Towards Real-Time Stream Quality Prediction: Predicting Video Stream Quality from Partial Stream Information

Amy Csizmar Dalal*, Emily Kawaler, and Sam Tucker

Department of Computer Science, Carleton College
Northfield, MN, USA
{adalal, kawalere, tuckers}@carleton.edu

Abstract. While mechanisms exist to evaluate the user-perceived quality of video streamed over computer networks, there are few good mechanisms to do so in real time. In this paper, we evaluate the feasibility of predicting the stream quality of partial portions of a video stream based on either complete or incomplete information from previously rated streams. Using stream state information collected from an instrumented media player application and subjective stream quality ratings similar to the Mean Opinion Score, we determine whether a stream quality prediction algorithm utilizing dynamic time warping as a distance measure can rate partial streams with an accuracy on par with that achieved by the same predictor when rating full streams. We find that such a predictor can achieve comparable, and in some cases markedly better, accuracy over a wide range of possible partial stream portions, and that we can achieve this using portions of as little as ten seconds.

Keywords: Quality of Experience, Quality of Service (QoS), Streaming Media, Measurement, Performance, Reliability.

1 Introduction

Determining the subjective, user-perceived quality of a media stream in a scalable and quantifiable way is a difficult problem. As with all Internet-based applications, there is a complex interplay between network congestion conditions and the effect these congestion conditions have on application performance. Knowing how end users perceive the quality of audio and video streamed on-demand over computer networks, and the relationship between stream quality and network congestion, can lead to better design of streaming protocols, computer networks, and content delivery systems.

A number of studies have explored the idea of combining the ease and convenience of objective measurements with the information offered by a subjective

* This work is sponsored by grants from the Howard Hughes Medical Foundation and from Carleton College. Early versions of this work were sponsored by Hewlett-Packard Laboratories.

N. Bartolini et al. (Eds.): QShine/AAA-IDEA 2009, LNICST 22, pp. 20–33, 2009.

rating such as the MOS [1] to discern user-perceived stream quality. Some, like [2] and [3], correlate measurements on both the sender and receiver sides. Others, like [4] and [5], use the Emodel [6], an objective mechanism for assessing audio quality using transmission parameters. An alternate approach is to utilize application-layer objective metrics, taken at the client's machine through an instrumented media player application [5,7,8,9]. These approaches allow one to take measurements as close to the user as possible, in some cases without requiring the user's participation, providing a more accurate assessment of the state of the application at any given time.

In previous work [10,11,12], we demonstrate that objective data collected from an instrumented media player application can be used to *predict* subjective quality ratings with a high degree of accuracy (typically 70-90%) when input into a stream quality predictor that assigns ratings using a nearest-neighbor heuristic and dynamic time warping (DTW) as its distance measure. While our success rates are quite high, we base our predictions on complete stream data well after the stream has finished playing out. A more practical approach would be to predict subjective quality ratings in real time, as the stream is playing out. Modeling such a system on our previous work, such a predictor would be trained using objective and subjective measurements from past streams ahead of time, and apply this information to the task of predicting quality ratings for streams as they play out.

An important intermediate step in this process is to determine if this same stream quality predictor can accurately predict the user-perceived quality of a video stream using only partial information about the stream to be rated and/or the streams in the training set, and if so, if some portions are better or worse than others in terms of accuracy. This is the focus of this paper.

The input to our stream quality predictor consists of set of video stream state information, namely packet retransmissions, collected from an instrumented media player application for 228 video streams, with corresponding user-perceived quality ratings for these same streams. We first train the predictor using all of the available data and ratings for all streams, then test the predictor by having it predict ratings for ten-second and fifteen-second portions of these same streams. In addition, we train the predictor on various portions of the original streams, and then test the predictor on portions of the original streams as well. We compare the predictor's accuracy in these scenarios to the predictor's accuracy when training and testing on full streams. Our results show that in most cases, our predictor is about as accurate using partial stream data as it is using full stream data. We also demonstrate that we fare slightly better when we use partial stream data for both training and testing than when we train the predictor on full streams to predict the quality of partial streams, and that this holds for many combinations of training and test stream portions, even ones that are dissimilar in time from each other. In some cases, we can consistently achieve hit rates above 90%, in particular when we select similar portions (in time) from similar streams. Finally, we show that we need as little as ten seconds of data to achieve these hit rates.

The rest of this paper is structured as follows. We review the characteristics of video streams that can be exploited to infer stream quality in Sect. 2. We discuss our stream quality prediction algorithm, describe how partial stream prediction can be used to prove the feasibility of real-time stream prediction, and present the methodology we use to form partial streams from our data, in Sect. 3. In Sect. 4, we describe the source data for the experiments and the mechanism we use for evaluating predictor accuracy. Section 5 presents the results of our experiments and discusses their implications on stream quality prediction system design. We conclude the paper in Sect. 6 and highlight areas for future work.

2 Video Stream Characteristics

We have developed an instrumented version of Windows Media Player [12] that collects application-layer data about the state of a media stream at predefined intervals (currently, one second) using ActiveX hooks. Figure 1 shows an example of the data collected by our tool for a stream several minutes in duration that experiences a moderate level of network congestion. The plot illustrates a few ways in which the media player reacts to the presence of congestion on the network: for example, the number of retransmitted packets increases and the rate at which packets are received decreases as soon as congestion is detected on the network, while the number of lost packets rises later on in the plot. The plot also shows a transient period, several seconds in duration, at the start of

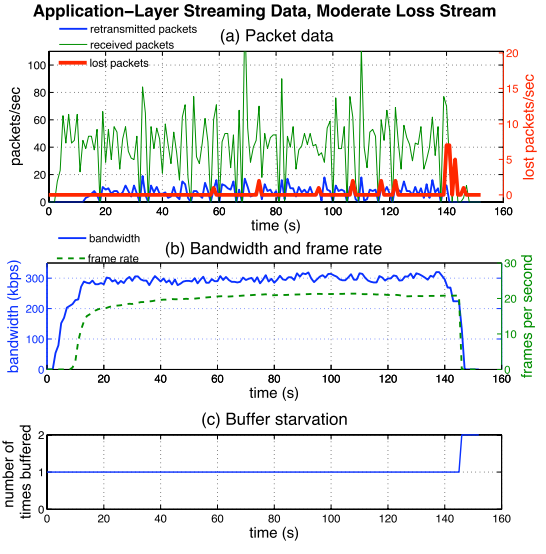

Fig. 1. Time-series data collected by the instrumented media player application. (a) Packet-level data: retransmitted packets and received packets are on the left y-axis, lost packets on the right y-axis. (b) Bandwidth and frame rate, on the left y-axis and right y-axis, respectively. (c) Buffer count, including the initial startup buffering period.

the stream, where the packet reception rate, bandwidth, and frame rate all rise to their steady-state levels as the player and server negotiate the connection between them. There is a comparable transient period at the end of the stream, where we see an uptick in the number of lost packets reported and a buffer starvation event occurring, as the player and server account for packets that will not be able to be recovered by the end of the stream.

Streams that have been exposed to similar levels of network congestion will most likely show similar patterns of retransmitted packets, lost packets, etc., even if the exact occurrences and durations do not match exactly. Streams that exhibit these similar characteristics will also exhibit similar user quality ratings, particularly once individual user biases have been accounted for. Ideally, there will be one or more measurements that most strongly reflect these quality ratings. In [12], we found that retransmitted packets are the most strongly influential on user-perceived stream quality. Thus, we can reduce the stream state information to just this one measurement over time, and discern stream similarity based on this measurement.

3 A Methodology for Real-Time Stream Prediction Using Partial Streams

Exploiting objectively-measured stream data to predict user-perceived stream quality ratings resembles problems that are classic data mining problems. By comparing patterns within the application layer metrics to user quality ratings for that stream, we can understand the effects of network congestion on user perception of stream quality.

Our stream quality prediction algorithm is described in detail in [10]; here, we briefly summarize its operation. Our particular problem calls for using knowledge of pre-labeled data to predict labels on new data[13,14,15]. The goal is to produce a predictor by training, i.e. running a data mining algorithm, on a set of labeled data. The predictor can then be tested on unlabeled data. Our data consists of a set of measurements collected from the instrumented media player on a given set of streams. The labels in this case are the quality ratings assigned by users who watched these streams as measurements were being collected (see Section 4 for details on how this data was obtained).

Our predictor uses a *nearest neighbor* algorithm, which locates all of the rated streams in the training set which are closest to the unrated stream, subject to some distance metric. A single rating is produced from the set of ratings for the closest points: if there is one nearest neighbor, assign the unrated stream that stream's rating; otherwise, compute the mean of the ratings and assign that value to the unrated stream. The distance metric used by our predictor is an extension to *dynamic time warping* (DTW), a generalization of Euclidean distance designed for use with time series data, that facilitates its use on multi-dimensional time series [16]. Briefly, DTW is based on the assumption that two time series may be quite similar, even if the precise timing between the two series is misaligned. While DTW aligns the start and end points of each time series

(stream), it allows points in mid-stream to align with the closest appropriate point. This fluidity often results in more accurate predictions and pattern identifications. A stream of unknown quality that exhibits packet loss on a periodic basis, for example, is expected to have similar quality to another stream that also loses packets periodically. However, it should not be a requirement for similarity between such streams that the packet losses occur *at precisely identical times*. To reduce the computational time and (quadratic) complexity inherent in DTW, we apply two optimizations: the popular Sakoe-Chiba band [17,18], which limits the distance that one time series can shift relative to the other; and Keogh minimum bounds [17], to quickly determine candidates for the set of nearest neighbors.

Preparing this predictor is a two step process. The first step, *training*, consists of reading in and storing the state information collected for a single stream rated by a single individual. The second step, *tuning*, consists of selecting the proper predictor parameters or inputs: K, the number of neighbors to use for predicting the quality of a stream, and w, the width of the Sakoe-Chiba bands, which we do using a leave-one-out cross-validation procedure on each training set.

In previous work, we have trained and tested this predictor using all of the data collected from a single media player application for a single user who watched and rated a particular media stream subjected to a particular level of network congestion. While doing so gives us a good idea of the accuracy of the predictor under the best of circumstances, it is not realistic. A production stream quality prediction system will have to assign ratings to incomplete streams. To mimic these circumstances, and as an important intermediate step to determine the feasibility of predicting stream quality in real time, we consider mechanisms for reducing the available information about a stream in the predictor's training phase and test phases.

One approach is to train our predictor using all of the information available from each stream, then have the predictor assign ratings to smaller portions of the available test streams. The advantage of this approach is that the predictor does not require full stream information before assigning a rating to a test stream. The disadvantage is that DTW can perform poorly when training and test stream sizes are severely mismatched; since DTW fixes the start and end points of the streams, it compacts the longer (training) streams to match the shorter streams (to be rated), which may mean that we lose valuable information about the longer stream in the process. Another approach is to train our predictor using only the information that it is likely to have about the streams it will be rating: in this case, smaller portions of the available training streams. The advantage to this approach is that the stream sizes are similar, allowing for potentially better matches by DTW, as we have shown previously [10].

It is desirable to determine the smallest portion of a stream for which our predictor achieves accurate stream quality predictions, as well as the "optimal" location in the training and test streams from which to take these samples. A unique challenge in this case is to determine how to best select comparable intervals from the full streams, which have different durations, such that we can

easily compare shorter and longer streams. We select arbitrarily small portion sizes, ten and fifteen seconds, and divide each stream into smaller substreams of these lengths. We also divide each stream into the same number of substreams, regardless of the total length of the stream, by taking our portions at certain percentages from the start of the stream (in this case, between 1% and 90%). Thus, a thirty-second stream and a four-minute stream will yield the same number of substreams. This means that the smaller streams will be somewhat overrepresented in our training set and that the longer streams will be somewhat underrepresented in our training set. However, it also means that we can easily match up substreams from different source streams, without worrying about not having an analogous period from the source stream.

Different streams will have different stream state characteristics which will vary over the lifetime of the stream. The transient and steady-state periods, for instance, will have different characteristics; the steady-state period's state information may also reflect the current level of action in the video, or the duration from the start of the stream. To determine how best to match up different portions of the stream during the training and test phases of our prediction algorithm, we use the following approach. We first train our predictor as usual with the full stream information. When testing, we rate each substream using the full stream training information. This demonstrates how well the predictor does when it has less information in the testing phase than in the training phase. We then train the predictor using, in turn, each possible substream, and then test it on each possible substream. This demonstrates how well training and test stream intervals match up when taken from similar and dissimilar points in the stream, as well as from similar and dissimilar source streams.

4 Experiments

Our data collection mechanism, testbed network, and experimental setup are described in detail in [10]; we summarize these briefly below.

Our data collection testbed consists of a set of 14 client machines on a subnet of a small campus network, and a media server on an isolated subnet with a router which runs NIST Net software [19]. The media server is a 2.4 GHz Pentium processor machine with 512 MB of RAM, running Windows Server 2003 and Windows Media Server 2003 software, streaming RTP over UDP. The NIST Net router is a 700 MHz processor machine with 512 MB of RAM, running Linux kernel 2.4.21-27 and NIST Net version 2.0.12. The client machines have 3.4 GHz Pentium processors and 1 GB of RAM and run Windows XP SP2 and Windows Media Player version 10.

Table 1 lists the source streams used in this study, which were selected to provide some variety in duration, style, content, and amount of action. NIST Net applies randomly-distributed packet losses on the testbed network, over the duration of each stream, at percentages of 0, 5, 15, and 25; there was no additional delay or delay jitter applied to the network. The network packet losses, which we determined experimentally, are higher than those typically seen

Table 1. Description of the source streams used in this study

Name	**Time (mm:ss)**	**Action Level**	$\overline{BW}$ **(kbps)**
Ad	0:30	Moderate	273
Trailer	2:22	High	273
News	4:09	Moderate	331

in computer networks, both to overcome the mechanisms that Windows Media Player uses to mitigate the effects of network congestion [9] and to affect the media experience in an obvious fashion that influences the streams in the same manner each time.

We showed our study participants each of the three streams twice, once with no packet loss introduced and once with either 5, 15, or 25% packet loss, blindly randomized over the participants. The participants rated the audio, video, and overall quality of each stream using seven-point scales, which allows for slightly finer granularity in participant responses [20,21]. The measurement tool collected data from each stream simultaneously. From these experiments, we collected data from a total of 38 participants and their respective client machines, yielding data for 228 streams in total.

Normalizing user ratings mitigates the factors that affect user ratings, such as individual sensitivity to encoding differences, by basing ratings on the biases of the particular user in question. We use a z-score to normalize ratings, $z_s = \frac{r_s - \bar{r}}{\sigma_r}$, where r_s is the user's quality rating for stream s, $\bar{r}$ is the average of the user's quality ratings on all streams viewed, and σ_r is the standard deviation of the user's quality ratings on all streams viewed.

We measure prediction accuracy by a *hit rate* metric, where hit rate is the percentage of time a prediction falls within 0.8 standard deviations of the user's z-score for that stream. This corresponds to approximately plus or minus one point on the raw seven-point scale.

Using the data we collected, we first trained our predictor on each of the three full streams, then used this training data to assign ratings to the ten and fifteen second long substreams described in Section 3. We then trained the predictor on each substream and used this training data to assign ratings to each substream. In the discussion below, we refer to substreams as "partial streams".

5 Results

In this section, we present our results for the two experiments described above: training on full streams and rating partial streams, and both training on and rating partial streams. For space reasons we only present the results for the ten-second stream portions; the results for the fifteen-second stream portions are nearly identical. As a point of reference, Table 2 lists the hit rates achieved by our predictor when both training and testing on full streams. With one exception,

Table 2. Hit rates for the stream quality predictor when training and testing on full streams

Training Stream	Test Stream			Params
	Ad	Trailer	News	{K, w}
Ad	88.2	80.3	80.3	3, 1
Trailer	72.4	89.5	80.3	6, 0
News	64.5	75.0	86.8	8, 2

the hit rates achieved by this predictor are all above 72%, with hit rates above 80% for the majority of the train/test stream scenarios.

5.1 Assigning Ratings to Partial Streams with Full Stream Training Sets

Figure 2 illustrates the accuracy of the predictor when training on full streams and assigning ratings to ten second portions of the streams. The x-axes indicate the percentage offset from the start of the stream from which the portion was taken. The plots show a clear transient period at the start of each testing stream, lasting anywhere from 7% to 30% from the start of the stream, during which hit rates are below 60%. They also show a transient period at the end of the stream for Ad, but not for Trailer or News. During the steady-state period, hit rates fluctuate between 70% and 85% when either Ad or Trailer is used as the training stream. These hit rates are comparable to slightly lower than the hit rates achieved by the predictor when training and testing on full streams. Hit rates are also comparable when the predictor assigns ratings to steady-state portions of the Ad stream when News is the training stream.

When News, the longest stream, is the training stream, hit rates decrease significantly (to below 70%) when the predictor rates portions of either Trailer or News. Here we have hit upon a possible limitation of our system: the longer the training stream, the less its characteristics match portions of the test streams. Our results indicate that this limit is somewhere between the durations of Trailer and News (2:20 and 4:10).

5.2 Assigning Ratings to Partial Streams with Partial Stream Training Sets

Figure 3 plots the accuracy of the predictor when training on and rating ten second portions of the streams. The percentages along the x- and y-axes indicate the percentage offset from the start of the stream from which the ten second portion was taken. The z-axis shows the hit rate for the predictor for a given training and test stream combination.

When Ad is the test stream (the stream to be rated), shown in the first column of plots in the figure, the best hit rates occur from about 40% of the way through the test stream until about 70% of the way through the stream, with hit rates typically between 80 and 90%. This is a significant improvement

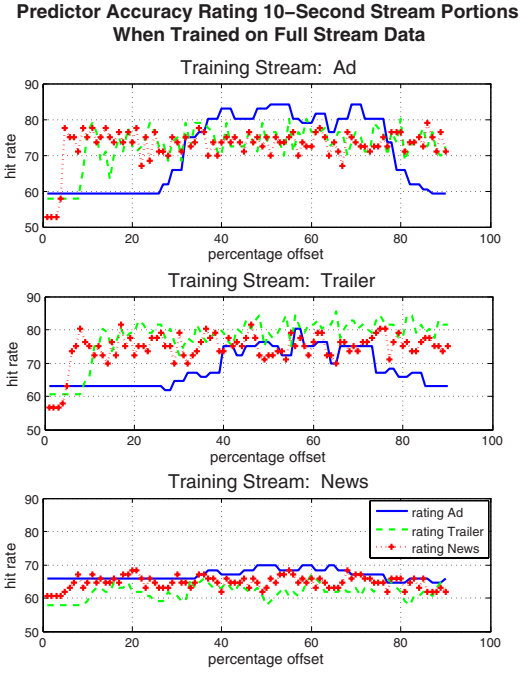

Fig. 2. Hit rates when training on full streams and testing on 10-second portions of the streams, for all combinations of training and test streams

over the predictor's accuracy when using the full streams for training and testing (which are 72 and 65% for Trailer and News as training streams, respectively). Here, using smaller portions of streams that are dissimilar in length when both training and testing actually benefits the predictor, removing the pathologies that make it difficult to accurately match up the two streams when using DTW.

When portions of Ad are used as the training stream and portions of either Trailer or News are used as the test streams, hit rates are between 75 and 85% during the steady-state period, These hit rates are comparable to slightly lower than the hit rates achieved when the predictor trains and rates full streams.

The plots for the training/test stream combinations Trailer/Trailer, Trailer/News, News/Trailer, and News/News all exhibit similar characteristics to each other. During the steady-state periods, the predictor successfully rates the test stream between 80 and 90% of the time for News/News and Trailer/Trailer, and between 75 and 85% for News/Trailer and Trailer/News, which is comparable to the hit rates when training and testing on full streams.

Figure 4 shows the portions of the training and test streams for which the predictor is most accurate. The plots show that when the training and test stream portions are taken from the same source stream, the best hit rates cluster around the diagonal, which means that the predictor does best when the training and

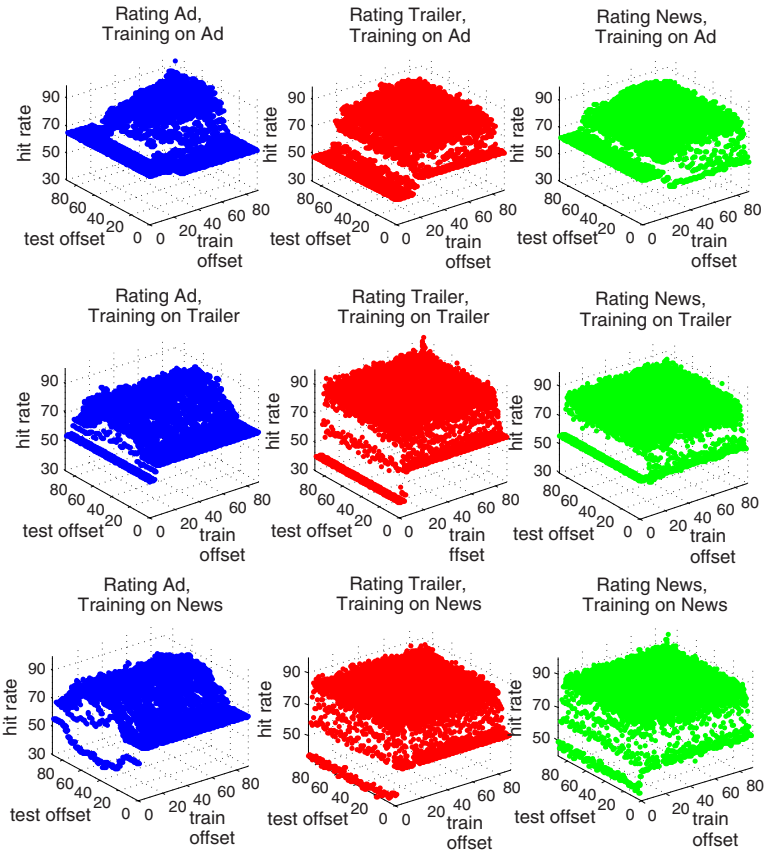

Fig. 3. Hit rates when training and testing on 10-second portions of the streams, for all combinations of training and test streams

test streams are selected not just from the same source stream, but from the same portion of the stream. In fact, the most significant result here is that the predictor is actually able to achieve hit rates over 90% for Ad and News and over 95% for Trailer under these circumstances. When the training and test streams are taken from different source streams, the best hit rates are between 85 and 90%, which is still rather high. We also see that we do not necessarily have to pull our training and test stream portions from similar time periods in the stream to achieve such high hit rates.

5.3 Discussion

Our results show that accurate predictions are quite possible, even with intervals as small as ten seconds long, when only partial information about a stream is available. In general, hit rates were at least in the neighborhood of, if not better

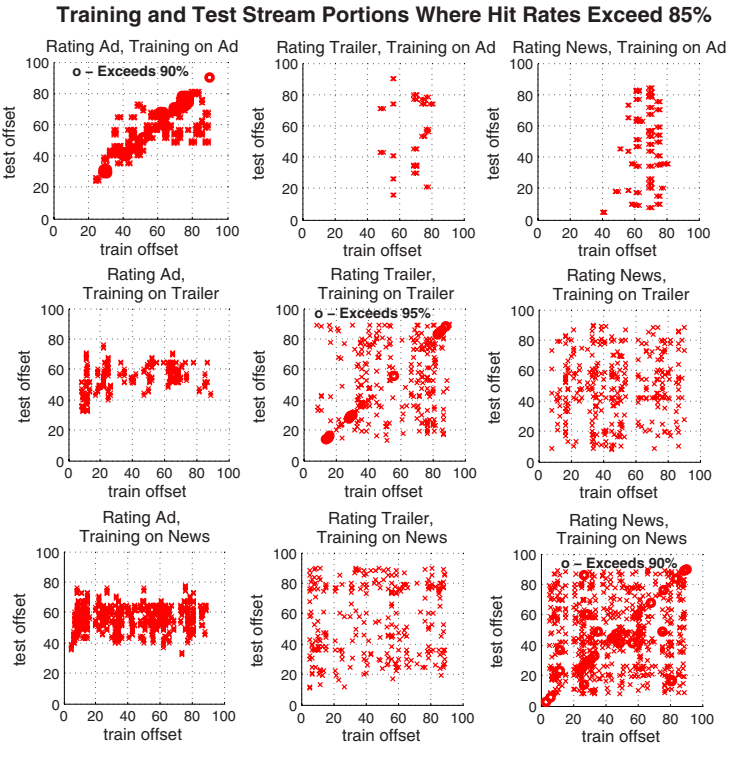

Fig. 4. Training and test stream combinations that yield hit rates above 85%. The Ad/Ad and News/News plots show data for hit rates above 88%, while the Trailer/Trailer plot shows data for hit rates above 90%. The circles indicate hit rates that are better than 90% (or 95%, in the case of Trailer/Trailer).

than, the hit rates achieved by the same predictor when using all available stream information for training and testing.

In general, a real-time stream quality prediction system should avoid training or testing during the transient period of the streams. Training and testing during these periods leads to unreliable results and inaccurate ratings, because the characteristics of this portion of the stream are dissimilar to the characteristics of the steady-state portions of the streams. With very short streams, where the transient period is relatively long compared to the length of the stream, we should also avoid training and testing during the end-of-the-stream transient period, since for short streams we often see a big uptick in certain measurements to help make up for the lack of recovery time during stream play-out. For longer streams, our results do not show an appreciably noticeable end-of-stream transient period.

If we use all available stream information (full streams) in the training phase of our predictor, then the predictor can accurately predict stream quality ratings on ten-second stream intervals between 70 and 85% of the time, assuming that

one of the shorter streams (Ad or Trailer) is used as the training stream. If we use News, our longest stream, as the training stream, then we are not able to achieve such accurate results, due to the compacting of the longer stream by DTW.

If we use partial stream information in both the training and testing phases of our predictor, then as long as the predictor avoids training or testing during the stream's transient period, it has a lot of freedom in terms of choosing appropriate training and test intervals. This is particularly true when the training and/or test stream portions are pulled from Trailer or News. This means that we can achieve results that are just as accurate, on balance, if we take the training and test intervals from very different portions of the stream as if we took them from similar portions of the stream. In a real-time stream prediction system where storage space and time to locate and load training results may be at a premium, this means we can pre-select a few portions of a stream and use any of them as our training data when assigning a rating to a new stream.

It is possible to find training and test stream portions for which the predictor can achieve better than 85% accuracy. If we can guarantee that our training and test streams have similar characteristics, as is the case when our training and test streams are both taken from Ad, or from Trailer, or from News, and take our training and test stream portions from approximately the same time period, our predictor can actually achieve hit rates above 90% (or above 95% in one case).

6 Conclusion

This paper examines the feasibility of real-time stream quality prediction, by studying whether a nearest-neighbor stream quality predictor using DTW as a distance measure can accurately rate streams based on partial stream state information. To answer this question, we examine two scenarios. In the first scenario, we train our predictor with full stream state information and attempt to rate streams where we have removed all but a small portion of stream state information. In the second scenario, we train our predictor on small portions of the full streams and then attempt to assign ratings to small portions of the full streams. We have shown that there is a wide range of training and test stream combinations that yield acceptably high hit rates, on par with or better than that achieved by the same predictor when using full stream information for both training and testing, and that we can do so using as little as ten seconds of information from each stream. This means that a stream quality prediction system operating in real time does not have to worry about using training and test streams from the same time period in the stream; training portions pulled from the end of a stream can accurately rate portions from earlier in the stream, and vice versa. We have also demonstrated that our predictor is especially accurate when we take ten second portions of the streams that are nearly identical in stream characteristics and in where in the stream they occur, achieving hit rates above 90 or even 95%. This indicates that it is possible to design a highly

accurate stream quality predictor with minimal stream information (as little as ten seconds from each stream), if we know some characteristics of the training and test streams *a priori*.

This work represents a proof-of-concept of the feasibility of real-time stream quality prediction systems, and as such there are extensions of this work that we are currently pursuing. Our data set consists of videos streamed over UDP, rather than the more ubiquitous TCP; we are currently in the process of collecting more data for streams over TCP. From a systems perspective, we are also working on a very basic prototype system to determine how best to collect, store, and evaluate stream data in real time. Finally, we are modifying the measurement tool and measurement infrastructure to enable us to collect stream ratings at intermediate points during a stream, to further improve the accuracy of our stream quality predictor.

References

1. P.910, I.T.R.: Subjective video quality assessment methods for multimedia applications. Recommendations of the ITU, Telecommunications Sector
2. Wolf, S., Pinson, M.H.: Spatial-temporal distortion metrics for in-service quality monitoring of any digital video system. In: Proceedings of SPIE International Symposium on Voice, Video, and Data Communications, Boston, MA (September 1999)
3. Ashmawi, W., Guerin, R., Wolf, S., Pinson, M.H.: On the impact of policing and rate guarantees in Diff-Serv networks: A video streaming application perspective. In: Proceedings of SIGCOMM 2001, San Diego, CA (August 2001)
4. Clark, A.D.: Modeling the effects of burst packet loss and recency on subjective voice quality. In: Proceedings of the IP Telephony Workshop, New York (March 2001)
5. Calyam, P., Mandrawa, W., Sridharan, M., Khan, A., Schopis, P.: H.323 Beacon: An H.323 application related end-to-end performance troubleshooting tool. In: Proceedings of NeTS 2004, Portland, OR (October 2004)
6. G.107, I.T.R.: The Emodel, a computational model for use in transmission planning. Recommendations of the ITU, Telecommunications Sector (1998)
7. Wang, Y., Claypool, M., Zuo, Z.: An empirical study of RealVideo performance across the Internet. In: Proceedings of IMW 2001, San Francisco, CA (November 2001)
8. Loguinov, D., Radha, H.: Measurement study of low-bitrate Internet video streaming. In: Proceedings of IMW 2001, San Francisco, CA (November 2001)
9. Nichols, J., Claypool, M., Kinicki, R., Li, M.: Measurement of the congestion responsiveness of Windows streaming media. In: Proceedings of NOSSDAV, Kinsdale, Ireland (June 2004)
10. Csizmar Dalal, A., Musicant, D.R., Olson, J., McMenamy, B., Benzaid, S., Kazez, B., Bolan, E.: Predicting user-perceived quality ratings from streaming media data. In: Proceedings of ICC 2007, Glasgow, Scotland (June 2007)
11. Csizmar Dalal, A., Olson, J.: Feature selection for prediction of user-perceived streaming media quality. In: Proceedings of SPECTS 2007, San Diego, CA (July 2007)

12. Csizmar Dalal, A.: User-perceived quality assessment of streaming media using reduced feature sets. Technical report, Carleton College (April 2009)
13. Hand, D., Mannila, H., Smyth, P.: Principles of Data Mining. MIT Press, Cambridge (2001)
14. Dunham, M.H.: Data Mining: Introductory and Advanced Topics. Prentice Hall, Englewood Cliffs (2002)
15. Tan, P., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley, Reading (2005)
16. Vlachos, M., Hadjieleftheriou, M., Gunopulos, D., Keogh, E.: Indexing multidimensional time-series with support for multiple distance measures. In: KDD 2003, pp. 216–225. ACM Press, New York (2003)
17. Keogh, E., Ratanamahatana, C.: Exact indexing of dynamic time warping. Knowledge and Information Systems 7(3), 358–386 (2005)
18. Sakoe, H., Chiba, S.: Dynamic programming algorithm optimization for spoken word recognition. IEEE Trans. Acoustics Speech Signal Process 26, 43–49 (1978)
19. Carson, M., Santay, D.: NIST Net: a Linux-based network emulation tool. SIGCOMM Comput. Commun. Rev. 33(3), 111–126 (2003)
20. Krosnick, J.A., Fabrigar, L.R.: Designing rating scales for effective measurement in surveys. In: Survey Measurement and Process Quality, pp. 141–165. Wiley-Interscience, Hoboken (1997)
21. Tang, R., William, M., Shaw, J., Vevea, J.L.: Towards the identification of the optimal number of relevance categories. Journal of the American Society for Information Science 50(3), 254–264 (1999)