% by 2023 according to the Ericsson ConsumerLab report [6, 8, 15]. Therefore, Internet

service providers (ISPs) and content providers are in a vanguard position where they must provide quality service, specifically for videostreaming traffic, which according to reports provided by Ericsson occupies about 63% of all internet traffic [15], which has forced the implementation of standards that allow monitoring the quality of experience (QoE). QoE is defined according to the International Telecommunication Union, in recommendation P10/100 as the "degree of user satisfaction or dissatisfaction with an application or service" [14]. For QoE estimation, the recommendations provided by ITU P.1203 are mainly used, where the key performance indicators KPI are described. Also, this recommendation makes a parametric quality assessment based on progressive and adaptive download bitstream for video services that employ TCP (Transmission Control Protocol).

Providing the way to evaluate the complete audiovisual quality, the individual audio and video quality, which together with the initial loading delay and the stalling events make up the objective QoE [3], these are treated normally as one of the factors that negatively influence user satisfaction [27], generating discomfort in the user with the service provided. Therefore, the detection of these events plays a determining role for ISPs and video service distributors (LVS or OD) or OTT, due to the degradation they produce in the quality of the experience, also to being key performance indicators (KPI) in obtaining the objective quality of the experience (ITU-P1203) [11]. The stalling events are playback interruptions (stopping or rebuffering events) that occur at the presentation layer level and can be considered anomalous events that occur very rarely in normal video playback, but are produce a significant degradation in the (QoE) [3].

Nevertheless, for this paper the outliers are the essential input for the identification of stalling events, since they are values located outside the distribution ranges and allow us to understand and identify its occurrences based on analysis trafficc metrics. Currently, research already speaks of the presence of stalling events [27],[1],[23],[9]. However, in general the contributions are based exclusively on the detection of these events from the buffer dynamics or in other cases from their manual identification, making it a slow, time-consuming and complex process.

Therefore, this paper presents an analysis for the automatic detection of stalling events using Machine Learning (ML) algorithms. Firstly, a dataset was generated in a client-server architecture, where *8* video metrics and *9* audio metrics are captured using the *dashif-player* tool. Using the dataset we proceed to form clusters with unsupervised learning techniques, which relationships can be established in the metrics and determine the presence of stalling events. Once the different clusters have been identified and the dataset labeled, we use supervised learning algorithms to classify the stalling events. Subsequently, considering that stalling events are anomalous events and their occurrence is low, it is essential to use specialized outliers analysis techniques in their identification, so that they allow establishing the presence of this events using different algorithms; Finally, we present method and scenario for the detection of stalling events and the performance of different algorithms and methods to detect this outliers.

This paper is organized as follows, section II will provide a description of the background related to the fundamental aspects. Section III discusses the materials and methods, which include the construction of the dataset and proposed escenario. In section IV the results and analisys are discussed and finally, the conclusions are presented in section V.

2 Background

First, this section describes in general terms how stalling events are generated and their relationship to the quality of experience (QoE), then we explain in more detail what they are and the possible ways in which they can be detected. Finally, we present how these events are detected and the tools used in current research reviewed in the literature.

Services that generate video traffic need significantly high network performance to achieve favorable conditions that allow normal playback of videos, for which network operators normally adjust their network parameters in such a way that allows ensuring quality of service is normally attached to parameters such as bandwidth, content buffering and packet loss [27]. However, according to [1], for network providers or content providers, more than the specific parameters of the network performance or the content itself, what they seek is to measure the degree of customer satisfaction with the service

provided, avoiding user desertion and guarantee the recurrence of the services provided.

To achieve this purpose, the term "quality of experience" was adopted, which depending on the network provisions, physical, algorithmic and processing resources, the most appropriate method is used for its estimation, whose result is closely related to stalling events and the initial loading delay. Which, according to [12] are the main causes of degradation in user satisfaction with the services provided (KPI), greatly compromising forms the display of the video content. According to [4], in most cases, when a poor quality of experience occurs, customers prefer to cancel the service and not report to their network operator that the service has errors, generating million-dollar losses for both content and network providers. Therefore it is important to identify these events.

Stalling events are defined in [23],[9],[27] as events that generally occur due to rebuffering, which occur when the buffer reaches its maximum level and a retransmission is necessary until the buffer data level exceeds the threshold assigned by the video player to start playback. Furthermore, the author in [2] mentions that of all the different factors that can affect QoE for adaptive video transmission over HTTP, stalling events are the factor with the greatest influence on QoE, which is why it is essential its detection and quantification to get closer to a more precise QoE. Therefore, it is indisputable that stalling events present a strong relationship with the QoE for a service, generating an influence where it can be stated that, if the number of stalling events is greater then the quality of the experience decreases [11], [27], [29]. It is at this point where it is necessary to develop a method capable of detecting stalling events.

For this reason, this paper focuses to build a method based on the detection of stalling events, for the construction of this method. Firstly, a review is initially proposed based on 3 thematic axes that are strongly related to the presence of stalling events, like its definition, the way in which different authors detect and quantify stalling events and the different tools developed to quantify stalling events.

2.1 stalling events

The stalling events are an anomalous behavior that can be seen in the estimation process, this is one of the fields that has received the most attention in recent years as mentioned in [16], these can be defined in different ways depending on the context, for example in some cases it is called exceptions, defects or atypical behavior. Within the detection of anomalies there are several challenges, but one of the most significant is the noise in the data, which is usually confused with anomalies. To avoid falling into biases caused by noise, it is vitally important to carry out a complete exploratory analysis of the data and verify that the measured values for each of the characteristics are within the normal distribution ranges, as mentioned in [19].

Likewise, within anomaly detection there are several types of data labeling for the construction of models, which include supervised, semi-supervised and unsupervised learning; the supervised one includes labels for both the normal classes and the anomalous data; the semi-supervised one only labels the normal classes and the unsupervised one behaves like a cluster in search of finding patterns in the data [19]. Of all of them, given the idiosyncrasy in the behavior of stalling events that occur with very low frequency but have a disturbing effect on the user, the use of semi-supervised learning is better in this case, since it only labels normal classes. Therefore, its response is much faster as long as the data labeling is correctly characterized. In conclusion, it was possible to establish that although there are works aimed at detecting stalling events using different machine learning techniques, there was no evidence of the use of algorithms based on outlier detection for their identification, which is why it is essential to carry out an exploration of the different AD (Anomaly Detection) algorithms, because they are capable of identifying atypical behavior and the nature of stalling events is the same. Therefore, including AD algorithms for their identification should theoretically increase the performance of the proposed method.

2.2 Detect and Quantify stalling events

In the literature review we found works that provide clues to understand the behavior of stalling events, this It is visible in works such as [16, 20, 29, 32] on which the bitstream, user buffer size, bitrate, rebuffering time, total video time, the encoding format and the total rebuffering time among

other metrics to identify when a stalling event occurs. Likewise, in articles such as [9],[27],[4], they propose the need to develop tools for the detection of *stalling* events considering metrics obtained from parameters such as video playback behavior, stochastic properties of buffer allocation, network bit rate dynamics, and the requested video quality for buffer-based video playback, furthermore, the authors in [2] determine the number of stalling events produced during a session by manually recording them.

In [10] the autors do not directly identify *stalling* events, however they propose a real-time predictive model that is capable of quantifying the QoE for adaptive video services, making it possible to establish to what extent the variation in the bit rate produces stalling events and how the QoE changes as these events occur, allowing the bitstream to be deliberately modified to generate stalling events that change the QoE and also detect their presence during playback recommendation a video. The work in [28] can be considered as a main reference for the present proposal since the authors determine the length of stalling events, their quantity, the time prior to the occurrence of an event, the frequency and the rebuffering rate from the client player, validating the quality of the collected data through subjective tests.

Another reference work is presented in [18], there the authors propose to detect and predict stalling events through traffic classification by extracting a set of characteristics from TCP traffic that are analyzed using PFA (Principal Features Analysis). From the results obtained from the analysis of the TCP flow and the correlation obtained, they determined that the TCP SYN flag is the element with the greatest relevance in the detection of stalling events. However, to increase the performance of the models and get closer to a more precise detection of these events, it is also necessary to consider the TCP sequence number and the size of the TCP window [17, 24]. Since, they are two factors that greatly influence the characteristic distribution diagram presented in [18] but that are not taken into consideration in the input layer of the models.

In summary, the presence of stalling events, which are mentioned in the literature, are exclusively related to the buffer as the only element that generates the presence of these events[10]. according to [14] the latency must be linked to the buffer and the lack of frames in the buffer, so it is implicit that new requests must be generated to the server seeking to store frames in the buffer to present them to the user, the latency causes a delay in the arrival of the frames or video segments, this produces stops in the video playback on the client (stalling events), this could be corroborated by working on the detection of stalling events, using the dashif platform, which uses DASH (Dynamic Adaptive Streaming HTTP), a standard developed by MPEG published as an international standard [26].

DASH was designed to avoid the incompatibilities presented among other proprietary adaptive streaming techniques, for which it defines concepts such as: periods, segments, performances and sets. DASH uses a manifest file which is of type mpd; The MPD (Media Presentation Description) file is stored on the HTTP server and describes the characteristics of each of the content representations and their segments. In a live streaming, the MPD file will be updated every certain interval, the algorithm adapts to the network conditions, especially those of video quality that are strictly linked to the bitrate rate, a high value in this rate , causes the size of the buffer to grow very quickly and any unforeseen change in the quality of service on the network strongly affects the playback of the video [9].

As a conclusion to this thematic axis, it can be seen how stalling events are atypical values generated by network fluctuations and unexpected behaviors in the client player, hence why in no standard was it specified how they should be captured, in addition to the recommendations given by the ITU, it is made clear that it is not their object to detect stalling events, which is why it is necessary and imperative to establish a mechanism and/or method that allows the detection of stalling events. On the other hand, in some documents found in the literature such as [27], [18], [31] they mention that in addition to the metrics used in traffic classification, other characteristics such as bytes per flow and the duration of the flow, in order to increase the performance of the classifiers; It is very important to keep in mind that when adding features you must be careful not to fall into overfitting. A good way to do this is to apply feature engineering techniques and avoid including features that bias the models.

2.3 Tools for Quantify stalling events

The following topic addresses the different tools developed to quantify stalling events. In the literature, some authors carry out the analysis of stalling events directly in estimating the quality of the experience according to the developed method as in [31] and others present novel tools to quantify them at the MOS level (Median Opinion Score) as in [5], due to the MOS is a measure used to evaluate the quality of media signals and corresponds to a scalar value [13]. In [5] they propose a prototype called YOUQMON, which allows the detection of stalling events for clients watching YouTube videos by combining passive traffic analysis techniques to detect stalling events in YouTube video streams using a QoE, which allows stalling events to be mapped into an average opinion (MOS), score that reflects the end user experience.

[7] presents a stream-based machine learning approach (ViCry), that analyzes stalling events for YouTube streaming sessions in real time from encrypted network traffic. Likewise, they ensure that the constant memory characteristics are extracted from the flow-based encrypted network traffic and are fed into a random forest model, which predicts whether the current time interval contains stalling events or not. At [21] they developed a framework for analyzing ABR streaming performance, called You Slow, a web browser plugin that can detect and report live buffer blocking. This software allows you to analyze the behavior of the buffer in real time and in theory detect certain stalling events, however, its functions are limited and in some cases they are not precise.

As a conclusion to this topic, we can talk about certain tools that are capable of quantifying stalling events. However, these tools are limited to the YouTube platform, which uses encrypted traffic, they are also independent of the recommendation and use their own type of QoE estimation. Furthermore, according to the results obtained in the systematic review, the relevance and timeliness of the research topic can be appreciated. It is observed that all research efforts on this topic are focused on improving the QoE estimation of the client in the playback of videos and thus reduce the user defection rate for the services provided by network and content providers.

3 Materials and Methods

This section describes the general methodology and scenario for evaluating ML models that seek to detect stalling events in a VoD service. First, an exploratory analysis is performed to identify stalling events and label the data to adapt the models. Once the dataset is labeled, tests are carried out with algorithms that are part of semi-supervised learning, which include individual and ensemble algorithms. The first of them can be of four categories that include linear models, proximity-based, probabilistic and neural networks, all of them using only one type of algorithm to identify anomalies. On the other hand, ensemble methods combine individual classification algorithms to increase performance in anomaly detection.

3.1 Proposed scenario

At this moment the method for detecting stalling events is in an initial state, in which the first scenario presented in Figure 1 is a 3-tier client-server architecture connected to a WAN network of a tier ISP. 1, which uses a videostreaming CD that guarantees low latencies in video transmission, and is also complemented by the 3-level architecture that guarantees practical, reliable handling and management over the network. Additionally, it allows centralized control of it, which allows obtaining more precise metrics of the requests that can be made on the data logic and being able to observe network behaviors from the business logic, without having to modify any parameters in the client presentation logic, which presents a video player from the *DASH-IF* platform and through *webscrapping 8* video metrics were captured with 1 second sampling that allow determining the occurrence of stalling events, which are: buffer size, rate bitrate, latency, lost frames, download rate, playback rate, download and ratio. With this data, the initial dataset is created , which consists of: 964 instances labeled in 3 classes, *0* No stalling events, *1* medium possible probable event and *3* highly probable stalling event.

From audio characteristics were captured: buffer length, bitrate, download rate, lost frames and latency, in addition, packet parameters were obtained such as: TCP sequence number, TCP window

size, packet with duplicate ACK, previous packet lost, IP packet length, IP header size, TCP header size and the TCP-ACK, TCP-URG, TCP-PUSH, TCP-RESET, TCP-SYN and TCP-FIN flags. Once the data is stored in the database, it is explored using EDA (Exploration Data Anlysis) in order to know what its behavior, observe patterns and recognize statistical distributions that can be useful in the future. Subsequently, the data are stacked and cleaned, it is appropriate to analyze and filter them according to the convenience of the field of action. Additionally, if required, main characteristics analysis (PFA) is used to establish which of them are suitable to be included within the dataset that will form part of training and testing the models. The models are tested, characterized and validated based on cross-validation tests, with the use of statistical significance tests being necessary in the event that some models are close in the validation results. Once the model with the best behavior has been identified with respect to the captured data, it is deployed in a controlled environment with metrics captured in 15-second windows.

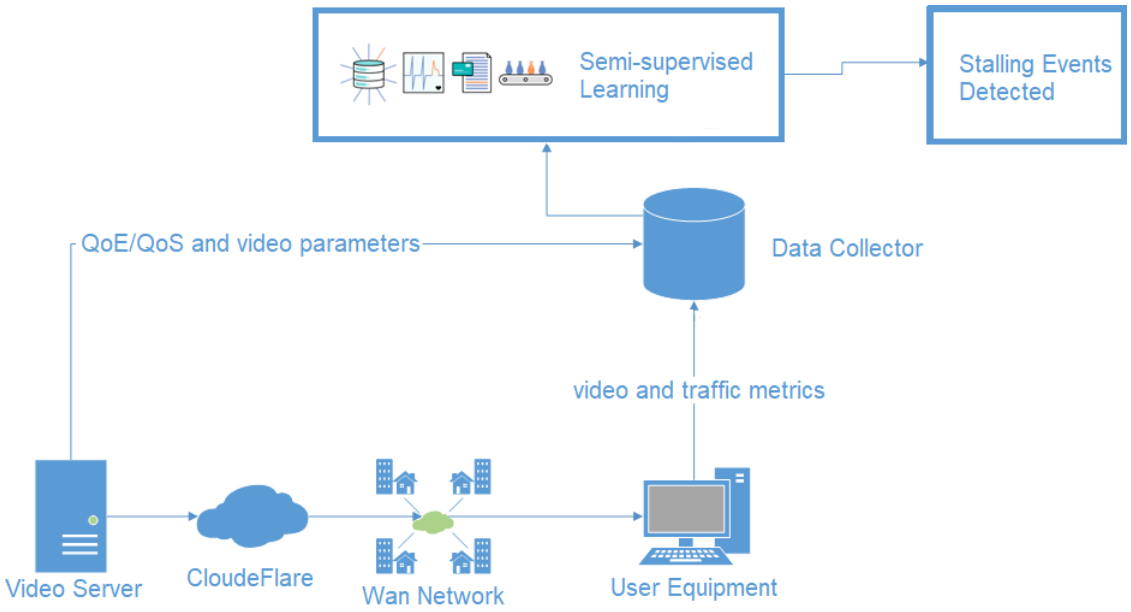


Figure 1: Stalling event detection structure

3.2 Proposed method

The proposed method for the detection of *stalling* events is presented in Figure 2, the first stage is the process of loading data corresponding to the metrics that allow identifying a stalling event using web *scrapping* from dashif. The second stage consists of characterizing the data in order to define or establish a behavioral pattern. In this case, unsupervised learning algorithms were used to generate clusters under the probability of occurrence of a stalling event (high, medium and low), this makes data labeling easier. Once the dataset is labeled based on the characterization carried out using the clusters, the stalling events are classified using supervised learning algorithms using outlier detection algorithms.

Then, the stage 3 corresponds firstly to the training of the supervised learning algorithm, at this point it is necessary to execute discriminatory tests in order to determine with certainty the algorithm that should be used in the method. Stage 4 corresponds to testing the supervised learning algorithm (classification) developed for this purpose, for which the relevance of said trained algorithm must be verified using the metrics for evaluating classification performance. Furthermore, the Stage 5 corresponds to the validation of the algorithm, using a different dataset than the one used before to train the algorithm, subsequently it is analyzed and interpreted with the trained algorithm and the output data if it correctly classifies the presence of stalling events. Finally, Stage 6 is responsible for refining the algorithms, this is mainly oriented to the configuration of hyperparameters that the algorithms based on neural networks or the outlier identification algorithms have in the outlier factor.

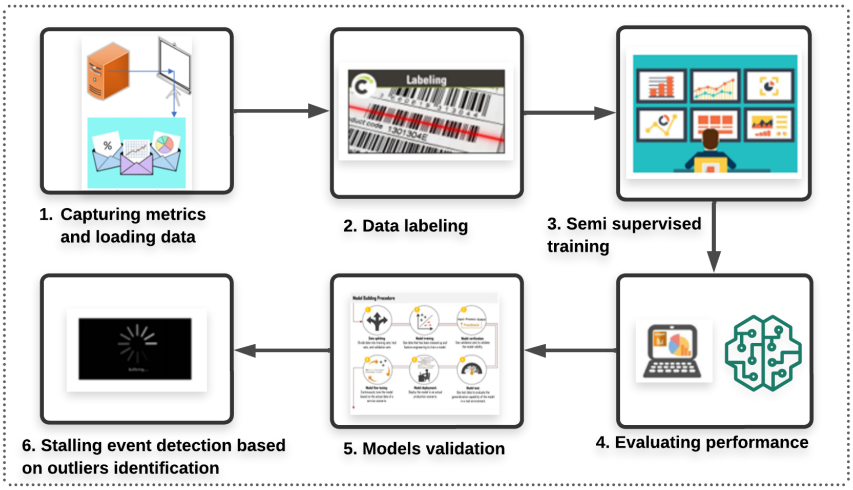


Figure 2: Stalling event detection phases

Once the process was completed we found that the supervised models did not meet the requirements for the identification of stalling events since it suffered from biases in the data and anomalous behaviors in the data flow generate a giant negative effect on them. Therefore, these models did not fit the behavior of the data in real cases. In addition, we observed that it is more efficient identify if there are events with two classes and then, we decided to relabel the dataset, the first class with the occurrence of events and the second class without stalling events. Then, knowing that stalin events can be treated as an outlier, we decided to use semi-supervised algorithms based on the Khan and Madden SSAD (Semi-Supervised Domain Adaptation) taxonomy that divides the detection of outliers into 6 branches [32]. Two of them on assembled algorithms (Outliers ensembles and Combinatios based algorithms) and the other four on individual algorithms (Neural-networks, Probabilistics, Proximity-based and Linear models). We chose an algorithm for each branch. Among the algorithms considered are: ocSVM (oneClass Support Vector Machine), LOF (Local outlier Factor), kNN (k-Nearest Neighbors), BRM (Baggin Random Minner), Isolation Forest and VAE (Variational AutoEncoder) respectively for each branch [30].

4 Results and discussion

In this section we first show a statistical analysis of the data obtained, then we use supervised learning techniques were used to obtain a first impression of the behavior of the models with the data and we used different data balancing techniques to determine if we can improve the performance of the algorithms, in this case we made a comparison of algorithms that are commonly used such as Naive Bayes, Random Forest, decision tree among others.

Then, watching the behaviors of algorithms we determined that the supervised models did not meet the requirements for the identification of stalling events since it suffered from biases in the data and anomalous behaviors in the data flow generate a giant negative effect on them. Therefore We decided to relabel the datase and propose the use of semi-supervised algorithms for the detection of Stallion events, which also treat them as if they were outliers. Finally, we present preliminary results in the use of these algorithms.

For the statistical analysis of the data, we obtained the median, mode, standard deviation, variance, mean absolute deviation, IQR and quartiles as shown in Table 1, among them the ones that provide the most information are the variance since it is a measure of dispersion that represents how variable the data are respect to its mean. The standard deviation that measures how far apart the data are, the IQR, which is the interquartile range, is defined as the difference between the third quartile and the first quartile, to find skewed distributions, which makes it a very suitable tool to observe the data dispersion.

Table 1: Statistical analysis of the data

	Median	Mean	Mode	STD	Variance	MAD	IQR	Quartiles		
								1	2	3
Buffer size	29,97	30,08	37,05	17,3	299,49	14,94	29,87	14,82	30,08	44,69
Bitrate	8206	8276	14932	4703,17	21002	4108,32	8439,8	4094	8276	12534
Index Download	1,022	0,9921	0,8373	0,5759	0,3317	0,497	0,986	0,513	0,992	1,499
Current Index	8	9	8	0	0	0,988	9	8	9	10
lost frames	6	7	4	2	4	1,865	7	5	7	9
Latency	0,7491	0,7	1	0,1719	0,029	0,151	0,3	0,6	0,7	0,9
Download	2,9998	2,993	1,796	1,153	1,3307	1	2,01	1,99	2,99	4

Then, considering that stalling events are not predicted but detected, supervised learning classification algorithms are used to establish said behavior. Where supervised algorithms need data labeling so that the algorithms learn from their behavior and can predict the presence of these events in the future. In the case of this paper, *3* possible probabilities of occurrence of stalling events are considered, which are low with "0", medium with "1" and high with "2", once the data was cataloged it could be observed the unbalanced behavior of the classes in figure 3

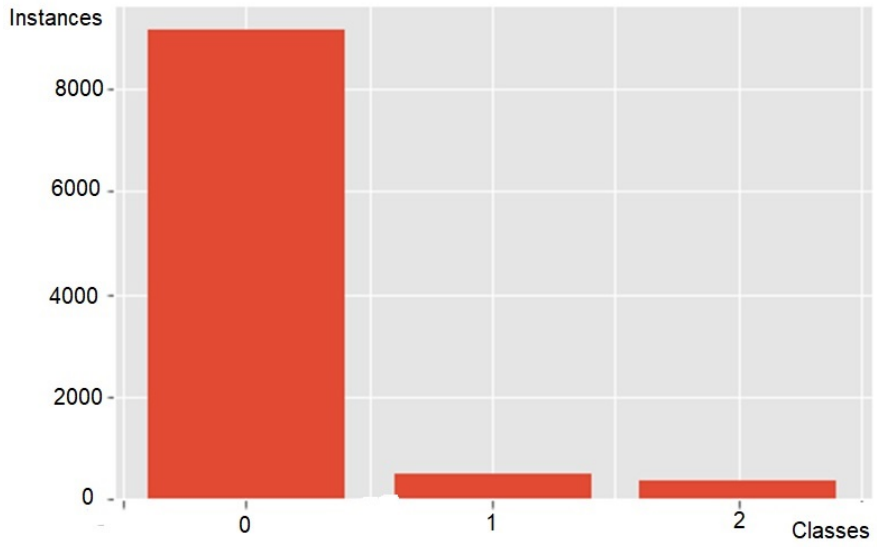


Figure 3: Amount of data per class.

Figure 3 shows an imbalance in the classes, which is one of the biggest problems in classification because it harms the algorithms in their information generalization process, affecting the minority classes, generating a false sensation that the model works well. We use supervised learning techniques such as random forest, kNN and logistic regression were used to obtain a first impression of the behavior of the algorithms with the data. Initially, random forest was tested, which is a joint learning method composed of multiple decision trees, for which *100* trees were used for data analysis. From this analysis, the behavior of the confusion matrix in Table *2* was obtained, where it can be seen that the classifier is biased towards class *2* due to the number of samples there and its precision is almost 100% in all its classes.

Table 2: Performance Metrics Random Forest

	Class 0	Class 1	Class 2
Precision	1	1	1
Recall	1	1	1
F1-score	1	1	1

In the case of k-NN, which is an algorithm in charge of classifying each new data into the group

that corresponds to it, depending on whether it has k neighbors closer to one group or another, it calculates the distances of a new element to each of the existing ones and ordering said distances from smallest to largest to select the group to which it belongs. Figure 4(a) (left side) represents the average value of the algorithm's behavior as the value of k (number of neighboring points taken into account to classify the groups already defined) increases with 20% of test data and 80% training data, Figure 4(b) represents the value of k for 30% test data and 70% training data.

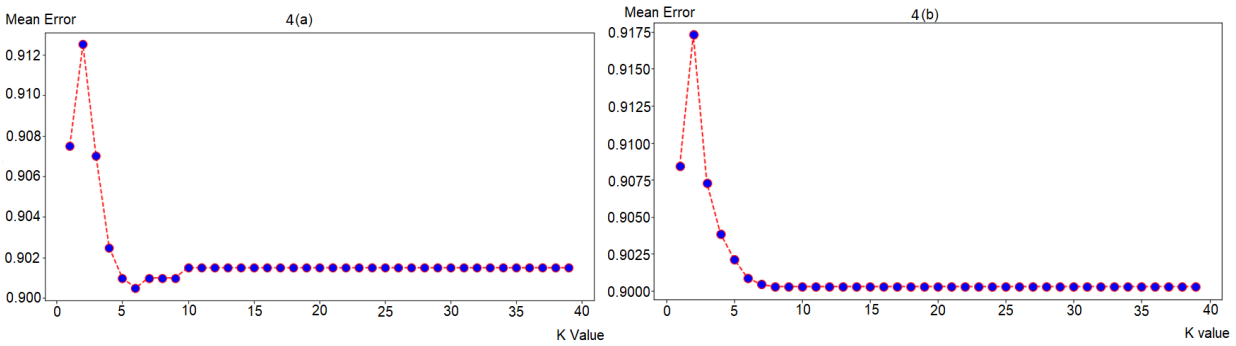


Figure 4: Figure 4(a) Error rate k, 20% test and 80% training Figure 4(b): Error rate k, 30% test and 70% training.

With the k-NN algorithm, again a bias is observed due to the imbalance of classes in table 3, the results of both precision and recall lead us to believe that because of the number of samples for each of the classes and that in class 3 everything is classified correctly.

Table 3: Performance Metrics k-NN

	Class 0	Class 1	Class 2
Precision	0.93	0.9	1
Recall	0.94	0.91	1
F1-score	0.93	0.90	1

It is necessary to perform class balancing to observe correct behavior of the algorithms, otherwise the model is biased to learn and predict behavior in the data. In this case, *5* different balancing techniques were used to observe the behavior of the classes. First, is the adjustment of model parameters that adjusts certain parameters or metrics of the algorithm itself to try to balance the minority classes by penalizing the majority class during training, usually the weight/balanced parameter is used to balance the classes, this is used with logistic regression for classification and the report in Table 4 was obtained, where class *1* still presents a low percentage of precision.

Table 4: Ranking report using penalty in algorithm

	Class 0	Class 1	Class 2
Precision	0.96	0.79	1
Recall	0.99	0.97	0.99
F1-score	0.97	0.87	0.99

The second method for data balancing is to modify the dataset, for this certain samples of the majority class are eliminated to reduce it and try to balance the situation, however certain samples can be eliminated that may be essential and contain information, to eliminate them, some criteria must be followed. In this case, the k-NN algorithm is used to reduce the majority class that allows them to be eliminated to balance these classes. Table 5 shows the classification report where it continues to present problems with class *1*.

Table 5: Performance Metrics k-NN with elimination

	Class 0	Class 1	Class 2
Precision	1	0.76	1
Recall	0.99	1	0.99
F1-score	1	0.87	0.99

The third method of class balance is artificial samples, creating certain samples using algorithms that try to follow the trend of the majority class. The dangerous thing about this is that the natural distribution of that class can be altered and generate confusion in the classification model. This is known as majority class oversampling, Table 6 shows the classification report, where the precision in the algorithm is slightly more balanced between the different classes.

Table 6: Performance Metrics k-NN with artificial samples

	Class 0	Class 1	Class 2
Precision	0.96	0.83	1
Recall	0.95	0.97	0.99
F1-score	0.96	0.89	1

The fourth method used is resampling with Smote-Tomek, which consists of simultaneously applying a subsampling algorithm and an oversampling algorithm at the same time on the dataset. In this case, the SMOTE tool was used for oversampling, where points of close neighbors are searched and points are added in a straight line between them, and Tomek is used for undersampling, which removes those of different classes that are close neighbors and allows us to see the border of the classes, Table 7 shows the classification report where the same behavior as in the previous case is observed.

Table 7: Performance Metrics k-NN with resampling

	Class 0	Class 1	Class 2
Precision	0.96	0.83	1
Recall	0.95	0.97	0.99
F1-score	0.96	0.89	1

The last method is the ensemble of models with balancing, which consists of training various models and together obtaining the final result, but making sure to take balanced training samples, in this case an ensemble classifier is used that uses bagging and the model It will be a DecisionTree, where its performance is observed in table 8, where excellent behavior is observed in each of the classes. These data balancing techniques are useful. However, the guideline in Figure 5 is the correct way to do it since it does so using a proven guideline that takes into consideration the elimination of certain outliers that can bias oversampling.

The approach proposed in [22] focuses on the entire data set (regardless of class) . They identify outliers from the original data set and then upsample the instances with replacement. The goal of this step is to increase the number of these rare cases within the data set so that when SMOTE is applied to obtain class balance, more synthetic instances are generated near the neighborhood of the outliers. The original data were searched for outliers using the IQR algorithm, which identified *49* outliers. These cases were oversampled with replacement by 500%, resulting in *245* outliers and subsequently added back to the original data, i.e., $(768 - 49) + 245 = 964$ instances.

$$Intervals = Q1 - 1.5IQR \; y \; Q3 + 1.5IQR \tag{1}$$

The IQR interval is given by ecuation 1, where Q1 is the first quartile, Q3 the third quartile and IQR the interquartile ratio. After the process of adding the data to the original dataset, SMOTE is

Table 8: Performance Metrics Desicion tree

	Class 0	Class 1	Class 2
Precision	0.99	0.97	1
Recall	1	0.99	1
F1-score	0.98	0.99	1

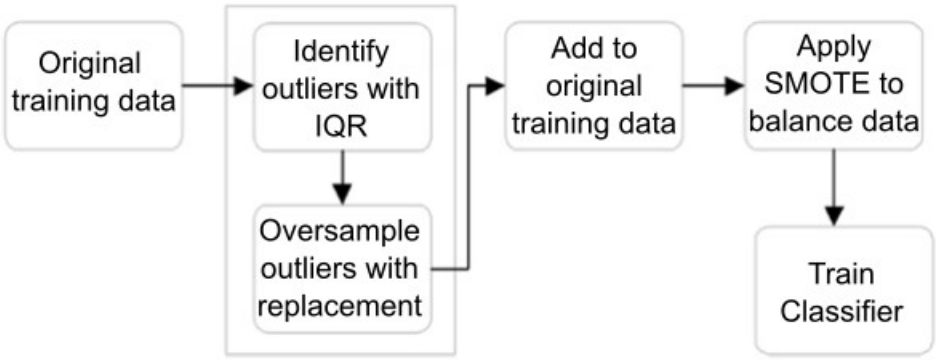


Figure 5: Data balancing procedure before applying oversampling for correct class balancing [22].

applied to search for close neighbor points and points are added in a straight line between them to finally train the different algorithms. The tested algorithms obtained their performance score with the K-Fold Cross-Validation test, which is an iterative process that consists of randomly dividing the data into k groups of approximately the same size, k-1 groups are used to train the model and one of the groups is used as validation. This process is repeated k times using a different group as validation in each iteration, generating k error estimates whose average is used as the final estimate and is the performance seen in Figure 6.

Figure 6 shows that the best performing algorithms are those that use model assembly methods that obtain a weighted average of different algorithms that compose them. There are two techniques in the assembly methods, the first of them is bagging or also known as Bootstrap aggregation, which consists of extracting several samples at random so that each variable in the original population is equally likely to be selected, with these samples the different models are trained separately in submodels, the prediction of the final output combines all the projections of all the submodels, within this technique is Random forest. The other technique used in ensemble methods is boosting, which works by training a model with the entire training set and subsequent models are built by adjusting the residual error values of the initial model, this means that this technique tries to give greater weight to those observations that the previous model estimated. Once the sequence of models is created, the predictions made by the models are weighted to create a final estimate, within this technique is XGBoost.

According to the first cross-validation analyzes for the first *4* stages of the method proposal for event detection, parity is observed mainly within *4* methods as seen in Figure 3, within which knn is discarded. due to its poor performance. Given this great parity within the *4* algorithms, it was necessary to do a statistical significance test with a 5x2 test, within which it was observed which 2 of the algorithms had the best response assertiveness regarding the detection of stalling events. Given this, we conclude the implementation of supervised learning algorithms is a first step towards the automatic detection of stalling events, however, it is necessary to review in the literature algorithms designed for outlier detection, which is our topic of interest. The first step is to understand how outlier detection algorithms work in different learning methods, the first of which is supervised learning, which assumes the included data as correctly labeled. A minimal approximation in this situation is to develop a predictive model for normal and non-normal data classes.

Each hidden data point is compared to the model to decide which class the data point belongs to. The main problem with supervised outlier detection is that usually finding outlier data points is

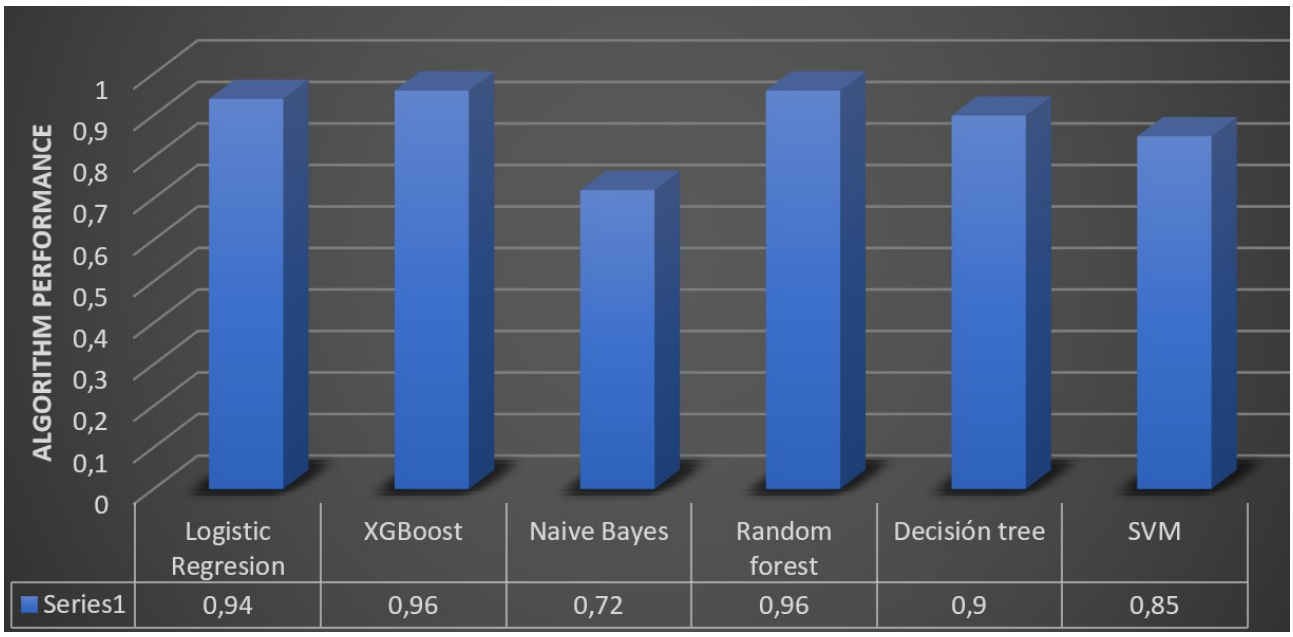


Figure 6: Cross validation results.

more complicated than normal data points, which can lead to class imbalance. For example Decision trees cannot perform well with imbalanced data, while neural networks perform better. Nevertheless, Semi-supervised outlier detection algorithms assume that the labeled data is for the normal class only, the idea with this is that a normal class model is learned and then outliers are detected deviating from the model, this is known as one-class classification.

Therefore, for future works, we will used outlier detection techniques with semi-supervised learning, since our purpose is to detect only stalling events that are considered positive classes. For this purpose, we consider the use of the SSAD taxonomy developed by Khan and Madden in [32]. Of the algorithms that have used this taxonomy we selected ocSVM, LOF, KNN, BRM, ISOLATION FOREST and VAE, for a preliminary test. all of them were evaluated using the ROC (Receiver Operating characteristic) curve in an initial test based on diferent outliers detection types.

The division that exists of the different algorithms in the detection of outliers using semi-supervised algorithms, they are mainly: 1) linear models that are a set of characteristics in linear combinations, of which we use ocSVM; 2) The proximity-based ones that recognize different anomalies in nearby objects, we use the local outlier value (LOF) and k-NN, the first of them recognizes the objects that belong to the normal (negative) class and objects that belong to the class of anomaly (positive) according to proximity, on the other hand k-NN assumes that an object belongs to the same class as its nearby objects and uses feature similarity to discover certain generalities that describe the objects. 3) probability-based models depend on chance; 4) Neural networks, which is a set of numerical methods that try to recognize fundamental relationships in a data set using a process inspired by the functioning of the human brain, from there we did tests with VAE, which is an automatic encoder based on neural networks and uses the recreation error as failure point. Finally, different algorithms were used based on ensemble methods such as BRM (Baggin random miner), ocRF and Isolation forest. Moreover, in the preliminary test the algorithms that performed best were VAE and Isolation forest with AUC values close to 1. These algorithms have such high values because in some cases they allow themselves to be biased by the negative class and we will show more detail about its performance in future works.

5 Conclusions

In recent years, due to the constant growth of video traffic on the Internet, the challenges for content providers have greatly increased in order to guarantee user satisfaction and avoid user desertion, as well as quality algorithms. of objective experience (QoE) are presented as an alternative to constantly monitor user satisfaction.

The QoE estimation is linked to key performance factors that profoundly affect its assessment, such as stalling events, which produce such a great impact that some authors call them the main causes of user desertion. Therefore, it is vitally important to detect and quantify these events in order to increase user satisfaction. Normally, the identification of these events is done by an operator who manually characterizes them, making it a monotonous, costly process, delayed and complex, for this reason it is necessary to develop a method that is capable of calculating the appearance of these events and also has the power to feed the algorithm for estimating the quality of the experience.

The development of this method firstly was done using supervised models but when we performed tests, we realized that they were not good enough given the imbalances in the data, in which we test different balancing techniques and observe their behavior. However, given the results in real data and the difficulty of finding the stalling events as outliers. we decided to developed this method for the detection of anomalies within semi-supervised learning, in order to prioritize performance and avoid biases in the data. According to the model analysis, it was possible to verify that the best performing algorithms are the assembly methods that use the SSDA taxonomy given their ability to use various classifiers to give a score. However, for the present case it is necessary to review other different techniques from XGBOOD and ocRF considering that these algorithms can be biased by the number of samples in the negative classes (normal values) and the few samples in the positive classes (stalling events).

On the other hand, the analyzes with neural networks were able to establish what are interesting algorithms for detecting outliers given their ability to handle large amounts of data and characteristics. In the case of this paper, in the preliminary tests using VAE (Variational autoencoders) and random forest obtained the best results with an AUC value close to *1*, this is believed to be a consequence of the few characteristics that feed the algorithm that encodes and generates the latent space, for this reason it should be tested with algorithms such as MO-GAAL AND SO-GAAL. It is necessary to implement the framework in charge of detecting stalling events to validate the behavior of the algorithms since they were trained and gave good results.

Author contributions

The authors contributed equally to this work.

Conflict of interest

The authors declare no conflict of interest.

Funding

This research is part of the project: "*Fortalecimiento de capacidades de CTeI para la innovación educativa en educación básica y media, mediante uso de la plataforma de recomendaciones de contenidos de vídeo (vLRF) en instituciones oficiales y privadas del municipio de Popayán*" BPIN: 2020000100654. Financed by the *Sistema General de Regalias* of the nation of Colombia and supported by the *Secretaria de educación* of the city of Popayan and the Univsersity of Cauca.

References

[1] Ameigeiras, P.; Azcona-Rivas, A.; Navarro-Ortiz, J.; Ramos-Muñoz, J.J.; López-Soler, J.M. (2012). A simple model for predicting 487 the number and duration of rebuffering events for YouTube flows, *IEEE Communications Letters*, 16, 278–280, https:488//doi.org/10.1109/LCOMM.2011.121311.111682.

[2] Anwar, M.S.; Wang, J.; Ullah, A.; Khan, W.; Ahmad, S.; Fei, Z. (2020). Measuring quality of experience for 360-degree videos in virtual reality, *Science China Information Sciences*, 63, 1–15, https://doi.org/10.1007/s11432-019-2734-y.

[3] Bermudez, H.F.; Martinez-Caro, J.M.; Sanchez-Iborra, R.; Arciniegas, J.L.; Cano, M.D. (2019). . Live video-streaming evaluation using the ITU-T P.1203 QoE model in LTE networks, *Computer Networks*, 165, 106967, https://doi.org/10.1016/j.comnet.2019.106967.

[4] Casas, P.; and Wassermann, S. (2018). Improving QoE prediction in mobile video through machine learning, *Proceedings of the 2017 8th International Conference on the Network of the Future, NOF*, 2018-Janua, 1–7, https://doi.org/10.1109/NOF.2017.8251212.

[5] Cases, P.; Seufert, M.; Schatz, R. (2013). YOUQMON, *ACM SIGMETRICS Performance Evaluation Review*, 41, 44–46, https:530//doi.org/10.1145/2518025.2518033.

[6] Castaneda Herrera, L.M.; Campo Munoz, W.Y. and Duque Torres, A. (2022). Video Streaming Service Identification Using Incremental Learning on Software-Defined Network, PRZEGLĄD EEKTROTECHNICZNY, R. 98 NR 8/2022. https://doi:10.15199/48.2022.08.17

[7] Dong, Y., Zhao, J., and Jin, J. (2017). Novel feature selection and classification of Internet video traffic based on a hierarchical scheme. Comput. Networks, 119, 102-111.

[8] Ericsson, *Ericsson ConsumerLab Report*, April, 2022. [Online]. Available: https://www.ericsson.com/en/reports-and-papers/consumerlab/reports/5g-next-wave, Accesed on 1 May 2023.

[9] Gadaleta, M.; Chiariotti, F.; Rossi, M.; Zanella, A (2017). D-DASH: A Deep Q-Learning Framework for DASH Video Streaming, *IEEE Transactions on Cognitive Communications and Networking*, 3, 703–718, https://doi.org/10.1109/TCCN.2017.2755007.

[10] Ghadiyaram, D.; Bovik, A.C.; Yeganeh, H.; Kordasiewicz, R.; Gallant, M. (2014). Study of the effects of stalling events on the quality of experience of mobile streaming videos. *IEEE Global Conference on Signal and Information Processing*, Global SIP, pp. 989–993, https://doi.org/10.1109/GlobalSIP.2014.7032269.

[11] Ghadiyaram, D.; Pan, J.; Bovik, A.C. (2019). A Subjective and Objective Study of Stalling Events in Mobile Streaming Videos. *IEEE 485 Transactions on Circuits and Systems for Video Technology*, 29, 183–197, https://doi.org/10.1109/TCSVT.2017.2768542.

[12] Ghadiyaram, D.; Pan, J.; Bovik, A.C. (2018). Learning a continuous-time streaming video QoE model. Processing *IEEE Transactions on Image*, 27, 2257–2271, https://doi.org/10.1109/TIP.2018.2790347.

[13] Hoßfeld, T.; Heegaard, P.E.; Varela, M.; Möller, S. (2016). QoE beyond the MOS: an in-depth look at QoE via better metrics and their relation to MOS. *Quality and User Experience* , 1, 1–23. https://doi.org/10.1007/S41233-016-0002-1.

[14] Installations, T.; Line, L.; Systems, D. ITU-T Vocabulary for performance, quality of service and quality of experience. International Telecommunication Union 2017.

[15] Internet users in the world 2022 | Statista, url = https://www.statista.com/statistics/617136/digital-population-worldwide/, urldate = 2022-05-15

[16] Lee, K.; Lee, C. H. and Lee, J. "Semi-Supervised Anomaly Detection Algorithm Using Probabilistic Labeling (SAD-PL)," in IEEE Access, vol. 9, pp. 142972-142981, 2021, doi: 10.1109/ACCESS.2021.3120710.

[17] Mantu, R.; Chiroiu, M.; T,ăpus,, N. (2024). Framework for evaluating TCP/IP extensionsin communication protocols,International Journal of Computers Communications&Control, 19(2),4906, 2024.https://doi.org/10.15837/ijccc.2024.2.4906

[18] Martinez-Caro, J.M.; Cano, M.D. On the identification and prediction of stalling events to improve qoe in video streaming. Electronics (Switzerland) 2021, 10, 1–17. https://doi.org/10.3390/electronics10060753.

[19] Martino, G.; Gruenhagen, A.; Branlard, J.; Eichler, A.; Fey, G. and Schlarb, H. "Comparative Evaluation of Semi-Supervised Anomaly Detection Algorithms on High-Integrity Digital Systems," 2021 24th Euromicro Conference on Digital System Design (DSD), Palermo, Italy, 2021, pp. 123-130, doi: 10.1109/DSD53832.2021.00028.

[20] Meixner, B.; Kleinrouweler, J.W.; Cesar, P. 4G/LTE channel quality reference signal trace data set. Proceedings of the 9th ACM Multimedia Systems Conference, MMSys 2018 2018, pp. 387–392. https://doi.org/10.1145/3204949.3208132.

[21] NamHyunwoo.; KimKyung-Hwa.; CalinDoru.; SchulzrinneHenning. YouSlow. ACM SIGCOMM Computer Communication Review 2014, 44, 111–112. https://doi.org/10.1145/2740070.2631433.

[22] Nnamoko, N.; Korkontzelos, I. Efficient Treatment of Outliers and Class Imbalance for Diabetes Prediction. Artificial Intelligence in Medicine 2020, 104, 101815. https://doi.org/10.1016/j.artmed.2020.101815.

[23] Orozco H, f. (2018). A Survey on Feature Selection Techniques for Internet Traffic Classification, *International Conference on Computational Intelligence and Communication Networks*, 1375–1380, 2015.

[24] Poorzare, R.; Calveras Augé, A. (2023). Deep Learning TCP for Mitigating NLoS Impairments in5G mmWave,International Journal of Computers Communications&Control, 18(4), 4874, 2023.https://doi.org/10.15837/ijccc.2023.4.4874

[25] QoE/stallingdataset, *QoE dataset for stalling event detection*, 2024. [Online]. Available: https://github.com/datasetQoEStall/Dataset, Accesed on 12 Jun 2024.

[26] Robitza, W.; Goring, S.; Raake, A.; Lindegren, D.; Heikkilä, G.; Gustafsson, J.; List, P.; Feiten, B.; Wüstenhagen, U.; Garcia, M.N.; et al. HTTP adaptive streaming QoE estimation with ITU-T rec. P.1203 - Open databases and software 2018. pp. 466–471. 526 https://doi.org/10.1145/3204949.3208124.

[27] Serral-Gracià, R.; Cerqueira, E.; Curado, M.; Yannuzzi, M.; Monteiro, E.; Masip-Bruin, X. An overview of quality of experience measurement challenges for video applications in IP networks. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 2010, 6074 LNCS, 252–263. https://doi.org/10.1007/978-3-642-13315-2_21.

[28] Seufert, M.; Casas, P.; Wehner, N.; Gang, L.; Li, K. Features that Matter : Feature Selection for On-line Stalling Prediction in Encrypted Video Streaming. IEEE INFOCOM 2019 - IEEE Conference on Computer Communications Workshops (INFOCOM WKSHPS) 2019, pp. 688–695.

[29] Staelens, N.; Coppens, P.; Van Kets, N.; Van Wallendaef, G.; Van Den Broeck, W.; De Cock, J.; De Turek, F. On the impact of video stalling and video quality in the case of camera switching during adaptive streaming of sports content. 2015 7th International Workshop on Quality of Multimedia Experience, QoMEX 2015 2015. https://doi.org/10.1109/QOMEX.2015.7148102.

[30] Scikit-learn, *Machine Learning in Python*, 2023. [Online]. Available: https://scikit-learn.org/stable/, Accesed on 15 June 2024.

[31] Tao, X.; Duan, Y.; Xu, M.; Meng, Z.; Lu, J. Learning QoE of Mobile Video Transmission with Deep Neural Network: A Data-Driven 528 Approach. IEEE Journal on Selected Areas in Communications 2019, 37, 1337–1348. https://doi.org/10.1109/JSAC.2019.2904359.

[32] Villa-Pérez, M.E.; Álvarez-Carmona, M.; Loyola-González, O.; Medina-Pérez, M.A.; Velazco-Rossell, J.C.; Choo, K.K.R. (2021) Semi-supervised anomaly detection algorithms: A comparative summary and future research directions. *Knowledge-Based Systems*, 218, 106878, https://doi.org/10.1016/j.knosys.2021.106878.

Journal's webpage: http://univagora.ro/jour/index.php/ijccc/

This journal is a member of, and subscribes to the principles of,
the Committee on Publication Ethics (COPE).
https://publicationethics.org/members/international-journal-computers-communications-and-control

Cite this paper as:

Celis Velez, C. V.; Castañeda Herrera, L. M.; Arciniegas Herrera, J. L.; Bermúdez Orozco, H. F.; (2024). Automatic Detection of Stalling Events using Machine Learning Algorithms, *International Journal of Computers Communications & Control*, 19(5), 6636, 2024.
https://doi.org/10.15837/ijccc.2024.5.6636