Deep learning model for multimedia Quality of Experience prediction based on network flow packets

Manuel Lopez-Martin, Belen Carro, Jaime Lloret, Santiago Egea, Antonio Sanchez-Esguevillas

Abstract— **Quality of Experience (QoE) is the overall acceptability of an application or service, as perceived subjectively by the end user. In particular for Video Quality (VQ) the QoE is dependent of video transmission parameters. To monitor and control these parameters is critical in modern network management systems, but it would be better to be able to monitor the QoE itself (both in terms of interpretation and accuracy) rather than the parameters on which it depends. In this paper we present the first attempt to predict video QoE based on information directly extracted from the network packets using a deep learning model. The QoE detector is based on a binary classifier (good or bad quality) for seven common classes of anomalies when watching videos (blur, ghost...). Our classifier can detect anomalies at the current time instant and predict them at the next immediate instant. This classifier faces two major challenges: first, a highly unbalanced dataset with a low proportion of samples with video anomaly, and second, a small amount of training data, since it must be obtained from individual viewers under a controlled experimental setup. The proposed classifier is based on a combination of a Convolutional Neural Network (CNN), Recurrent Neural Network (RNN), and Gaussian Process (GP) classifier. Image processing which is the common domain for a CNN has been expanded to QoE detection. Based on a detailed comparison, the proposed model offers better performance metrics than alternative machine learning algorithms, and can be used as a QoE monitoring function in edge computing.**

***Index Terms*—Quality of Experience; Convolutional Neural Network; Deep Learning; Recurrent Neural Network**

M. Lopez, B. Carro, A. Sanchez and S. Egea are with Universidad de Valladolid
J. Lloret is with Universitat Politecnica de Valencia

I. INTRODUCTION

QoE is defined by ITU-T as “the overall acceptability of an application or service, as perceived subjectively by the end user”. The ability to evaluate the QoE in a communication system, and especially in a system involved in video transmission, is critical. One of the main objectives of modern network management systems is to monitor and guarantee end-user Quality of Experience (QoE), hence the importance of an accurate QoE monitoring system. This need is even greater with highly configurable networks (e.g. Software Defined Networks (SDN) and edge computing), where precise and reliable information about end-user quality perception is needed to dynamically reconfigure network resources [1,2].

Edge computing is a way to streamline the flow of traffic between cloud computing services and particular devices (e.g. IoT) and provide real-time local data analysis at the edge of the network, near the source of the data. The capabilities provided by edge computing can be improved if they are leveraged using real-time QoE estimates. This is even more valuable for video transmission networks whose real-time nature makes more important a rapid reaction to QoE degradation [1,2,3,4].

Fig. 1 shows an abstract view of data distribution and processing services for IoT applications. The cloud/central services are responsible for application management and overall coordination. The end devices (IoT devices) produce and consume operational data and commands. Finally, the distribution/edge processing services (middle layer in Fig. 1) are intended to facilitate communication, increase availability and performance and add distributed services closer to the end devices. This middle layer can host services that would otherwise be difficult to deploy at the cloud location (slow and unreliable access) or at the IoT devices (lacking processing capabilities). The QoE predictor proposed here is intended to be deployed as a quality monitoring service at the edge processing layer in Fig. 1.

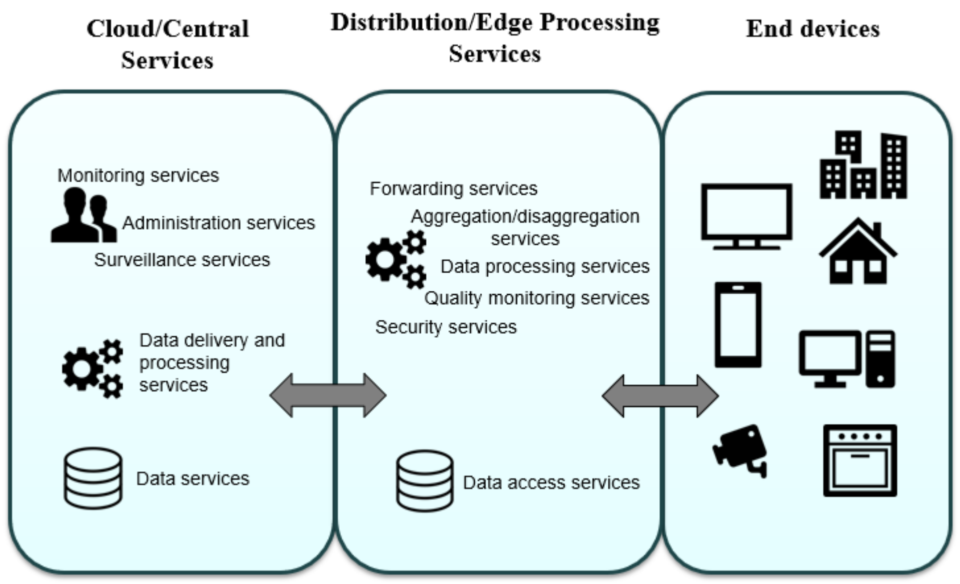


Fig. 1. Abstract view of data distribution and processing services for IoT applications

As the demand for video services increases in parallel with the storage and processing capabilities at the edge layer [3], it is now possible to host highly demanding video processing services in this layer, which allows to offer new network capacities based on automatic and intelligent analysis of video transmissions and QoE-aware network management and video traffic prioritization and scheduling [1,2,3,6]. Hence the importance of more robust and accurate QoE predictors that can make better use of the new processing platforms (e.g. GPUs) at the edge layer [3]. Our proposed predictor is based in a deep learning model that is especially suitable for these new platforms.

At present, the usual way to evaluate QoE is either to carry out experiments with individuals as testers or to calculate it indirectly from Quality of Service (QoS) network parameters (jitter, delay, packet loss,..) [4,5,6]. Another approach, recently being actively explored is applying machine learning (ML) to video QoE estimation. The resulting QoE detector is able to predict QoE directly from information contained in the transmitted videos, the network packets or end-user recorded events (e.g. related web activity). This approach is the one taken on this work in order to build a video QoE detector from network packets information using deep learning models. This is also the most advanced and precise approach [6] that shifts the focus of video quality assessment from QoS (system oriented) to QoE (user oriented).

Building a QoE detector raises important challenges. First, it is difficult to construct a training dataset, since it is obtained in a controlled experiment with several individuals who have to evaluate the quality of the video. This makes it very difficult to acquire large datasets, which are normally needed to train a classification algorithm. Secondly, the training datasets are highly unbalanced, as the number of errors observed in the videos is normally much smaller than the number of non-anomaly events. And third, the subjective judgment of quality, assumed by QoE, necessarily implies noisy results (even using Mean Opinion Score (MOS)), which can make it even more difficult to assess the performance of the algorithms.

Additionally to all former considerations, other important objective in our case was to have a QoE detector which could be integrated into a network management system to monitor network quality (as observed by the end-user), allowing at the same time an efficient network reconfiguration and control (in our case an SDN network). Therefore, QoE detector could identify the QoE score of the video transmitted at the current time-interval, but also be able to anticipate (predict) the quality score for the next time-interval. Since, this prediction can be crucial to anticipate actions on network resources.

Having in mind these challenges, the proposed QoE detector consists of a deep learning classifier that is based on the combination of a Convolutional Neural Network (CNN) and a Recurrent Neural Network (RNN) with a final Gaussian Process (GP) classifier. The classifier implements a binary classification (good or bad quality) for seven usual classes of video anomalies (blur, ghost, columns, chrominance, blockness, color bleeding and black pixel [6]) that can happen when watching the videos.

In order to design the final model, we have tried several alternative architectures and different ML models. We present a complete analysis of the results obtained from these alternative models. The impact of several algorithms' hyper-parameters and design decisions has also been analyzed. Similarly, the process for generating and transforming the training data is presented in detail.

QoE detector utilizes a training dataset created specifically for this work. The dataset was obtained from a controlled experiment in which several individual viewers evaluated video transmissions in a time interval of 1-second and under different network configurations.

The main contributions of this work are: (1) First application of deep learning models to video QoE prediction. (2) Prediction based on network packet information. (3) Network flows treated as pseudo-images that allow applying a CNN. (4) Excellent prediction performance for not extremely unbalanced labels with a small dataset.

The structure of the paper is the following: Related work is presented in Section II. The work performed is described in Section III. The results are discussed in Section IV and finally, Section V provides discussion and conclusions.

II. Related work

There is no similar work in the literature presenting a deep learning solution to video QoE assessment based on information contained in network packets and trained with end-user QoE evaluations in a controlled experiment.

Nevertheless, there is a solid work done on automatic Video Quality Assessment (VQA) based on the identification and processing of parameters extracted from the video. In [6] a thorough review of QoE modeling and methodologies is presented. Authors in [4,5] propose an analytical expression for video QoE calculation based on several parameters: jitter, delay, bandwidth, loss packets and zapping time for IPTV video transmissions.

There is also relevant literature on ML models that are applied to features extracted from the videos (or network packets) in order to rank its quality, usually in accordance with quality assessments obtained from end-users. In this line, in [7] they use QoS parameters to predict QoE using a dataset built from subjective end-user scores, and applying machine learning algorithms based on Support Vector Machine (SVM) and Decision Trees.

A survey of ML techniques used to capture the relationship between QoS parameters and QoE scores is provided in [8], where most of the common machine learning algorithms (Linear Discriminant Analysis, Random Forest (RF), SVM, Naïve Bayes, K-Nearest Neighbors) are applied to the automatic identification of QoE from QoS network parameters.

Considering Content Delivery Networks (CDN), [9] gives a review of the reasons why developing an objective method of quality assessment based on video transmission parameters is extremely difficult due to the complex relationships between these parameters, the user's perception and even the nature of the content. Furthermore, the authors propose the application of ML algorithms (Decision Trees, Naïve Bayes and Logistic Regression) to predict the QoE based on transmission parameters (bitrates, latency,..) and end-user engagement attributes (playtime, number of visits,..).

The prediction of streaming video QoE is proposed in [10] applying several regression models such as Ridge and Lasso Regression, and ensemble methods such as Random Forest (RF), Gradient Boosting (GB) and Extra Trees (ET).

None of the above references apply the new deep learning models and they do not provide a short-term QoE prediction based on network packet information. The QoE score generally provided is a single score in contrast to the simultaneous prediction of seven QoE anomalies/errors, which is provided in this paper. Comparison of performance results between these works is not significant, since the datasets used and the areas of application are too different. In this context, the present work has to be considered as an alternative option available in this topic area.

III. work description

This section presents the experimental configuration employed to generate the training data, the necessary data preparation and a description and comparison of the prediction models applied.

A. Experimental setup: data generation

To generate the data that the QoE prediction models will use, it was necessary to establish an experimental setup that would allow identifying the QoE of the video transmissions while recording the associated network flow packets. The resulting data are multivariate time series, in time-steps of 1-second, which contain the network packets transmitted in each time-step plus the presence or absence of seven video transmission errors in that time-step.

The topology of the experimental setup included three components: (1) A video transmission server, which allowed us to vary the characteristics of the video. (2) The clients, where the video streams are visualized by the end user to label them with QoE errors. And, (3) a packet analyzer (Wireshark) that extracts the network parameters on the end user's side. This configuration allows us to vary several network and video features (jitter, delay, bandwidth, packet loss, bitrate ...) and test their impact on network packets and their associated visual effects. We used several network protocols (HTTP, RTP and UDP) to increase the variety of video transmissions.

B. Data preparation

The data generated, as described in the previous section, is further processed to extract aggregate information associated with each time-step, in 1-second intervals. The new features formed by these aggregates are organized into samples, finally forming a time-series of vectors (samples).

To build the training dataset, an ad-hoc application was developed as a feature extractor. The feature extractor identifies packets belonging to a specific multimedia transmission, extracting certain IP header information from the packets, namely the size of the application layer and the inter-arrival time between consecutive packets. Later, these two features are expanded in a collection of 40 statistical attributes that includes means, standard deviations, root mean squares, maximums, minimums and percentiles. In addition, the number of packets transferred in the ingoing and outgoing directions is counted and also included as a feature. All these features are normalized to the range [0-1] with a previous log normalization for features with high values ranges.

Finally, the QoE information provided by the end users in terms of the possible errors observed in each time-step is appended to the collection of attributes as labels. We have evaluated seven QoE errors: columns, blur, ghost, chrominance, blockness, color bleeding and black pixel. The resulting vector time-series are described in Fig. 2.a.

To train with the least possible number of samples, we perform an additional transformation of the data in Figure 2.a to arrange it in small elementary flows used for training (see Figure 2.b).

Figure 2 shows the complete process to obtain and transform the training data.

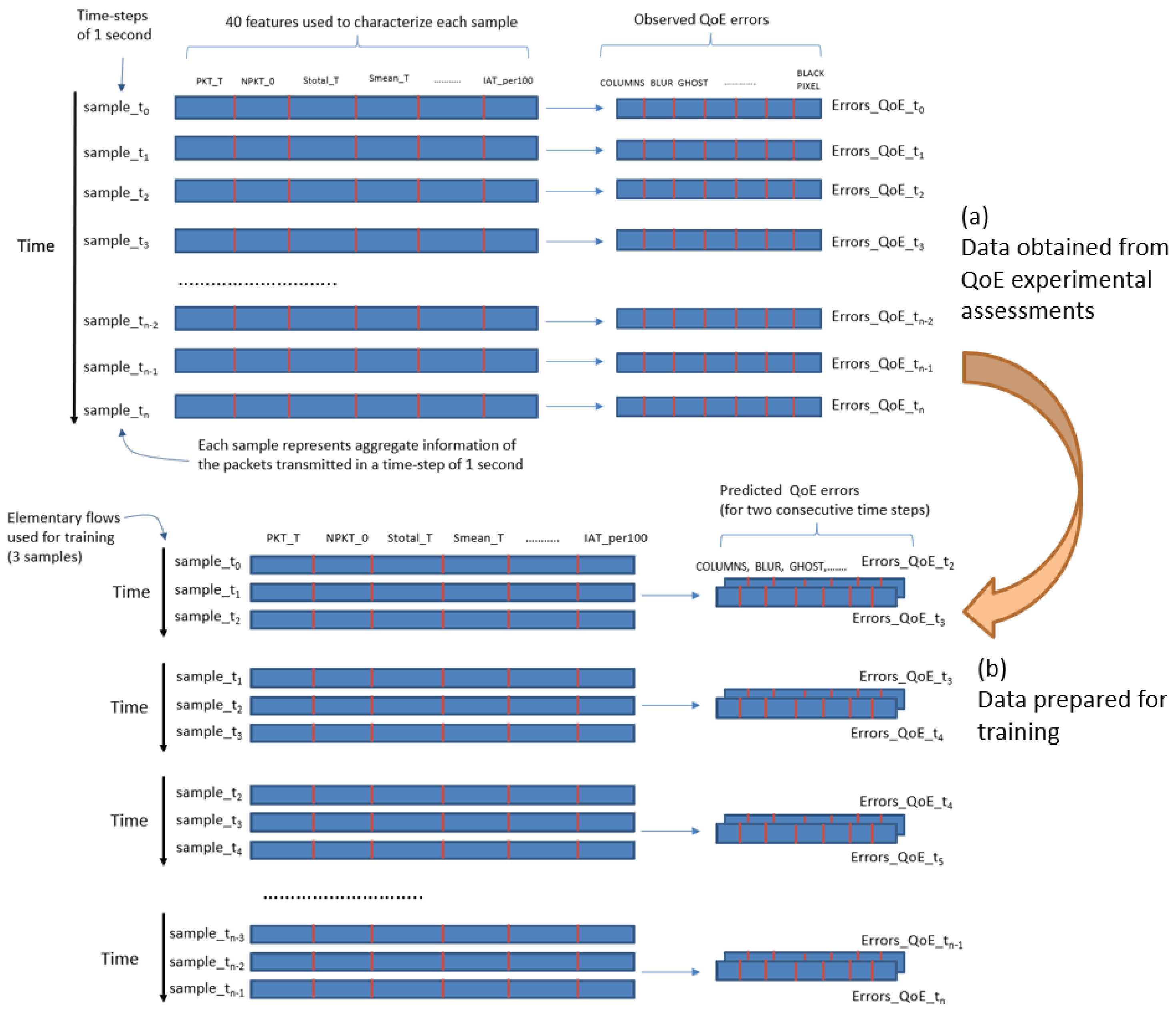


Fig. 2. Training data formed by aggregate data samples (a) and final configuration of the training data, arranged to be used by the models (b).

Fig. 2.b shows the training data ready to be finally used by the models. It can be seen that the data is arranged in small "elementary" flows of 3 samples (corresponding to an elapsed time of 3 seconds). These elementary flows form small vector time-series which are the data entry for training and prediction. The flows are obtained according to a sliding window of width 3 and offset 1 applied to the data in Fig. 2.a. The offset causes the successive flows to have one overlapping sample. For each elementary flow, the models will be trained with QoE errors for the current time-step and the next time-step. In this way, at prediction time, we will be able to detect which errors are occurring in the current time-step and predict errors in the next time interval.

Following the arrangement shown in Fig. 2.b, we finally obtain 2078 elementary flows, which we then divide into 1766 training flows and 312 test flows (15% of total flows). These will be the final data sets that will be used to train and validate all models presented in this paper.

The result vectors for QoE errors are binary vectors (labels), with 1 indicating that an anomaly was present and 0 otherwise.

It is also important to mention the strong imbalance in the distribution of label values. Fig. 3 gives the frequency distribution for QoE label values. It can be observed that the absence of anomalies is much more frequent than the opposite. This fact adds an additional difficulty to the models and has to be considered in the analysis of results.

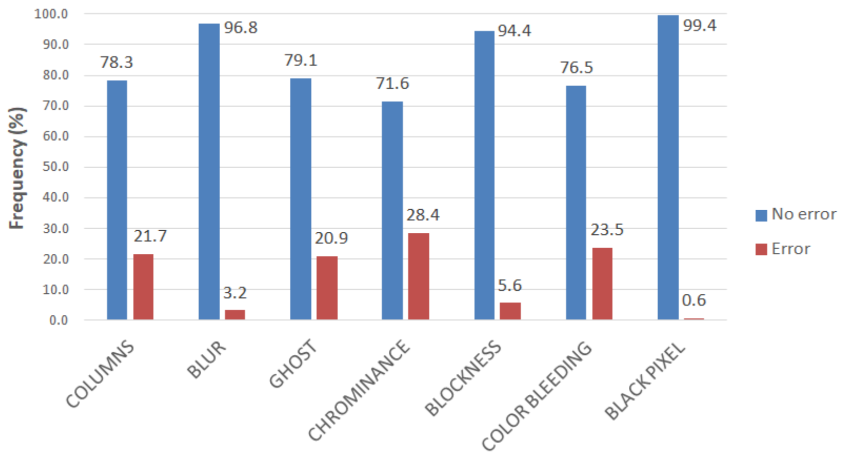


Fig. 3. Frequency distribution for QoE label values.

C. Models for QoE prediction

In this section, we present the different prediction models applied for this work.The prediction models studied can be organized into three groups: (1) classical ML models (Random Forest, Logistic Regression...), (2) deep learning models (CNN and RNN) and (3) a combination of deep learning models plus a Gaussian Process (GP) classifier.

The results obtained by the third group are the best, followed by the second group and the first group of models producing the worst results. Nevertheless, the results difference is not so much significative between the second and third group, but it is rather significative between these two groups and the first group. Results details are provided in section IV.

For the first group, the models studied have been: Linear Support Vector Classifier (SVC), SVC with an RBF kernel, Logistic Regression, Multilayer Perceptron (MLP), RF, Gradient Boosting Method (GBM) and Adaboost. For GBM and Adaboost we used decision trees as elementary classifiers.

Focusing on the second and third groups of algorithms, in order to find the best model for QoE prediction, we have explored several deep learning architectures. A model combining a CNN [11], RNN [12] and GP Classifier [13] has provided the best prediction performance. The implementation of the RNN network has been based on an LSTM [14], which is a special type of RNN.

A CNN [11] is a Neural Network (NN) that applies several convolutional filters to image-formatted data. The filter's weights are learned using Stochastic Gradient Descent (SGD). Each filter applied to the image generates a new scaled-down image that is arranged along a new dimension. The generated multi-dimensional image produces a new data representation that contains invariant properties of the original image at different scales and levels of abstraction.

With a similar strategy to [15], to apply a CNN to a time-series of feature vectors, we assimilate the data to an associated pseudo-image (elementary flow), to which a CNN can be applied. An additional advantage of using deep learning models (mainly the use of the CNN network) is to provide automatic feature engineering (representation) of network flows.

An RNN [12] is an NN intended to treat time-series data, by iterating the NN with the sequential input data and an additional internal state. The internal state is updated at each iteration step. The internal state values are the output of the RNN. The output can be sampled at the end of the iteration process or at each step of the iteration producing as many outputs as iteration steps.

Considering all the architectures for the second and third group of models, we have arrived to three canonical models: Model 1, uses only RNN layers. Model 2, applies a sequence of CNN and RNN layers, and Model 3, adds a GP Classifier to a sequence of CNN and RNN layers. When applying the GP Classifier [13] we have used Model 2 (already trained) as our initial network, then using one of the last layers of this network as the input to the GP Classier. With this configuration, we have trained the GP Classifier by adjusting the lengthscale and variance parameters of the RBF kernel used in this case.

The GP Classifier is based on the so called Laplace approximation [13], which tries to approximate with a Gaussian function a non-Gaussian posterior formed by applying a logistic link function to the output of an intermediate latent function. The resulting squashed outcomes produced by the link function are associated to classification probabilities. The kernel [13] chosen has been a Radial Basis Function (RBF) kernel with two adjustable parameters: lengthscale and variance. To train the GP Classifier consists on tuning these two parameters plus an additional noise variance parameter associated with the likelihood of the model.

Considering the difficulties to apply a multi-label GP classifier, we have applied 14 independent GP binary classifiers (one per label). The GP classifier is a non-parametric algorithm that makes full use of the data available. Interestingly, the increase in performance obtained with the addition of the GP classifier is not so important as to discard the use of the simpler deep learning network, especially given the additional memory and time processing required by adding the GP classifier.

In Fig. 4, the best architectures obtained for Model 2 and 3 are presented (Model 1 being a subset of Model 2). The description of the architecture, given in Fig. 4, follows the notation provided in [15] where more details about the layers and their connections can be found. As a summary, a 2-dimensional image-formatted time-series of samples is presented to the network where a sequence of two CNN layers (first two layers in Fig. 4) extracts new features which are added to a new additional dimension (third dimension). This new dimension is flattened out again to two dimensions before entering into two additional LSTM layers. The LSTM layers are intended to process the time sequence information creating a final embedding of the data into a one-dimensional vector that is delivered to a fully connected final network that eventually generates the expected predictions. The second LSTM layer produces a single output (a vector) while the first one provides as many outputs as iteration steps; these multiple outputs are associated to an extra temporal dimension (a matrix in Fig. 4). The final layers are two fully connected layers.

For all architectures, the training was done with 200 epochs. An epoch is a single pass of all the samples in the training dataset. We have used Rectified Linear Units (ReLU) for the activation functions, except sigmoid activation for the last layer. The loss function employed was Binary Cross-Entropy and the optimization was performed with mini-batch Stochastic Gradient Descent (SGD) with Adam.

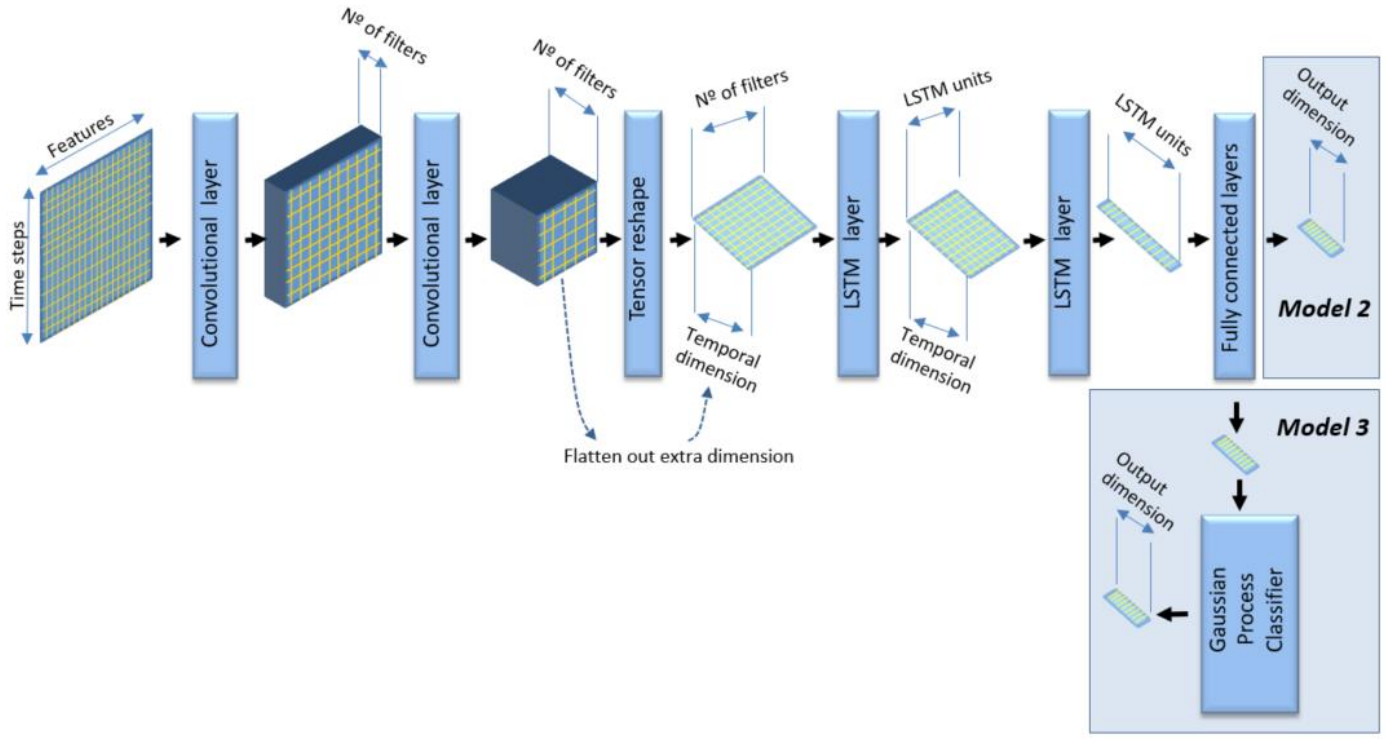


Fig. 4. Architecture for the best deep learning network proposed for this work (Model 2) and alternative combination with GP (Model 3). More details in [15].

IV. RESULTS

This section presents a discussion of results for the different models under evaluation. We focus the analysis on the impact of design decisions, mainly aimed at the architecture of the models and the length of the elementary network flows used for the training.

We provide the following performance metrics: accuracy, precision, recall, F1-score and Area Under the ROC Curve (AUC). F1-score and AUC are particularly suitable considering the highly unbalanced distribution of QoE errors.

For the definition of accuracy, F1, precision, recall and AUC we follow [15]. All performance metrics given in this section are obtained with a test dataset not used at any time during training.

A. Analysis of results

For this problem, we have 14 distinct QoE anomalies (labels) to be detected (7 are anomalies detected at the current time-step and other 7 are anomalies for the next time-step). The anomalies can occur simultaneously, being a multi-label classification problem, considering all the labels, but with a separate error probability calculation for each label. There are two possible ways to give results in this case: aggregated and one-by-one results.

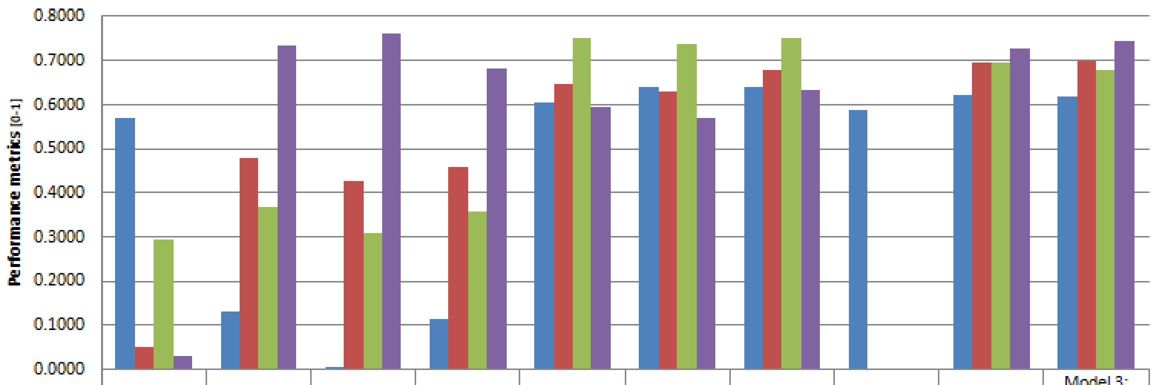
For one-by-one, we focus in a particular class (label) in isolation of the other labels, simplifying the problem to a binary classification task for each particular label (one by one). In the case of aggregated results, we try to give a summary result for all labels. There are different alternatives to perform the aggregation (micro, macro, samples, weighted), varying in the way the averaging process is done.

In this section, we provide the performance metrics obtained for all the models analyzed for this work. The models are described in a previous section (Section III.C). The performance metrics for all the models are presented in Fig. 5. The performance metrics in Fig. 5 are aggregated metrics using a weighted average.

In Fig. 5, we can see that Models 2 and 3 present the best results in terms of F1-score. Of these models, the best accuracy is given in Model 2 (0.6218) and the best F1-score is obtained for Model 3 (0.6987). The slightly better result of Model 3 has to be balanced with the greater needs of memory and processing time for this model. We see that, in general, the metrics obtained are not high, but these metrics are formed by adding the results of 14 labels. In addition, there are three highly unbalanced labels (blur, blockness and black pixel) that significantly worsen the results. This can be seen in Fig. 6, where the performance metrics are calculated for each label separately.

Considering the other models, Random Forest offers fairly good results but the final F1-score (0.6787) is below the best models due to poor results in the recall metric. Focusing on the F1-score, which is our preferred metric for aggregated results (Fig. 5), Model 3 provides a 3% increase over Random Forest. To calculate AUC scores it is necessary to obtain prediction probabilities, which can be cumbersome and not accurate for some models. For example, SVC requires an additional Platt scaling or some other alternative method to retrieve prediction probabilities. Therefore, the AUC score is not considered in Fig. 5 for aggregated results for all models.

It is important to highlight that the worst result in Fig. 5 is for Model 1. This model is formed exclusively by an RNN network and does not include a CNN in its architecture. This is a rather unexpected result as the inclusion of a CNN was not anticipated as something so critical, considering the time-series nature of the predictors.



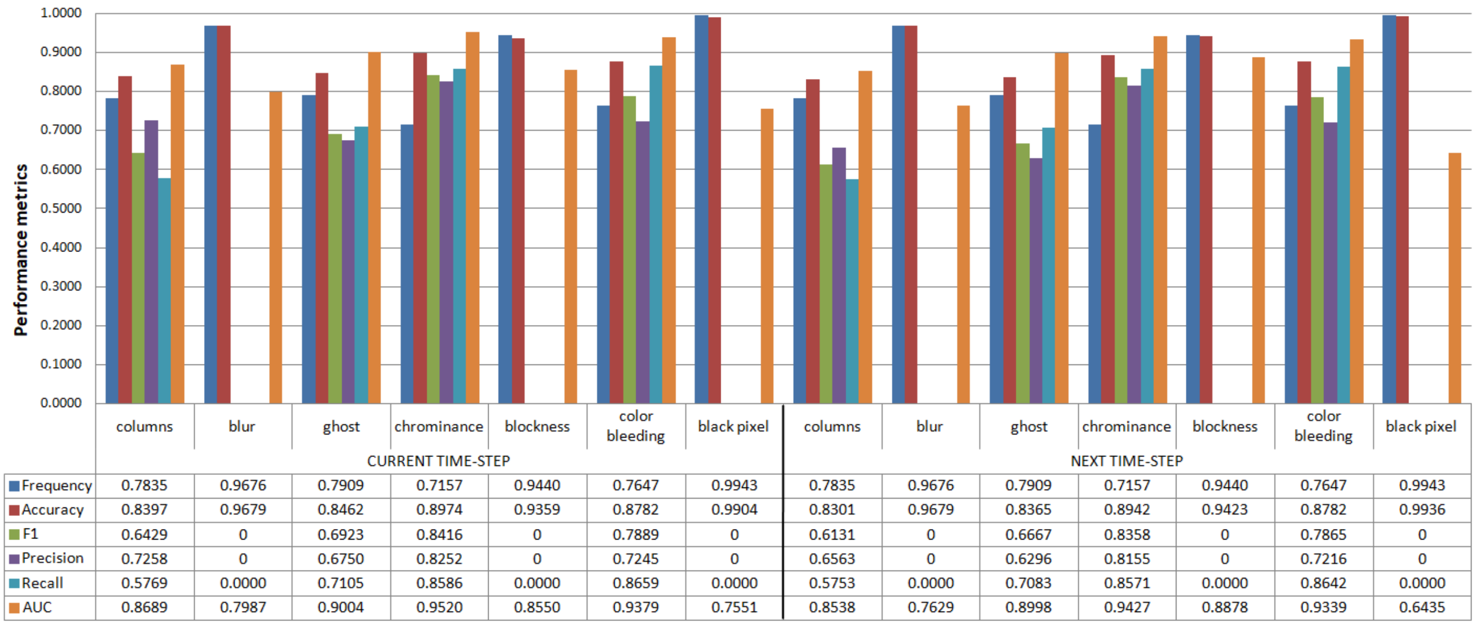
	MLP	Linear SVC	SVC - RBF Kernel	Logistic Regression	GBM	Adaboost	Random Forest	Model 1: RNN	Model 2: CNN+RNN	Model 3: CNN+RNN+GP
🟦 Accuracy	0.5705	0.1314	0.0064	0.1122	0.6058	0.6410	0.6378	0.5865	0.6218	0.6186
🟥 F1	0.0512	0.4799	0.4286	0.4585	0.6476	0.6301	0.6787	0	0.6965	0.6987
🟩 Precision	0.2938	0.3676	0.3098	0.3570	0.7495	0.7356	0.7513	0	0.6949	0.6792
🟪 Recall	0.0305	0.7327	0.7618	0.6814	0.5942	0.5693	0.6330	0.0000	0.7271	0.7424

Fig. 5. Performance metrics (aggregated) for QoE classification for all models

In Fig. 6 is provided the one-by-one results for the classification of the 7 labels for the current and next time-steps. The AUC score and the frequency of the non-error value are given for all the labels, in addition to the metrics: accuracy, F1, precision and recovery. The results are obtained with Model 2.

It is important to note in Fig. 6 that for some labels the results are very good (chrominance and color bleeding), for others are not bad (columns and ghost) and very bad for the rest (blur, blockness and black pixel). For these latter labels, the detector always assigns the most frequent value, thus making the accuracy almost identical to the frequency of the most frequent value (non-error value). This happens for the labels with the most unbalanced values.

The F1 and AUC values are ranked in a similar order (Fig.6). It is interesting to note that the performance metrics get worse at the next time-step vs. the current time-step, as expected, but the reduction in performance is quite small, which is good news as it confirms the implicit initial hypothesis that the prediction of QoE was possible.



	columns	blur	ghost	chrominance	blockness	color bleeding	black pixel	columns	blur	ghost	chrominance	blockness	color bleeding	black pixel
	CURRENT TIME-STEP							NEXT TIME-STEP						
🟦 Frequency	0.7835	0.9676	0.7909	0.7157	0.9440	0.7647	0.9943	0.7835	0.9676	0.7909	0.7157	0.9440	0.7647	0.9943
🟥 Accuracy	0.8397	0.9679	0.8462	0.8974	0.9359	0.8782	0.9904	0.8301	0.9679	0.8365	0.8942	0.9423	0.8782	0.9936
🟩 F1	0.6429	0	0.6923	0.8416	0	0.7889	0	0.6131	0	0.6667	0.8358	0	0.7865	0
🟪 Precision	0.7258	0	0.6750	0.8252	0	0.7245	0	0.6563	0	0.6296	0.8155	0	0.7216	0
🟦 Recall	0.5769	0.0000	0.7105	0.8586	0.0000	0.8659	0.0000	0.5753	0.0000	0.7083	0.8571	0.0000	0.8642	0.0000
🟧 AUC	0.8689	0.7987	0.9004	0.9520	0.8550	0.9379	0.7551	0.8538	0.7629	0.8998	0.9427	0.8878	0.9339	0.6435

Fig. 6. Performance metrics (one-by-one) for QoE classification. Results obtained with Model 2.

In conclusion, the classifier achieves excellent prediction results for the non-extremely unbalanced labels. These results are more significant considering the small number of training samples available and the noisy nature of error detection (subjective component).

B. *Impact of time-series length*

Considering the small amount of data available, the length of the elementary flows used for training is an important parameter to be considered.

Increasing the number of samples used in each elementary flow reduces the number of training flows and imposes a tougher requirement in the amount of information needed to perform prediction.

Interestingly, the best performance metrics are obtained for three samples per elementary flow. The performance does not increase when increasing the number of samples per flow beyond three, implying that prediction is conditioned on the amount of previous information, but information too distant in time is not only less useful but indeed a problem. Similarly, reducing the number of samples below three also decreases the prediction performance.

V. Conclusion

This work shows that it is possible to apply new deep learning models whose origin focused mainly on the areas of video, audio and language processing for the prediction of QoE of transmitted videos. We have extended the work in [15], demonstrating that a CNN can be applied to a time-series of samples (formed by aggregated information from network packets) to predict QoE for video transmission. As far as we know, there is no previous application of a deep learning model to QoE prediction, being this work the first contribution to this area.

The proposed model can be integrated into a network management system to monitor network quality (as observed by the end-user), which is an essential part of a self-adapting network (e.g. SDN, edge computing...). The model is applicable to a real-time environment (in time-steps of 1-second) and is able to predict video QoE for current and near-future video transmissions.

The best proposed model includes a combination of CNN and RNN networks, being the CNN network the most critical piece. We have extended the study with the inclusion of a GP Classifier that, being a non-parametric model, could make full use of the scarce data available. This inclusion provides a slight improvement in the prediction results, but has to be considered in parallel with the increase in memory and processing time required.

In future work, we plan to investigate the application of generative models (e.g. variational autoencoders) to create synthetic data and explore one-shot learning advances.

ACKNOWLEDGMENTS

This work has been funded by the Ministerio de Economía y Competitividad del Gobierno de España and the Fondo de Desarrollo Regional (FEDER) within the project "Inteligencia distribuida para el control y adaptación de redes dinámicas definidas por software, Ref: TIN2014-57991-C3-2-P", and, also by the Ministerio de Economía y Competitividad in the Programa Estatal de Fomento de la Investigación Científica y Técnica de Excelencia, Subprograma Estatal de Generación de Conocimiento with the projects “Distribucion inteligente de servicios multimedia utilizando redes cognitivas adaptativas definidas por software, Ref: TIN2014-57991-C3-1-P" and “Red Cognitiva Definida por Software Para Optimizar y Securizar Tráfico de Internet de las Cosas con Informacion Critica”, Ref TIN2017-84802-C2-1-P.

REFERENCES

[1] J. Wang, J. Pan, and F. Esposito, “Elastic urban video surveillance system using edge computing,” *Proceedings of the Workshop on Smart Internet of Things (SmartIoT '17).* ACM, New York, NY, USA, 2017, Article 7
[2] K. Bilal, A. Erbad, "Edge computing for interactive media and video streaming," *2017 Second International Conference on Fog and Mobile Edge Computing (FMEC),* Valencia, Spain, May 8-11, 2017, pp. 68-73
[3] G. Ananthanarayanan et al., "Real-Time Video Analytics: The Killer App for Edge Computing," *Computer*, vol. 50, no. 10, 2017, pp. 58-67.
[4] J. Lloret et al., “A QoE management system to improve the IPTV network,” *International Journal of Communication Systems*, 24, 2011, pp. 118–138.
[5] M. Garcia et al., "A QoE Management System for Ubiquitous IPTV Devices," *2009 Third International Conference on Mobile Ubiquitous Computing, Systems, Services and Technologies*, Sliema, 2009, pp. 147-152.
[6] Y. Chen, K. Wu and Q. Zhang, "From QoS to QoE: A Tutorial on Video Quality Assessment," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 2, 2015, pp. 1126-1165.
[7] V. Menkovski, G. Exarchakos and A. Liotta, "Machine Learning Approach for Quality of Experience Aware Networks," *2010 International Conference on Intelligent Networking and Collaborative Systems,* Thessaloniki, 2010, pp. 461-466.
[8] S. Aroussi, A. Mellouk, "Survey on machine learning-based QoE-QoS correlation models," *2014 International Conference on Computing, Management and Telecommunications (ComManTel),* Da Nang, 2014, pp. 200-204.
[9] A. Balachandran et al., “Developing a predictive model of quality of experience for internet video,” *Proceedings of the ACM SIGCOMM 2013 conference on SIGCOMM (SIGCOMM '13)*. ACM, New York, NY, USA, 2013, pp. 339-350.
[10] C. G. Bampis, A. C. Bovik, “Learning to Predict Streaming Video QoE: Distortions, Rebuffering and Memory,” *arXiv*:1703.00633 [cs.MM], 2017.
[11] W. Rawat, Z. Wang, “Deep Convolutional Neural Networks for Image Classification: A Comprehensive Review,” *Neural Computation*, vol. 29, no. 9, 2017, pp. 2352–2449.
[12] Z. C. Lipton, J. Berkowitz, and C. Elkan, “A Critical Review of Recurrent Neural Networks for Sequence Learning,” *arXiv*:1506.00019 [cs.LG], 2015.
[13] C. K. I. Williams, D. Barber, "Bayesian classification with Gaussian processes," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 12, 1998, pp. 1342-1351.
[14] S. Hochreiter, J. Schmidhuber, “Long Short-Term Memory”, *Neural Computation*,” vol. 9, no. 8, 1997, pp. 1735–1780.
[15] M. Lopez-Martin et al., "Network Traffic Classifier with Convolutional and Recurrent Neural Networks for Internet of Things," *IEEE Access*, vol. 5, 2017, pp. 18042-18050.

BIOGRAPHIES

MANUEL LOPEZ-MARTIN [M’91, SM’12] (mlopezm@ieee.org) is a research associate and Ph.D. candidate at Universidad de Valladolid (Spain). His research activities include the application of machine learning models to data networks. He received his M.Sc. in Telecommunications Engineering in 1985 from UPM-Madrid and his M.Sc. in Computer Sciences in 2013 from UAM-Madrid. He has worked as data-scientist at Telefonica and has more than 25 years of experience in the development of IT software projects.

BELEN CARRO (belcar@tel.uva.es) received a Ph.D. degree in the field of broadband access networks from the Universidad de Valladolid, Spain, in 2001. She is Professor and Director of the Communications Systems and Networks (SRC) laboratory at Universidad de Valladolid, working as Research Manager in NGN communications and services, VoIP/QoS and machine learning. She has supervised a dozen Ph.D. students and has extensive research publications experience, as author, reviewer and editor.

JAIME LLORET [M’07, SM’10] (jlloret@dcom.upv.es) received his M.Sc. in Physics in 1997, his M.Sc. in Electronic Engineering in 2003 and his Ph.D. in telecommunication engineering in 2006. He is Associate Professor at the UPV and Chair of the Research Institute IGIC. He has authored more than 450 research papers and book chapters. He is EiC of the “Ad Hoc and Sensor Wireless Networks” and "Networks Protocols and Algorithms". He is ACM Senior and IARIA Fellow.

SANTIAGO EGEA (santiago.egea@alumnos.uva.es) received a Telecommunication Engineering Degree from Polytechnic University of Cartagena, Murcia, Spain. Currently, He is a PhD student at University of Valladolid, Valladolid, Spain. He is member of the Communications Systems and Networks (SRC) lab. His research interests include Signal Processing and Machine Learning specifically applied to telecommunication networks.

ANTONIO SANCHEZ-ESGUEVILLAS [M’07, SM’07] (antoniojavier.sanchez@uva.es) received a Ph.D. degree in the field of QoS over IP networks from the University of Valladolid, Spain, in 2004. He has managed innovation at Telefonica, has been Adjunct Professor and Honorary Collaborator at the University of Valladolid, supervising several Ph.D. students. He has coordinated very large international R&D projects, has over 50 international publications and several patents. His current research interests include digital services and machine learning.