A Quality of Experience management module

Vlado Menkovski, Georgios Exarchakos,
Antonio Liotta

Electrical Engineering Department
Eindhoven University of Technology
Eindhoven, The Netherlands
{v.menkovski, g.exarchakos, a.liotta}@tue.nl

Antonio Cuadra-Sánchez

Indra
Parque Tecnologico de Boecillo
47151 Valladolid, Spain
acuadra@indra.es

***Abstract*—User centric management of multimedia services necessitates understanding of how different system parameters affect the perceived quality of the service. Today's multimedia delivery systems are highly complex and diverse and span over multiple domains commonly in control of different entities. Furthermore, the end user devices are highly variable in capabilities and characteristics, which directly affect the perceived quality. In addition to these factors the perceived quality in multimedia is subjective by nature, which leads to complexities in accurate estimation with objective methods and costs with subjective estimation. To deal with the necessities of estimation of Quality of Experience (QoE) we have implemented a software module that aggregates information from probes and subjective quality feedback to develop QoE prediction models. These models are then used as part of the QoE management module to estimate the QoE of multimedia delivery streams in real-time. Additionally, the module calculates and suggests QoE remedies (improvements) for multimedia streams that underperform. The remedies are in the form of a single or multiple target parameter values that need to be reached in order for the stream to reach the desired QoE. The functionalities of this module enable an implementation of a user-centric management loop for multimedia services.**

Keywords-Quality of Experience, QoE, Machine Learning, Subjective Testing, User-centric services

I. Introduction

Management of multimedia streaming services incorporates optimizing the service performance and balancing the trade-off between the system resources and the delivered quality. To efficiently manage the services we need to understand how the different system resources and parameters affect the delivered quality.

First of all, perception of quality in multimedia content is subjective and highly varies on the mode of viewing such as handheld, desktop, large screen displays, on the characteristics of the display devices, screen size, resolution and color depth, to name a few. Next, the performance of multimedia services particularly streaming services also depends on factors prior to processing on the user terminal. The transport and encoding factors heavily influence the perceived quality or Quality of Experience (QoE) as well.

These factors, due to the high variability and diversity of the transport and communication systems are many and often outside of the control of the service provider. In light of this diversity and lack of performance guarantees, service providers tend to deploy management technique that uses 'standard' encoding parameters, regardless of the type of content or the display device used with the service. This technique does not deliver standard level of quality to the viewers, but a fixed burden on the system's resources. The lack of understanding of perceived quality from the service may result in suboptimal management decisions for the multimedia services.

User-centric management of multimedia services focuses on the quality as perceived by users rather than as delivered to them. In the former case the aim is user satisfaction while in the latter is the cancellation of any impairments during transmission. In certain environments, end-users may not be able to detect certain impairments, which do not affect their perceived quality. However, delivering more resources to those users is in fact, without reward. QoE-based multimedia streaming management is a more pragmatic approach and increases the capacity of the network as it contributes to a more careful resource allocation. That approach was initially introduced in [1] and consists of three phases: a) QoE estimation, b) design of potential recovery plans (remedies) and finally c) application of the most appropriate plan. As the latter is an operator specific step, the current work provides a method for predicting QoE with limited subjective data and an automated way for detecting appropriate recovery plans. In this paper we are additionally present specific software design and implementation details for the QoE estimation and remedies platform.

The basic idea behind is the execution of subjective studies and simultaneous network probing so that the correlation between network conditions and perceived quality is possible. This can provide enough information to teach a learning algorithm with the conditions that negatively affect QoE. Therefore, any detected condition that falls within that set of learning samples can be classified to the corresponding QoE level. Even though this process results in QoE estimation, the information collected have an even higher value, if combined. For instance, there is a clear view which conditions allow better QoE. A comparison of these conditions can indicate suitable actions and resource management decisions to move from one QoE level to another. These conditions of target parameter values are the QoE remedies. The application module outputs them as a list or group of lists of one or more parameter values. Each group represents one possible remedy. Based on some defined cost functions the utility of each remedy can be calculated and based on that the most useful can be implemented. However, this is highly implementation specific so this final step can only be implemented within the system of the provider itself.

II. QoE Estimation

QoE is by definition what the end-user experiences while using a service [2]. This characterization is in one form or another is what most agrees upon. However, this does not mean that there is such an agreement on the means of measuring the QoE. There is quite a variety of methods that focus on measurement of QoE; they vary in accuracy and complexity [3]. Most common methods use objective techniques to measure the signal distortion whether in the encoding or in the transport stage. The International Telecommunication Union (ITU) has developed standardization document [4] for the various QoE models. The ITU classifies models as parametric, bit-stream, media layer and hybrid model. This classification characterizes the models focus. The parametric models look at network statistics and protocol information through network monitoring. They measure the signal distortions based on transport error statistics and network performance. The bit-stream models derive the quality via analysing content characteristics collected from the coded bit-stream information. The media layer models focus on the media signal and uses knowledge of the Human Visual System (HVS) to predict the subjective quality of video. This can be computationally expensive, but more over none of these methods analyse the QoE from a holistic perspective. Each model has a subset of aspects of QoE and does not take into account all the factors that affect it. Discussing the QoE estimation method efficiency is difficult because there is lack of a standard for comparing between each other [3].

However, even without being able to compare them directly we can understand the drawbacks of the objective approaches. The methods that only look at the fidelity of the audio and video neglect the effect that the type of content has on the perceived quality, as well as how the content is perceived by the HVS.

Typical examples of measurement of signal distortion by pixel to pixel comparison are the Peak Signal to Noise Ratio (PSNR) and Mean Squared Error (MSE) methods. The drawback of these methods is that they compare the signals without any understanding of the HVS [5]. A simple shift in the image will decrease the PSNR value significantly even though this will not be perceived as loss of quality by the viewers.

Modeling the effects that the transport has on the delivered quality would mean looking at the Quality of Service (QoS) parameters. This approach is not very efficient and yields weaker results [6]. The authors of [6] propose looking at the problem in three layers. The bottom layer being the network layer produces the QoS parameters or more precisely the Network QoS (NQoS) parameters. The layer above presents the application layer which is concentrated on parameters like resolution, frame rate, color, codec type, and so on. These parameters are referred to as Application QoS (AQoS). The third or the top layer is the perception layer which is driven by the human perception of the multimedia content and is concentrated on spatial and temporal perception and acoustic bandpass [6]. The QoE, which is measured on the top layer, is a function of both AQoS and NQoS (1).

$$QoE = f\left(AQoS,\ NQoS\right) \tag{1}$$

The proposed framework in [6] discusses that arbitrating all of the QoS parameters together is significantly more effective in maximizing the QoE than looking at each of them individually.

Due to the subjectiveness of QoE, the most accurate way to measure it is by executing subjective test. The subjective studies are of significant importance because they can accurately convey the satisfaction of the viewers with the service. This is why subjective tests are commonly used for comparing the capabilities of different QoE estimation methods. Subjective testing usually entails execution of tests in tightly controlled environment with carefully selected group of subjects that statistically represent the population, which is using the service. Guidelines for the execution of different subjective studies are provided by the ITU [7].

The drawbacks of subjective studies are obvious from their description. They require significant effort and resources to be put into their design and execution. In [8] and [9], the authors present a method that only relies on initial limited subjective tests. From the results of these tests, statistical models are build that can predict the QoE on unseen cases. The subjective tests executed in this work are based on the method of limits [10]. The viewers are presented with video in descending or ascending quality. The viewer detects the point where the perceived quality changes from acceptable to unacceptable in the descending series (or vice versa in the ascending). From these results the authors using discriminate analysis [11] have developed models for predicting the quality on unseen cases. This approach is suitable for minimizing the need for cumbersome subjective studies while providing for estimation based on the user's subjective feedback. However, the accuracy of the prediction models is limited due to the statistical method used to build the prediction models. The work is further extended in [12], where Machine Learning methods are used to build prediction models for QoE. These prediction models both Decision Trees (DT) [13] and Support Vector Machines (SVM) [14] outperform the discriminate analysis approach.

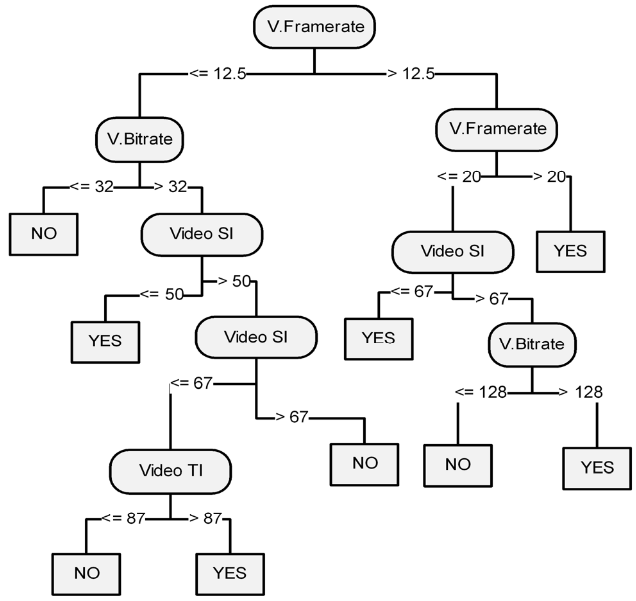

Figure 1. Decision Tree prediction model

In this work, the algorithm that performs best is C4.5 [15]. This is a DT induction algorithm, which builds a DT model from the training data. This DT model (Figure 1) consists of nodes and leaves. The nodes are associated with splitting rules, based on a single attribute. The leaves of the DT are associated with class values, so that, all the datapoints that fall on a particular leaf are classified with the associated class.

Based on this subjective data, the models classifies as QoE acceptable is "Yes" or "No". Using this model unseen cases with different values of the attributes can be now classified as QoE acceptable or not acceptable.

The models in [12] perform with accuracy of above 90% estimated using the cross-validation technique [16].

III. Improving QoE

Algorithms accurate QoE prediction models can be built using ML, having sufficient subjective data. Estimating the QoE is a crucial step in QoE-aware network management. However for a full implementation of the management loop we need to be able to maintain target QoE. Maintaining a target QoE involves determining the desired conditions that need to be achieved. In this section we are introducing a geometric technique that based on the QoE prediction model estimates the minimum needed changes in the measured stream parameters to improve the QoE.

This technique is enabled by the DT prediction models we use for estimating the QoE. One of the strengths of DT compared to other ML prediction models is their intelligibility. A DT in a way represents a set of rules stacked in a hierarchical way. Simple decision trees commonly define just a few rules that are deduced from the data and used for classification, but when the number of rules grows the size of the DT also grows, and with that, it loses its intelligibility. It is also possible to represent a DT model in the geometric space, defined by the dataset parameters. Consider each of the dataset parameters as a dimension in a hyperspace. Each of the datapoints form the dataset can be represented as a point in this hyperspace. The DT is represented by hyper regions formed by the leaves of the DT (Figure 2). Each node in the DT represents a split or a hyperplane that splits the hyperspace, until we reach a leaf, which carves out a hyper region. These hyper regions (as well as the leaves in the DT) are associated with a class label membership. So every datapoint or point in the hyperspace belongs to one of the regions of that the DT defined in the hyperspace, and as such is classified with the corresponding class label. In our particular case the hyper regions are associated class labels that are the QoE estimates.

In order to automate the QoE remedy estimation approach, we implemented an algorithm (Figure 3) that represents the DT in the hyperspace as follows:

This algorithm implements the DT representation in the dataset's hyperspace by generating a set of hyper regions that represent the tree leaves. Each hyper region contains a set of split rules that define the hyper-surface, which carves out the hyper region. The split rules are either representing an inequality of the type *Parameter1 >= Value1* or of the type *Parameter1 = Value1* depending on whether *Parameter1* is continual or categorical. If the leaf is on the left side of a continual *Parameter1* split then the split inequality will be 'more than or equal to', if it is on the right side the split inequality will be 'less than'.

Having a list of *HyperRegion*-s we can easily determine where each datapoint from the dataset belongs to, by testing the datapoint on the split rules of each hyper region. The hyper region is associated with the same class label as the leaf it represents, so all datapoints that belong to that region are classified as such.

In order to improve the QoE estimation of a particular stream, we need to look at the datapoint that was generated by the monitoring system for that stream. If the datapoint is classified with a QoE value that is not satisfactory, we look at the distance to a set of hyper regions $\overline{\Phi}$ that are associated with a satisfactory QoE value. The distance to each of the desired regions is the difference in parameter values that are needed in order to move the datapoint to the desired regions.

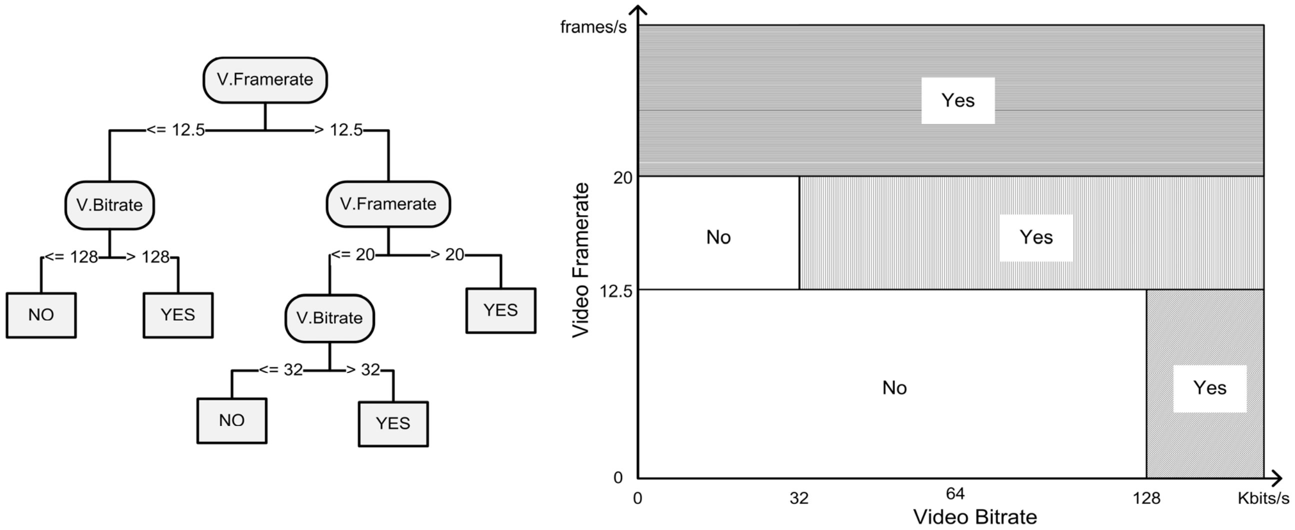

Figure 2. Simple decision tree in 2D space

The output of the algorithm is a set of distance vectors, which define the parameters that need to be changed and their change values.

To illustrate the matter better we can take an example from the laptop dataset from [12]. The prediction model built from this dataset is given in

Figure 1. If we look at the datapoint given in Table 1 we can see that this datapoint will be classified by the model as QoE = No ('Not Acceptable'). Since the V. Framerate is less than 12.5 and the V.Bitrate is less than 32 the datapoint reaches a leaf with 'Not Acceptable' class associated with it.

Now, what is the best way to improve the QoE of this stream?

```
Start from the root node and call a recursive method FindLeaves
  FindLeaves:
1)  If the node has children
    a)  Call FindLeaves on each child
    b)  Add the SplitRule on each of the Hyper Regions ( $\overline{\Phi}$ ) that are returned
        i)   If the leaf split is categorical add a Split Rule: Attribute = 'value'
        ii)  If the leaf split is continual add on the leaves from the left side SplitRule: Attribute < value, and on the leaves from the right side Attribute > value
    c)  Return the set of Hyper Regions ( $\overline{\Phi}$ )
2)  Else, you are in a leaf
    a)  Create an Hyper Region object
        i)   Assign the class of the leaf to the $\Phi$
        ii)  Return $\Phi$
```

Figure 3. DT to Hyper Region algorithm

First of all there are parameters that characterize the type of the content such as the Video SI and the Video TI and cannot be changed. In this dataset structure we are looking into increasing the V.Bitrate and V.Framerate. If we increase the V.Bitrate for this particular datapoint by one step to 64kbits/s we can see that the datapoint goes now down the decision tree to one of the bottom leaves, but it is still classified as QoE Acceptable = No. On another hand if we increase the V.Framerate to 15f/s we can see that the datapoint is classified as QoE Acceptable = Yes without adding more bandwidth.

TABLE I. EXAMPLE DATAPOINT

Video SI	Video TI	V. Bitrate	V. Framerate
67	70	32	10

We can deduce a rule from the model that a video with these characteristics needs to have higher V.Framerate for it to be perceived with high quality. However, this rule is not easily evident from only looking at the model. We can also imagine a system with large number of attributes that we can change where tuning this attributes the right way becomes an increasing problem. Further down this line of reasoning, if we want to make a system-wise improvement that will increase the QoE of most streams we cannot easily derive which parameters are best to be increased and by how much.

In the case of the example datapoint the algorithm returns the two possible paths:

- Increasing the Framerate to above 12.5f/s
- Increasing the V. Bitrate to above 32kbits/s and the Video TI to above 87

Since we know that increasing the Video TI is not an option, because this is defining the type of content we can see, then the only option is to increase the frame rate. In a general case, there can be many different paths to a hyper region with the desired class.

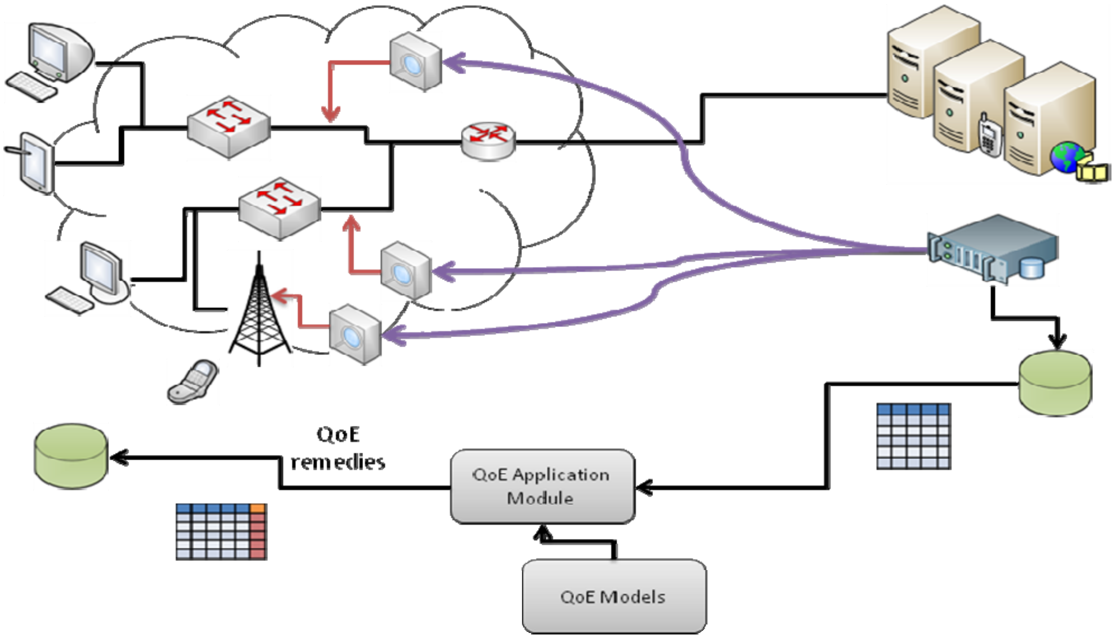

Figure 4. QoE estimation and remedies architecture

To automate the process we can assign cost functions to the change of the attribute values and automatically calculate the cheapest way to reach the desired QoE. In this manner attributes that are not changeable, such as the Video TI, can have infinite value of the cost function.

Given a datapoint and a target label the algorithm produces a set of change vectors. Each of the change vectors applied to the datapoint moves the datapoint to a hyper-region classified with the target label. In other words, each change vector is one possible fix for the datapoint.

$$\overline{\Phi} = FindLeaves(DT, QoE) \quad (2)$$

$$\overline{\Delta\varphi_i} = Distance(\Phi_i, \overline{d}) \quad (3)$$

$$\Delta\varphi_{optimum} = \min_i \left(Cost(\Delta\varphi_i \right) \quad (4)$$

In (2), $\overline{\Phi}$ is a set of regions with a targeted QoE value. The distance function in (3) calculates the vector of distances for each attribute to the target region in $\overline{\Delta\varphi}$. The optimal distance vector is the one with minimal cost (4) for the given input datapoint $\overline{d}$. The *Cost* function in (4) is dependent on the application. Each system has explicit and implicit costs associated with changes of specific parameters.

IV. SYSTEM ARCHITECTURE

The QoE estimation and remedies module is a software application that inputs monitoring data from the monitoring system in a predefined manner and outputs QoE prediction and remedies (Fig. 4). The monitoring system collects statistics of the multimedia service performance and records systems parameter values. These values are collected in a monitoring database. When the application model detects new monitoring data in the database it loads these records and executes a QoE prediction. The estimated value of the QoE is augmented to the original monitoring data and submitted in the database. Then the estimation is compared to the target QoE value. If the particular measured stream does not meet the target QoE a remedy is calculated. The remedies are a group of target parameter values that if reached will improve the stream QoE up to the target value. There could be more than one parameter value that needs to be changed in a single suggested remedy. More over the application will also suggest other possible groups of parameter values as alternative remedies. If costs are associated with the parameter change then the application can calculate the most effective way of improving the QoE by suggesting the cheapest remedy. However, this is highly system specific and needs to be defined by the service provider them self. One example of such a situation could be choosing between improving the QoE by increasing the bit-rate or lowering the packet loss. Increasing the bit-rate might be possible up to some level, lowering the packet loss may be a matter of using different transport protocol or it might not be possible in the given latency constraints.

Due to the complexities associated with defining the costs and utility of the parameter changes the QoE application module is just an enabling component of a QoE management platform. A significant integration effort needs to be implemented in order for a full QoE management platform to be implemented.

V. Software Implementation

The QoE module software is implemented as an extension of the Weka [17] machine learning platform. The Weka platform is provides for a flexible way to build and use ML prediction models. It can load the data from several existing database implementations of from a few text file formats. It has a clearly defined API (application programming interface), which allows for integration with other software components. It provides for extending the library for use adding more functionality and a wide variety of ML algorithm implementations.

The application consists of a main loop which instantiates data listeners. The listeners can be passively waiting for a data file to appear in a directory or actively pooling a database for new data. The main application prior to instantiating the listeners loads the instances of the prediction models. The prediction models are built using the Weka explorer user interface, and then serialized to a local directory.

The listeners now are able to query the models for prediction when new data arrives. They first parse and adapt the data accordingly then call the models to estimate the QoE. This value is recorded and then saved in an extra column in the monitoring database.

If the QoE estimation does not meet the minimum required level the remedy algorithm is executed to create the remedy suggestions.

The remedy algorithm starts by creating a list of HyperRegion (Fig. 5) objects from the give prediction model.

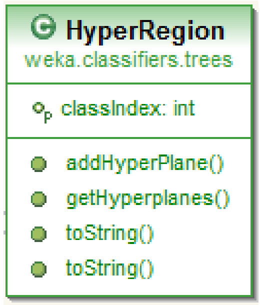

Figure 5. HyperRegion class diagram

For each leaf in the DT the algorithm outputs a HyperRegion object. The object contains a number of hyper planes that define the region in the data space. Each hyper plane corresponds to each split in the decision tree until the leaf is reached. Each of the HyperRegions has a classIndex that corresponds to the QoE level associated for the hyper region and DT leaf. For the implantation of the remedy algorithms we extended the J48 existing code to include a get distance method (Fig. 6).

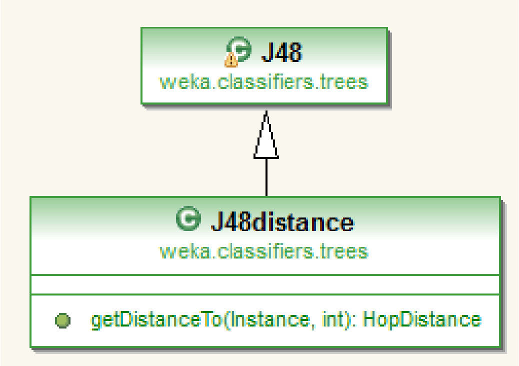

Figure 6. J48 Class diagram

This method is necessary to calculate the distance of a point to a particular hyper region and returns HopDistance object. The remedies algorithm calls the getDistanceTo() method from the model for the particular datapoint and each HyperRegion with QoE value at and above the target level.

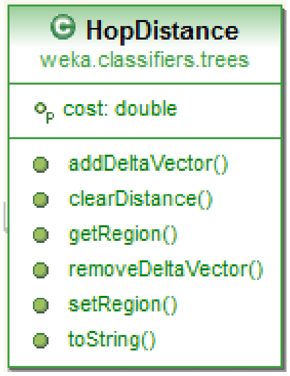

Figure 7. HopDistance class diagram

Each of the HopDistance (Fig. 7) objects contains a list of parameter values distances from the current to the target that need to be reached so that the data point can be moved into a HyperRegion with a satisfactory QoE. These HopDistance-s are then adapted into remedies and outputted from the module as suggestions. If the parameter value change costs are defined, then the algorithm can calculate the costs for each HopDistance or remedy and select the most cost effective one for output. At this point the module can be coupled with a management implementer who can execute action that will implement the given remedies and automate the full QoE management process.

VI. Conclusion

In this work, we present a user-centric approach and a software module implementation for management of streaming multimedia services. This approach is based on developing DT models with ML tools that can estimate the QoE of the streaming service. Furthermore, we have developed a geometric approach that represents the DT model in the dataset feature space, which derives a way for estimating the changes needed for improving the quality of specific streams. Effectively we have presented a functional description of a method that can estimate the QoE and find the possible remedies for improving it, if it is not satisfactory. We have analyzed the method through a worked example with data and models from a subjective study of mobile multimedia streaming. These models show that ML tools can build accurate DT prediction models, which can be used to estimate the possible remedies to reach satisfactory QoE. We present a software module implementation that integrates with a streaming service monitoring system and augments it with estimation for the QoE and its remedies based on the subjective feedback of the customers.

This ongoing research will continue towards the quantitative besides qualitative assessment of the method. There is an undergoing plan to deploy the system on an existing communication network in collaboration with network providers. These experiments are expected to give enough evidence about the financial benefits for a provider in deploying our system.

Acknowledgment

The work included in this article has been supported by Telefonica R&D and Indra (Spain). The authors thank María del Mar Cutanda, head of division at Indra, for providing guidance and feedback.

References

[1] V. Menkovski, G. Exarchakos, A. Liotta, and A. Sanchez, "Estimations and Remedies for Quality of Experience in Multimedia Streaming," in *Advances in Human-Oriented and Personalized Mechanisms, Technologies and Services (CENTRIC), 2010 Third International Conference on*, pp. 11-15, 2010.

[2] S. Winkler and P. Mohandas, "The Evolution of Video Quality Measurement: From PSNR to Hybrid Metrics," *Broadcasting, IEEE Transactions on*, vol. 54, no. 3, pp. 660-668, 2008.

[3] S. Winkler, "Video Quality Measurement Standards - Current Status and Trends," in *Proceedings of ICICS 2009*, 2009.

[4] A. Takahashi, D. Hands, and V. Barriac, "Standardization activities in the ITU for a QoE assessment of IPTV," *Communications Magazine, IEEE*, vol. 46, no. 2, pp. 78-84, 2008.

[5] S. Winkler, *Video Quality and Beyond*. Symmetricom, 2007.

[6] M. Siller and J. Woods, "QoS arbitration for improving the QoE in multimedia transmission," in *Visual Information Engineering, 2003. VIE 2003. International Conference on*, pp. 238-241, 2003.

[7] R. I. ITU-T, "910," *Subjective video quality assessment methods for multimedia applications*, 1999.

[8] F. Agboma and A. Liotta, "Addressing user expectations in mobile content delivery," *Mobile Information Systems*, vol. 3, no. 3, pp. 153-164, Jan. 2007.

[9] F. Agboma and A. Liotta, "QoE-aware QoS management," in *Proceedings of the 6th International Conference on Advances in Mobile Computing and Multimedia*, pp. 111-116, 2008.

[10] G. T. Fechner, E. G. Boring, H. E. Adler, and D. H. Howes, *Elements of psychophysics / Translated by Helmut E. Adler ; Edited by David H. Howes [and] Edwin G. Boring ; with an introd. by Edwin G. Boring*. New York :: Holt, Rinehart and Winston, 1966.

[11] W. R. Klecka, *Discriminant analysis*. SAGE, 1980.

[12] V. Menkovski, A. Oredope, A. Liotta, and A. Cuadra Sánchez, "Predicting Quality of Experience in Multimedia Streaming," in *Proceedings of the 7th International Conference on Advances in Mobile Computing and Multimedia*, pp. 52-59, 2009.

[13] J. Quinlan, "Induction of Decision Trees," *Machine Learning*, vol. 1, no. 1, pp. 81-106, Mar. 1986.

[14] A. J. Smola and B. Schölkopf, "A tutorial on support vector regression," *Statistics and Computing*, vol. 14, no. 3, pp. 199–222, 2004.

[15] J. R. Quinlan, *C4.5*. Morgan Kaufmann, 2003.

[16] R. Kohavi, "A study of cross-validation and bootstrap for accuracy estimation and model selection," in *International Joint Conference on Artificial Intelligence*, vol. 14, pp. 1137-1145, 1995.

[17] I. H. Witten and E. Frank, *Data Mining: Practical machine learning tools and techniques*. Morgan Kaufmann Pub, 2005.