A Learning Framework to Improve Video QoE in HTTP Adaptive Streaming SDN for Delivering Live Content

Gowrishankar Subramaniam Natarajan, R. Krishnaveni

Hindustan Institute of Technology and Sciences, Chennai, India.

***Abstract* – HTTP (Hyper Text Transport Protocol) adaptive streaming techniques suffers from start-up latencies and video stalls which impair Video Quality of Experience (QoE). We deal with these issues in 4 steps: i) we analyze the advantages of variable bit rate encoded streams on Adaptive Bit Rate (ABR) streaming and assess the impact of chunk sizes on video quality and scene complexity; ii) we propose a novel machine learning framework through a Gaussian Mixture model k-means clustering to isolate video stall events, the Machine Learning Pipeline runs these Support Vector Machines (SVM) algorithms to accurately predict the occurrence and duration of video stalls; iii) we employ Software Defined Networking (SDN) based OpenFlow controller to dynamically switch, re-prioritize tracks and chunk sizes across tracks to improve the overall Video QoE; iv) we further optimize this through an intelligent prediction of video stall occurrence and stall duration in accordance with network traffic conditions and scene complexity without compromising video quality.**

Keywords - Video Streaming Service, Multi-Channel Video Programming Distributor (MVPD), Video Stall prediction, QoE, SDN, Live Video Stream QoE, optimization

I. INTRODUCTION

Globally, Video traffic over the internet is projected to be 82 percent of all consumer internet traffic by 2021 according to Cisco [1]. It is also estimated that internet video traffic will grow four-fold between 2016 to 2021. Although On-demand video streaming dominates the video streaming traffic on the internet today, Live internet video is projected to grow 15-fold from 2016 to 2021 accounting for 13 percent of internet video traffic by 2021. Despite the obvious demand and rising popularity of streaming applications, video streaming continues to struggle with delivering video with guaranteed quality of service reliably due to varying network conditions and load. However, despite several developments delivering best-in class video quality over IP, especially in live video streaming, still presents a host of challenges. Significant amongst these challenges are network delays/jitters, video freezes and packet losses due to network congestion as studied by several researches [18-21]. The core problem of preventing downstream network congestion due to an increased traffic in proportion the number of users (live stream traffic increases proportionately with increase in user request) and consequently there is a reduction in available bandwidth overall. Adaptive Bit Rate HTTP Live Streaming or HLS proposed by Apple [23] and Dynamic Adaptive Streaming over HTTP (DASH) [22] adopted by the broader industry) streaming is a mechanism that works on the principle of temporal segmentation of video across multiple bit rates of the encoded video. This technique helps improve user perception of content quality, in networks with uncertain bandwidth by switching to lower bitrate streams smoothly but does not address the core problem of minimizing traffic congestion for live streaming. According to Seufert et al [2], Video stalls & startup latencies are two of the major factors impacting Video Quality of Experience (QoE) [30-36] for video delivery as proven in various studies. Of these two major factors, several techniques including reduced HTTP Live Streaming (HLS) segment sizes, pre-warming or pre-loading manifests & pre-buffering segments based on past user viewership patterns, popular channels etc. have been researched and experimented in the past to minimize startup latency of live video streaming. Dobrian et al [5] prove that Video stall is the single largest contributor and impact metric in assessing user engagement & video QoE.

A. Video Stalls

Video stalls occur due to a player buffer underrun which causes video playback to stop. Video player buffer underrun occurs when the average network throughput for the video streaming client is lower than the streaming bit rate causing the buffer to deplete, eventually leading to a point when playback cannot continue any more. Video playback cannot resume until the playout buffer has enough data to continue playback. Video stalls are mainly caused by buffering delays due to network traffic congestion and is particularly observable in live TV due to two primary reasons: First, the video player buffer tends to be shorter for live streaming so that live-lag delay (delay of the video stream playback as compared to same content being available on broadcast) can be minimized. Second, live streaming manifests are updated constantly at the origin and from thereon in the content delivery networks by the encoding pipeline which prevents manifests & segments from being pre-cached beyond the buffer levels in advance. These inefficiencies hamper the overall video quality significantly resulting in a poor end user video quality of experience.

The rest of this paper is organized as follows: Section II discusses previous work on analyzing video stalls and predicting re-buffering ratio. Then, Section III gives an overview of the Learning based framework – is built using machine learning pipeline based on K-SVM [24-29] that uses k-means clustering to correctly group the most significant video stall events and then runs these selected samples through a 2-norm soft margin SVM classifier to predict the occurrence and duration of video stalls. Section IV describes how the Video Stall Optimizer module runs on top of the SDN controller that we built[8] earlier to either a) prioritize video segments for guaranteed video delivery during a network congestion/video stall event and or b) switch to a lower bit rate variant in anticipation of a video stall to ensure a smooth streaming experience for the end client. Section V presents experimental results that analyzes the performance K-SVM based classifier with other popular classifiers such as AdaBoost, GradientBoost, Random Forest & regular SVM [37-43] and we show through extensive data analysis from real production logs that K-SVM based classifiers can predict video stalls, both occurrence and duration on trained videos with high fidelity and accuracy of about 97% for trained videos while the accuracy on untrained videos is more than 92% and Video Stream Optimizer takes proactive action to prioritize and re-route video traffic to avoid video stalls based on this prediction. Finally, Section VI gives conclusion.

II. RELATED WORK

Prior work on video stall analysis and improvements have primarily focused on analyzing the impact of video stalls on overall video QoE, analyzing the impact of network congestion on video stalls, video stall prediction based on chunk sizes, network throughput etc. Krishnan et al [4] showed that with each incremental delay of 1 second spent waiting for video to playback, essentially video stalling, results in a 5.8% increase in viewer abandonment rate. They also infer that viewers tend to watch video for lesser duration as they experience more rebuffering due to video stalls. Vasilev et al [6] propose a Bayesian network model to predict re-buffering ratio or the Stall variable and try to improve the accuracy of prediction through Logistic regression. Zanforlin et al [10] expressed video QoE in terms of the average structural similarity or SSIM index and use this to tag videos with polynomial co-efficients that provide a compact description of its specific SSIM behavior. They further cluster these SSIM characterized video tags into QoE classes and run them through resource management algorithms to handle traffic shaping during network congestion. However, our novel approach looks at measuring stall variable, both in terms of its occurrence and its duration, considering scene-complexity, network access round trip time, chunk size downloaded before and after a stall and the video bit rate variant requested by clients while examining the video session in its entirety. Dimopolous et al [7] considers chunk sizes in predicting video stalls of the video session but does not factor scene complexity, impact of network congestion etc. into consideration. Petrangeli et al[11] provide an innovative Machine Learning framework that studies & predicts video freeze prediction. However, their approach performs classification purely based on measurements obtained by the SDN controller without considering video & client configuration and assumes no knowledge of client request of various chunks at various bit rates chunk sizes or their duration. To the best of our knowledge, none of the prior work, consider that adaptive streaming clients are tuned to look for the peak bit rate mentioned in the HTTP Adaptive Streaming (HAS) manifest/playlist files as the advertised playback bit rate. Also, most of the research is focused around Constant Bit Rate (CBR) based ABR streams while the prime focus and novelty of our work is to study & analyze the impact of scene complexity and chunk sizes on Variable Bit Rate (VBR) [37] based ABR streams and how we can use this apriori knowledge to learn and predict the occurrence and duration of video stalls. This is made possible through Machine learning framework running K-SVM algorithm [9, 24-29] as proposed by Yao et al[9]. Our work also extends to using the video stall & QoE related predictions to optimally shape video traffic through the network during congestion to ensure higher video QoE.

III. VIDEO STALL PREDICTION USING K-SVM BASED LEARNING FRAMEWORK

This section describes the architecture, design & implementation of the K-SVM based Machine Learning framework for predicting video stalls. The core component of this framework is a Video Stall Predictor module. The Video Stall predictor dynamically decides whether a particular chunk/segment being requested by UE (User Equipment) clients is going to experience a stall or buffering at the UE client based on various features or attributes such as average bit rate of the chunk requested, network throughput or congestion metrics, chunk size etc. From a modeling standpoint, this problem can be modeled as a classification problem with 2 classes: Stall event (capturing both occurrence & duration of stall event) & No Stall Event. The classification will be based on metrics tracked by the controller in terms of network throughput in the access networks as well as video & client related metrics such as average video bit rates of the chunks/segments requested by clients, segment sizes. Optionally, we also consider expanding the classification to 3 classes - essentially granularizing the stall event into Major Stall, Minor Stall, No Stall events.

A. Factors Influencing/Causing Video Stall

Next, we critically examine the parameters/metrics that cause/influence video stall or buffering to occur at the

clients. Network congestion detection is a major cause for video stalling to occur. While there are several metrics not limited to the following such as packet loss, packet delays, Time of Day, Concurrent User thresholds, Bandwidth thresholds & end user Video QoE metrics that can help detect and measure network congestion, it is pertinent to look at those attributes that directly impact end user video QoE due to network congestion. Some of these measures such as Time of day & concurrent user thresholds make assumptions about network congestion without actually measuring and verifying if the network is congested in reality and more importantly if the subscriber video QoE is affected because of the network congestion. Similarly, although network traffic reaching bandwidth thresholds has a high degree of correlation with lower Video QOE, it is entirely possible for a link or resource to be at close or maximum capacity without causing subscribers to suffer a lower QoE. Critically, though, there is an underlying assumption that link or resource capacity has a fixed maximum capacity, but this is not necessarily the case either for mobile access networks or for fixed access networks. As a practical result, it is impossible to pick a threshold that is guaranteed to be below the point at which congestive collapse begins. More so, it is difficult to predict the bandwidth threshold at which network congestion starts to affect Video QoE and practically impossible to peg a threshold below which network congestion deteriorates irreversibly. Similar conclusions can be drawn for packet loss as well. It would be ideal if there was one standard metric to measure network congestion which could then be used to analyze and predict when network congestion is triggered. Unfortunately, there is no single metric that provides a good indication of when network congestion is triggered. Instead we look at metrics that have a direct correlation with network congestion, preferably a metric that is caused by network congestion. Round Trip Time on the access network (Access Round Trip Time - aRTT) is one such metric. It has been proven through several research studies [12-17] that aRTT increases dramatically when congestion is present, but it also has high correlation with subscriber assessments of quality of experience. Hence it is safe to state that access round trip time provides for one of the best possible congestion detection mechanisms today. When implemented in real-time, on a per-link basis, such an approach informs the SDN controller precisely where, when, and for whom congestion is manifesting.

As shown in Fig 1, ABR Streaming is traditionally designed for CBR encoded streams. Each track is encoded in CBR by the multi-bit rate ABR Encoder as part of the server AV pipeline. Currently we are seeing major services like Hulu & Netflix moving to VBR while a majority of MPVDs still operate on CBR. However, the challenge is that client adaptation schemes do not account for the VBR encoding within the stream and still expect CBR streams. Client ABR adaptation typically uses the peak bit rate as an indicative size of every segment. This frequently leads to very conservative bitrate choices affecting the overall Video QoE when there are variations in network throughput resulting in a stream switch. VBR is known to have better quality to bits ratio compared to CBR providing significant network bandwidth savings.

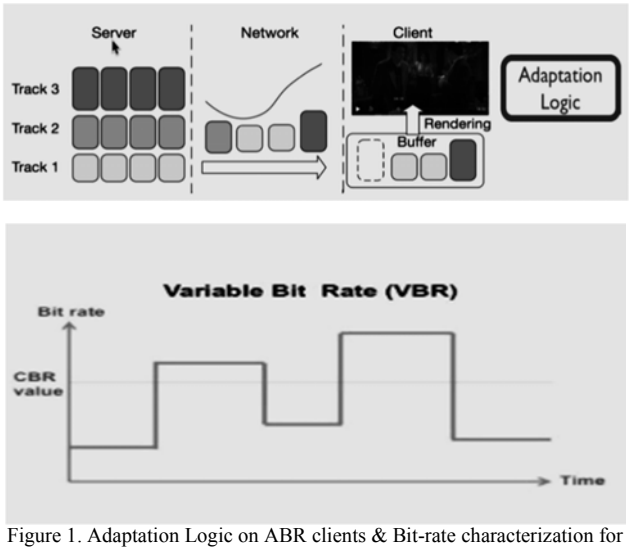

Figure 1. Adaptation Logic on ABR clients & Bit-rate characterization for CBR vs VBR encoded streams

Encoding quality can also be characterized as a function of video chunk size. Encoding quality varies across segments of different sizes, i.e. larger size segments have low quality versus smaller segments have higher quality. From Figure 2 below, it is clear that larger segments get low quality in the same track while larger segments also correspond to more complex scenes. This effectively means that more complex scenes get low quality in the same track. Innately the ABR streaming adaptation on the clients pick a lower quality track when there is a network congestion to minimize video buffering. However, this is quite contrary to what one would expect as it is important to ensure good quality playback for complex scenes. Good adaptation schemes should be scene-complexity aware. It's difficult to deliver good quality track when complex scenes are involved in today's ABR streaming adaptation. However, we can use relative segment size to infer scene complexity to deliver better quality segments for complex tracks. Since relative chunk size within a track is a good measure of scene complexity, our unique solution proposes using chunk size to also ensure better Video QoE by ensuring that a higher quality track is delivered within the constraints of the available bandwidth when network congestion occurs. This is a critical consideration for the Video Stall predictor and Video Stream Optimizer components of our solution to maintain a high video fidelity while minimizing video stalls during network congestion.

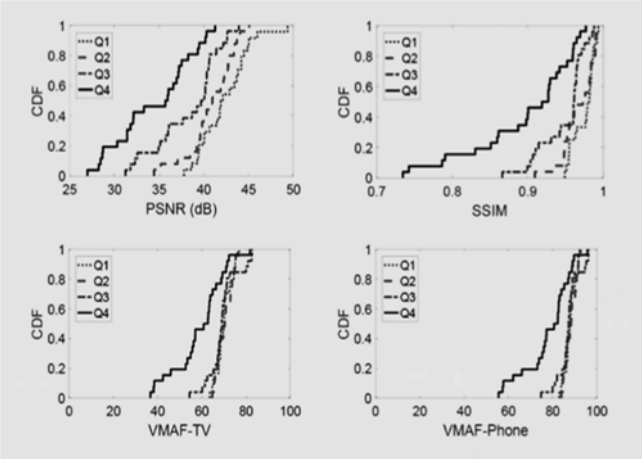

Figure 2 – Quality of segments in a VBR video (Big Buck Bunny, Tears of Steel, Elephant's Dream, Red Bull Playstreet)

As Dimopoulos et al [8] point out, chunk size is also a key factor in predicting video stalls. There is a strong correlation between video stalls and chunk sizes. From Fig 3 below, we observe that whenever video stalling occurs, we see a corresponding sharp reduction in chunk size. Further from the example below, it is apparent that there is significant drop in chunk sizes requested by the clients when the network experiences congestion as depicted in the second graph which shows corresponding increases in access round trip time. As we established earlier, access round trip time is one of the best measures of network congestion. This observation can also be validated by client's ABR streaming behavior which switches down to a lower bit rate stream if the player is rebuffering since chunk sizes for lower bit rate video segment are smaller than that of higher bit rate video segment.

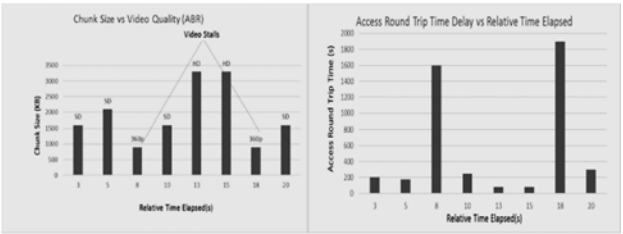

Figure 3 – Chunk sizes versus Video Quality and Network Access Round trip Time Delay over Time

B. Video Stall Predictor Module

The SDN based Open Flow Controller enforces its prioritization algorithm upon every client request for a new chunk of video segment to be downloaded. This prioritization logic is primarily dependent on the Video Stall Predictor module and Video Stream Optimizer Module – key components that lie at the core of our unique solution. The Video Stall Predictor is designed to detect the occurrence of a video stall before & during a chunk request download by the client. It performs the prediction by analyzing the chunk sizes of HTTP ABR requests made by the client, the network access round trip time measured by the SDN controller during the video session and the video bit rate of the chunk being requested by the client. The predictor handles the stall classification problem by initially clustering the data through K-means clustering. K-means is a clustering algorithm that has high efficiency especially with large data sets. Since K-means is an unsupervised learning algorithm, the result clusters are not known before executing the algorithm. Since it may take a few iterations to decide on the number of groups as input and measure of similarity plays a key role in improving the accuracy of the clustering. Since video stalls occur rather infrequently within a video session, determining the initial clusters & centroids which is key to the success of clustering and inherently increases the possibility of converging to a global optimum. Fig 4 below depicts the typical workflow of K-means algorithm.

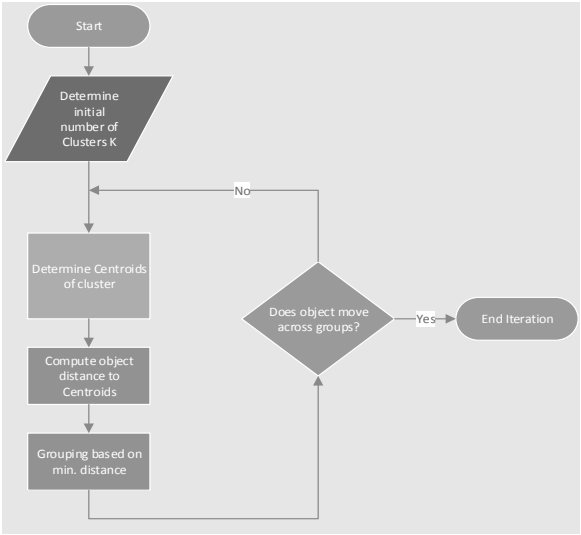

Figure 4 – K-means clustering algorithm

The primary inputs for the Video Stall Predictor are:

• segsizeABR, the chunk size of the video segment being requested by clients on the User Equipment;

• ΔsegsizeABR, the difference between two consecutive chunk sizes of the video segment requested by clients;

• aRTT, the network access round trip time when streaming requests are made by the client

• ΔaRTT, the difference in network access round trip time for consecutive streaming requests made by the client;

• videoBitRate, the requested segment quality level, which is expressed as an integer ranging from 0 (the lowest level) to videoBitRate_Max (the maximum quality level, different from video to video and not available to the controller);

• ΔvideoBitrate, the difference between the qualities of two consecutive segment requests.

The decision boundary between clusters, in our context, major stalls & no stalls as well as major stalls and minor stalls takes relatively fewer iterations through K-means clustering as the location of points along the decision boundary between these clusters can be classified with a cluster based on determining their Euclidian distance based on determining the data distribution in feature space(as captured above) and fine-tuning the location of the decision boundary accordingly. However, the decision boundary between minor stalls and no stalls is relatively fuzzy and is prone to misclassification. The objective of our approach is to leverage the clustering algorithm to select the misclassification points based on the differences of their labels and those of their neighbors, shortlist some of the most informative samples for training the Support Vector Model(SVM). This significantly reduces the size of the training set effectively. Essentially, we check the labels of N-nearest neighbors of data points for whom the cluster label is different from their true label indicating that they are on the edge region of the two clusters. In other words, it's easy to misclassify a minor stall as a no stall and vice versa. In order to prevent this from occurring, it is important to consider data points that are similar to that of the minor stall (or no stall) and use these to train the SVM model.

We use a modified version of the SVM classifier called 2-norm soft margin SVM. When the two classes are not linearly separable (e.g., minor stall and no stall), the condition for the optimal hyper-plane can be relaxed by including an extra term:

$$y_i(\mathbf{x}_i^T\mathbf{w} + b) \geq 1 - \xi_i, \quad (i = 1, \cdots, m)$$

For minimum error, $\xi_i \geq 0$ should be minimized as well as $||\mathbf{w}||$, and the objective function becomes:

$$\textbf{minimize} \quad \mathbf{w}^T\mathbf{w} \; | \; C\sum_{i=1}^{m}\xi_i^k$$

$$\textbf{subject to} \quad y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1-\xi_i, \text{ and } \xi_i \geq 0; \; (i=1,\cdots,m)$$

Here C is a regularization parameter that controls the trade-off between maximizing the margin and minimizing the training error. Small C tends to emphasize the margin while ignoring the outliers in the training data, while large C may tend to overfit the training data.

When $k = 2$, it is called 2-norm soft margin problem:

$$\textbf{minimize} \quad \mathbf{w}^T\mathbf{w} \; | \; C\sum_{i=1}^{m}\xi_i^2$$

$$\textbf{subject to} \quad y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1-\xi_i, \quad (i=1,\cdots,m)$$

Note that the condition $\xi_i \geq 0$ is dropped, as if $\xi_i < 0$, we can set it to zero and the objective function is further reduced.) Alternatively, if we let $k = 1$, the problem can be formulated as

$$\textbf{minimize} \quad \mathbf{w}^T\mathbf{w} \; | \; C\sum_{i=1}^{m}\xi_i$$

$$\textbf{subject to} \quad y_i(\mathbf{x}_i^T\mathbf{w}+b) \geq 1-\xi_i \text{ and } \xi_i \geq 0; \; (i=1,\cdots,m)$$

The algorithm based on 1-norm setup, when compared to 2-norm algorithm, is less sensitive to outliers in training data. When the data is noisy, 1-norm method should be used to ignore the outliers. We decided to deploy the 2-norm soft margin SVM since our data is curated, cleansed for any noise and k-means clustering further ensures that only data samples with massive information (both properly classified and misclassified data sets) are passed for training on the SVM model. This further improves the sensitivity of the prediction in terms of missing minor stalls or worse, predicting no stalls as minor stalls. This method of selecting samples ensures not only that the scale of the sample but also improves the accuracy of the prediction and avoids overfitting. The sample space is efficiently minimized through K-means clustering and the final training data for SVM are considered from the clustering result. Since they contain massive information about the clusters, they contribute to building the 2-norm soft margin SVM model with high fidelity and accuracy.

IV. PROACTIVE VIDEO QOE OPTIMIZATION USING OPENFLOW BASED SDN CONTROLLER

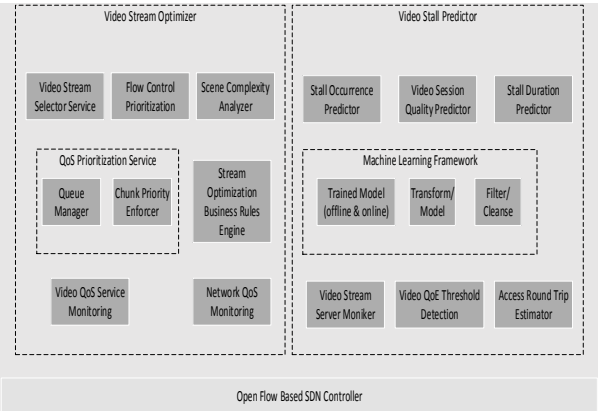

Figure 5 – Video Stall Prediction & Optimization – High Level Architecture Blocks

A. Video Stream Optimizer Module

Chunk sizes of ABR segments are measured by Video Stream Moniker service running on the SDN Controller. This service monitors the HTTP Adaptive Streaming (HAS) manifests being downloaded by the clients, the specific variant tracks being downloaded and also measures the duration and size of each chunk being downloaded by the clients. The risk of freezes is directly proportional to network access round trip time being measured by the OpenFlow SDN controller and the chunk size being downloaded by the client during the video streaming session: when congestion is high, video stalls are more likely to occur. ΔvideoBitRate & ΔaRTT are good indicators of whether the network throughput is improving or not. The risk of a video stall decreases when network throughput improves. In other words, the probability of occurrence of a video stall is low when ΔvideoBitRate is greater than zero and ΔaRTT is lesser than or equal to zero. ΔvideoBitRate is an important feature input for the Video Freeze predictor, as it factors in the playback buffer state of the client.

The high-level software architecture blocks of video stall predictor and video stream optimizer are show in Figure 5 above. During steady state the SDN controller relies on the Video Stall predictor to detect when a client is close to video stall condition and the Video Stream Optimizer module. There are no video stalls in this condition on the client. When the network starts to choke, and available network bandwidth drops, ΔaRTT continues to increase, the clients have to wait longer the segment to complete download before initiating the download of the next segment. This results in a reduction of video chunks in the playback buffer and can result in a video stall potentially. As discussed before, the increase ΔaRTT beyond a certain threshold can be used as a trigger to predict this video stall condition. This estimate along with the ΔsegsizeABR allows the Video Stall Predictor to predict accurately when the stall occurred along with the duration of the stall, since the predictor can estimate the video stall condition fairly accurately by tying in the increase of ΔaRTT threshold with the decrease in ΔsegsizeABR. The last two inputs, videoBitRate and ΔVideoBitRate, factor in the client behavior. Based on ABR logic of the clients, the player requests for a lower bit rate stream when network throughput is lower or the playback buffer within the client is low. In steady state, if the player is streaming a higher video bitrate then it is more prone to a video stall.

These measurements are highly valuable for video stall prediction. Clients would only increase the video bitrate requested only when they perceive that the network available bandwidth is higher. When the Video Stall predictor predicts a major stall outcome, the Video Stream Optimizer enforces traffic prioritization for the client requesting the video segment. The Video Stream Optimizer determines when and whether a client request has to be served through network Quality of Service (QoS) prioritization by judging and evaluating alternate path weights through Djikstra's algorithm and selecting the best end to end video path through Kruskal's algorithm as proposed in the earlier work done by Natarajan et al [8].

The Video Stream optimizer leverages the best alternate video server source for serving the chunk during a network congestion and prioritizes this traffic over others to guarantee quicker delivery of the video segment. The scene-complexity analyzer component within the Video Stream Optimizer analyzes the best quality track that can be delivered within the constraints of available network bandwidth through the network access round trip estimates. This is done by looking at the chunk sizes for the entire video stream/session instead of looking at the next video chunk to be delivered to ensure consistent video quality is delivered throughout the video session instead of bouncing across different variant tracks that may result in a poorer overall video. The Video Optimizer module avoids congesting the stream prioritization queues and allows a fair share of network bandwidth across all clients. This video stall detection is performed by a neural network based on the inputs mentioned above including chunk size, network access round trip time, the requested quality level and the number of consecutive stream prioritization experienced by a client.

The video stall prediction module predicts the probability of occurrence of a video stall event and video stream optimizer provides the best alternate path & prioritizing the video stream for next video segment requested. These two modules working together ensure a better prediction of video stall events due to network congestion, video player buffer under-run events with an accuracy of 97% for trained videos and 92% for untrained videos and resolve video stalls using a better alternate path for video segment delivery and or prioritizing video segments through SDN controller led flow control priority

queues depending upon available bandwidth in the prioritization queue.

V. EXPERIMENTAL RESULTS & ANALYSIS

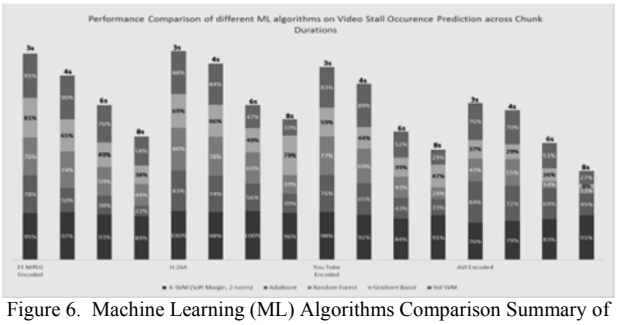

Figure 6. Machine Learning (ML) Algorithms Comparison Summary of Video Stall Occurrence Prediction (% correctly predicted stall occurrence)

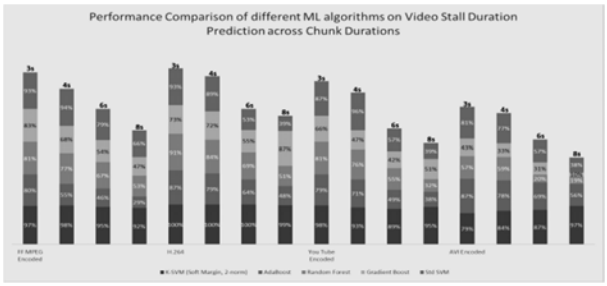

Figure 7. ML Algorithms Comparison Summary of Video Stall Duration Prediction (% correctly predicted stall duration)

The OpenFlow based Floodlight SDN network was simulated using Mininet & Open vSwitch (OVS) connected to a custom built SDN controller. Floodlight is an open source project and we developed a working application to monitor, predict and control QoS on Floodlight based SDN controller. On the physical network, this was implemented using multiple Raspberry PIs(acting as server, client nodes etc.) running OVS and are configured using the modified Floodlight based SDN controller. A major component of the Video QoE framework is the video stall predictor. Based on inputs such as video chunk size, the rate at which chunk size requests by clients increase or decrease, the network access round trip latency, rate of change of access round trip time and the video bit rate requested by the clients, the video stall predictor is able to predict the occurrence of video stalls. The video stall predictor is based on the K-SVM algorithm since K-means clustering improves the scaling efficiency and accuracy of the training samples by grouping the stall & no stall event classes and this data set is trained through SVM which generates optimal hyperplanes to accurately predict major stall, minor stall and no stall events. A large number of video streaming HAS logs and SDN network QoS related logs were collected to accurately train the video stall predictor model. Each data point of the training set is associated with a segment request made by a client and the associated QoS Network Monitor Service logs from the SDN controller and is composed of the six inputs of the predictor (see Section III) and a label (from the data set chosen from the cluster classification from K-means as input to the SVM model as part of the supervised learning) indicating whether the download of the requested segment resulted in a freeze or not. Machine learning algorithm is at the core of the Video Stall predictor. The logs are pre-processed, filtered and cleaned to ensure that we include all different types of video streaming requests, every type of video profile is requested, major streaming format requests are included, network & video configurations are captured in the logs. There was a total of 12500 clients' logs & nearly 17800 SDN & video head-end service logs have been collected, resulting in almost 3.8 million individual segment requests. Only 3.8% of the segment requests are affected by a video stall. Figure 5 & 6 capture the comparison analysis summary of various Machine Learning algorithms on their ability to correctly predict video stall occurrence and video stall duration. From the results, we can observe that K-SVM based machine learning was able to predict both video stall occurrence (96.5%) and duration (97%) with higher degree of accuracy and fidelity compared to other learning algorithms including Gradient Boost & Random Forest. In general, we observe that accuracy of most algorithms including K-SVM deteriorate as the chunk duration increases. This can be explained by the fact that players have to download minimum number of segments (based on their configuration) before they can start playback even when they recover from a network congestion. As the chunk duration increases, the time taken to download the segment will naturally increase resulting in a higher probability of re-buffering and video stall. Potentially there are more major and minor stalls versus no stalls. The ability of the learning algorithm to differentiate a minor stall versus a no stall comes under the scanner for higher chunk durations. While the relative accuracy of most algorithms is lower for higher chunk durations, we can clearly see that K-SVM still performs significantly better than other algorithms. Also, it is to be noted that this performance improvement is consistent across different encodings.

Further we observe that regular SVM based learning offers fairly reasonable degree of accuracy for small chunk durations, but prediction accuracy deteriorates as chunk duration increases. This can also be explained by the fact the SVM tends to perform better when combined with k-means clustering, which helps isolate highly informative data samples in each cluster and also provides misclassification data points across the cluster boundary – this is true especially for improving prediction accuracy around minor stalls and no stalls.

Figure 8 above, shows the prediction accuracy both on video stall occurrence and duration across trained and untrained data. As discussed earlier, Moving Picture Experts Group (MPEG-4) encoded video streaming logs and data samples were not used for training the machine learning models and were used directly for validation. While we observe that overall prediction accuracy is lower

for all learning models, K-SVM has a superior prediction accuracy with video stall occurrence (92%) and stall duration (94%) even on MPEG-4 untrained videos.

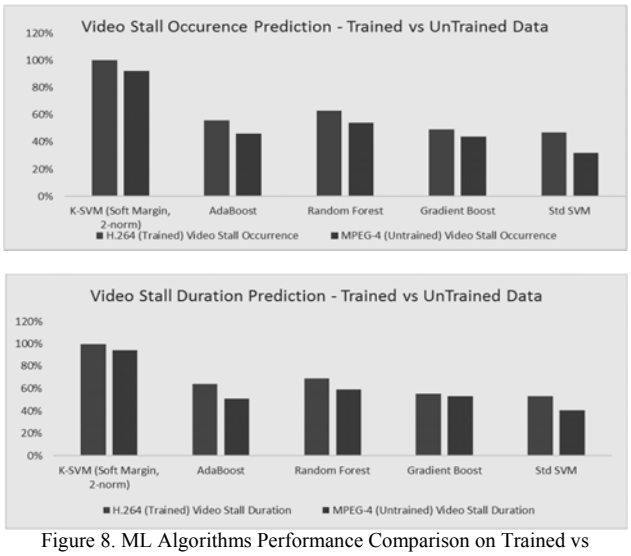

Figure 8. ML Algorithms Performance Comparison on Trained vs Untrained data

We consider these experimental to justify our choice of employing k-means clustering to initially classify the samples into different cluster groups based on the different feature inputs before applying SVM since the data is sharply skewed towards no stall groups.

The entire data set was divided into data for training the Video Stall predictor (80%) with a small percentage of data kept for validating the model (20%). The logs were collected for the following videos: Fast Forward MPEG (FFMPEG) encoded, H.264, YouTube encoded, Audio Video Interleave (AVI) encoded videos are used to train the predictor. The remaining 15% is used as validation set. Moreover, few MPEG-4 encoded videos were additionally used to validate the model as untrained videos. This choice allows to assess to what extent the predictor can adapt to untrained videos, which is a fundamental requirement for the real deployment of the proposed approach. We compare the performance of the K-SVM algorithm in the Video predictor using four other popular classifiers: Random Forest, AdaBoost, GradientBoost and standard SVM classifiers.

VI. CONCLUSION

In this work we presented a novel framework for detecting 3 major factors affecting the video streaming quality, i.e. stall occurrence, stall duration, video stream through flow control prioritization and QOE improvement through alternate path delivery driven by SDN controller. Next, we demonstrated through evaluations from streaming logs in a large scale MPVD broadband and mobile network that the proposed models can detect different levels of video stall occurrence, duration with an accuracy of 97% for trained videos and over 92% for untrained videos. One of the major discoveries of the paper is that the changes in chunk size, network access round trip delays, video bitrate requested by clients have a major impact on the occurrence & resolution of video stalls. The incorporation of these features in our video stall prediction framework results in significant improvements in accuracy, sensitivity of the prediction to detect minor stalls and high fidelity. We showed that the framework can perform very well on real production datasets including the untrained video sets using a few key performance metrics and without the need to instrument clients or additional vantage points, so it can easily be deployed by MPVDs in the future.

REFERENCES

[1] https://www.cisco.com/c/en/us/solutions/collateral/service-provider/visual-networking-index-vni/complete-white-paper-c11-481360.html

[2] Michael Seufert, Sebastian Egger, Martin Slanina, Thomas Zinner, Tobias Hoßfeld, and Phuoc Tran-Gia, “A Survey on Quality of Experience of HTTP Adaptive Streaming”, DOI 10.1109/COMST.2014.2360940, IEEE Communications Surveys & Tutorials

[3] X. Yin, A. Jindal, V. Sekar, and B. Sinopoli, “A Control-Theoretic Approach for Dynamic Adaptive Video Streaming over HTTP,” in ACM, SIGCOMM, 2015, pp. 325–338.

[4] S. Krishnan et al. “Video stream quality impacts viewer behavior: inferring causality using quasi-experimental designs". Networking, IEEE/ACM Transactions on,21(6):2001{2014, 2013}

[5] F. Dobrian, V. Sekar, A. Awan, I. Stoica, D. A. Joseph, A. Ganjam, J. Zhan, and H. Zhang. Understanding the Impact of Video Quality on User Engagement. In Proc. ACM, SIGCOMM, 2011.

[6] Vladislav Vasilev, J´er´emie Leguay, Stefano Paris, Lorenzo Maggi, M´erouane Debbah, “Predicting QoE Factors with Machine Learning”, 2018 IEEE International Conference on Communications (ICC)

[7] G. Dimopoulos, I. Leontiadis, P. Barlet-Ros, and K. Papagiannaki, “Measuring video qoe from encrypted traffic,” in Proc. ACM IMC, 2016.

[8] Gowrishankar Subramaniam Natarajan, R. Krishnaveni, and Frank J. Vijay, “Software Defined Networking (SDN) Based Video Service Using Djikstra’s Algorithm and Kruskal’s Algorithm”, in Journal of Computational and Theoretical Nanoscience, Vol. 14, 1–6, 2017

[9] Yukai Yao, Yang Liu, Yongqing Yu, Hong Xu, Weiming Lv, Zhao Li, Xiaoyun Chen, “K-SVM: An Effective SVM Algorithm Based on K-means Clustering”, in JOURNAL OF COMPUTERS, VOL. 8, NO. 10, OCTOBER 2013

[10] Marco Zanforlin, Daniele Munaretto, Andrea Zanella, Michele Zorzi, “SSIM-based video admission control and resource allocation algorithms”, in Second International Workshop on modeling, measurements and optimization of video performance over wireless networks, 2014

[11] Stefano Petrangeli, Tingyao Wu, Tim Wauters, Rafael Huysegems, Tom Bostoen, Filip De Turck, “A machine learning-based framework for preventing video freezes in HTTP adaptive streaming”, Journal of Network and Computer Applications 94 (2017) 78–92

[12] Daniel S. F. Alves , Katia Obraczka, “An Empirical Characterization Of Internet Round-Trip Times”, INTERNETWORKING RESEARCH GROUP — UCSC, 2017

[13] Srinivas Shakkottai, R. Srikant, Nevil Brownlee, Andre Broido, kc claffy, “The RTT Distribution of TCP Flows in the Internet and its Impact on TCP-based Flow Control”, Center for Applied Internet Data Analysis Research Transactions, 2004

[14] Phillipa Sessini and Anirban Mahanti, "Observations on Round-Trip Times of TCP Connections", University of Massachussets Research
[15] S. Biaz and N. Vaidya. "Is the Round-trip Time Correlated with the Number of Packets in Flight?" In IMC '03: Proceedings of the 3rd ACM SIGCOMM Conference on Internet Measurement, Miami Beach, USA, October 2003
[16] L. Brakmo and S. O'Malley. TCP Vegas: "New Techniques for Congestion Detection and Avoidance". In SIGCOMM '94: Proceedings on Communications Architectures, Protocols and Applications, Portland, USA, October 1994
[17] L.S. Brakmo, L.L. Peterson, "TCP Vegas: end to end congestion avoidance on a global Internet", IEEE Journal on Selected Areas in Communications, pp 1465 – 1480, Oct 1995
[18] Marcos Paredes Farrera, Martin Fleury, Ken Guild, and Mohammed Ghanbari, "Measurement and Analysis Study of Congestion Detection for Internet Video Streaming", Published in JCM, Oct 2010
[19] Kunwadee Sripanidkulchai, Bruce Maggs and Hui Zhang, "An Analysis of Live Streaming Workloads on the Internet", Proceedings of the 4th ACM SIGCOMM conference on Internet measurement, IMC 2004,
[20] Bor-Sen Chen, Sen-Chueh Peng, Ku-Chen Wang, "Traffic Modeling, Prediction, and Congestion Control for High-Speed Networks: Fuzzy AR Approach", IEEE TRANSACTIONS ON FUZZY SYSTEMS, VOL. 8, NO. 5, OCTOBER 2000
[21] Shuai Zhao, Mayanka Chandrashekar, Yugyung Lee, Deep Medhi, "Real-Time Network Anomaly Detection System Using Machine Learning", 11th International Conference on the Design of Reliable Communication Networks(DRCN), 2015.
[22] "Guidelines for Implementation: DASH-IF Interoperability Points", published by DASH Industry Forum, Apr 2015 - https://dashif.org/w/2015/04/DASH-IF-IOP-v3.0.pdf
[23] Apple HLS Live Streaming Specification (Pantos Spec) - https://developer.apple.com/documentation/http_live_streaming
[24] Yu Zong, Ping Jin, Dongguan Xu, Rong Pan, "A Clustering Algorithm based on Local Accumulative Knowledge", Journal of Computers, pp.365-371, vol.8, no.2, 2013.
[25] Corinna Cortes, Vladimir Vapnik, "Support-vector networks", Machine Learning, vol. 20, no. 3, pp.273-297, 1995.
[26] Qiang Wu, "SVM Soft Margin Classifiers: Linear Programming versus Quadratic Programming", Neural Computation, vol. 17, no. 5, pp.1160-1187, 2005.
[27] Juanying Xie, Shuai Jiang, Weixin Xie, Xinbo Gao, "An Efficient Global K-means Clustering Algorithm", Journal of Computers,, pp.271-279, vol.6, no.2, 2011.
[28] S.Sujatha, A.Shanthi Sona, "New Fast K-Means Clustering Algorithm using Modified Centroid Selection Method", International Journal of Engineering Research & Technology (IJERT), pp.1-9, vol.2, no.2, 2013
[29] Lin Yujun, Luo Ting Yao Sheng, Mo Kaikai, Xu Tingting, "An improved clustering method based on k-means", Fuzzy Systems and Knowledge Discovery(FSKD), pp. 734-737, 2012.
[30] Tasnim Abar, Asma Ben Letaifa Sadok El Asmi, "Machine Learning based QoE Prediction in SDN networks", 13th International Wireless Communications and Mobile Computing Conference (IWCMC), 2017
[31] R. Mok et al. "Measuring the quality of experience of HTTP video streaming". In IFIP/IEEE, International Symposium on Integrated Network Management (IM), pages 485{492. IEEE, 2011
[32] T. Hofeld et al. "Assessing effect sizes of influence factors towards a QoE model for HTTP adaptive streaming". In 6th International Workshop on Quality of Multimedia Experience (QoMEX), pages 111{116. IEEE, 2014
[33] Ayub Bokani, S. Amir Hoseini, Mahbub Hassan, Salil S. Kanhere, "Implementation and Evaluation of Adaptive Video Streaming based on Markov Decision Process", IEEE ICC 2016 - Mobile and Wireless Networking Symposium
[34] Alberto Testolin, Marco Zanforlin, Michele De Filippo De Grazia, Daniele Munaretto, Andrea Zanella, Marco Zorzi, Michele Zorzi, "A Machine Learning Approach to QoE-based Video Admission Control and Resource Allocation in Wireless Systems", 13th Annual Mediterranean Ad Hoc Networking Workshop (MED-HOC-NET), 2014
[35] Irena Orsolic, Dario Pevec, Mirko Suznjevic and Lea Skorin-Kapov, "YouTube QoE Estimation Based on the Analysis of Encrypted Network Traffic Using Machine Learning", IEEE Globecom Workshops (GC Wkshps), Dec 2016
[36] Christos G. Bampis, Alan C. Bovik, "Learning to Predict Streaming Video QoE: Distortions, Rebuffering and Memory", IEEE Transactions on Image Processing (Volume: 27, Issue: 5, May 2018), pp 2257-2271
[37] Le, Hung & Nguyen, Hai & Pham Ngoc, Nam & Pham, Anh & Cong Thang, Truong. (2015). A Novel Adaptation Method for HTTP Streaming of VBR Videos over Mobile Networks. Mobile Information Systems. 2016
[38] Shuo Wang, Xin Yao, "Multiclass Imbalance Problems: Analysis and Potential Solutions", IEEE Transactionson Systems, Man, and Cybernetics —PART B: Cybernetics, VOL. 42, NO. 4, August 2012
[39] Ji Zhu, Hui Zou, Saharon Rosset and Trevor Hastie, "Multi-class AdaBoost", Statistics & its interface, Vol 2 (2009), pp 349-360
[40] Schapire, R. and Singer, Y. (1999), "Improved boosting algorithms using confidence-rated prediction", Machine Learning 37297–336
[41] Zhang, T. (2004), "Statistical analysis of some multi-category large margin classification methods", Journal of Machine Learning Research 5 1225–1251
[42] Schapire, R., Freund, Y., Bartlett, P., and Lee, W. (1998), "Boosting the margin: a new explanation for the effectiveness of voting methods", Annals of Statistics 26 1651–1686
[43] Casas, Pedro & Wassermann, Sarah. (2017). Improving QoE Prediction in Mobile Video through Machine Learning. 10.1109/NOF.2017.8251212.