Paper

Modeling of Quality of Experience in No-Reference Model

Jakub Nawała, Lucjan Janowski, and Mikołaj Leszczuk

Department of Telecommunications, AGH University of Science and Technology, Krakow, Poland

Abstract—The key objective of no-reference (NR) visual metrics is to predict the end-user experience concerning remotely delivered video content. Rapidly increasing demand for easily accessible, high quality video material makes it crucial for service providers to test the user experience without the need for comparison with reference material. Nevertheless, the QoE measurement is not enough. The information about the source or error is very important as well. Therefore, the described system is based on calculating numerous different NR indicators, which are combined to provide the overall quality score. In this paper, more quality indicators than are used in the QoE calculation are described, since some of them detect specific errors. Such specific errors are difficult to include in a global QoE model but are important from the operation point of view.

Keywords—QoE, QoS, OTT, no-reference, quality assessment.

1. Introduction

Providing not only a high level of traditional Quality of Service (QoS), but also Quality of Experience (QoE) is a real challenge for Internet Service Providers (ISPs), audiovisual service providers, broadcasters and new Over-The-Top (OTT) service providers. Therefore, objective audiovisual data metrics are often carried out to monitor, troubleshoot, analyze and establish patterns of content applications working in real-time or offline scenarios. Since 2000, the work bound with the concept of QoE, in the context of different applications, has gained momentum and achieved business recognition.

Many researchers focus on different ways to assess the quality of vision applications, taking into account additional information used in the evaluation process. Usually, two main approaches (metrics classes) are distinguished. The first approach is called full-reference (FR), and assumes unlimited access to the original (reference) video sequences. FR metrics are usually the most accurate at the expense of higher computational effort. The second class is commonly referred to as a no-reference (NR) approach and is based on the quality assessment without knowledge of the original material. Due to the missing original signal, NR metrics may be less accurate than their FR counterparts are, but tend to provide much better computational efficiency and provide information in the case of missing the source.

In this paper, we present a NR-based metric on a numerous different NR metrics, which predicts a single quality distortion. The metric itself connects the previously developed metrics by a machine-learning algorithm. The single quality distortion metrics follow the idea of key performance indicators (KPIs) [1]. For developing the global metric, a full reference Video Quality Metric (VQM) was used [2].

Most models of quality are based on the measurement of typical artifacts/KPIs, such as blur, blockiness or jerkiness, and produce Mean Opinion Score (MOS) forecasts. Therefore, many of the algorithms generating an expected value of MOS use a blend of blur, blockiness and jerkiness metrics. Weighting between each KPI can be a simple mathematical function. However, if one KPI is not correct, the global result of prediction is completely wrong. Other KPIs – such as exposure, noise, block-loss, freezing, slicing, etc. – are usually not considered in prognosis of the MOS [3].

Although not standardized, NR video quality assessment methods do exist. Zhu *et al.* presented in [4] model based on discrete cosine transform (DCT) and non-linear sequence-level features to subjective scores mapping by the usage of trained multilayer neural network. Experimental results have proven that NR metrics can compete with its FR and reduced reference (RR) counterparts. However, due to its nature, the NR approach used is both distortion specific and data driven, as compared to the more universal FR algorithms. This conclusion is not surprising, considering the fact that the authors focused solely on H.264/AVC compression as a fundamental source of distortions. On the other hand, findings shown in [5] suggest the possibility to introduce a data independent NR solution. Li, Guo and Lu use spatiotemporal 3D DCT to extract features in both space and time. This information is further used to calculate a small set of parameters, which after temporal pooling for the entire sequence, be mapped to subjective scores. Thanks to thorough training and testing on various databases, the authors of [5] verified data independence of their solution. Nonetheless, the best results were obtained for sequences distorted with only a single artifact source, making this solution not globally applicable.

It is worth mentioning that both [4] and [5] use the luminance channel solely. This concept is also applied in presented work due to higher human visual system (HVS) sensitivity for luminance (rather than color) changes.

Another thing to consider about the solution described in this article is the lack of temporal pooling and subjective scores mapping, which makes it difficult to directly compare our work with others. Those missing concepts remain to be implemented and tested in the near future.

The remainder of this paper is structured as follows. A general overview of software structure and quality metrics listing is given in Section 2. Section 3 presents experimental threshold values for metrics, along with a methodology used to obtain them. A detailed description of the operation of the presented software is given in Section 4, which is further divided into Subsections 1 to 5, all of which provide a comprehensive guide to the development process. Integration of quality evaluation software package with the IMCOP system is provided in Section 5. Section 6 concludes the paper.

2. Structure

Aiming to allow easier evaluation and debugging of the software, the authors decided to design it in a modular manner. This basically means that each of the metrics may be easily detached or attached to the whole topology. Utilizing such a strategy makes it possible to comfortably and efficiently modify the package functionality. In this way, the final shape of the application may be precisely carved to fit the desired use-case scenario.

The software consists of 15 visual metrics, which together form KPIs that could be used to model predicted QoE, as seen from the perspective of the end-user. The following set of metrics is formed:

1. Exposure [6],
2. Freezing,
3. Interlacing [7],
4. Spatial activity [8],
5. Temporal activity [8],
6. Letterboxing,
7. Pillarboxing,
8. Blockiness [8],
9. Noise [7],
10. Slicing [1],
11. Block-loss [1],
12. Blur [8], [9],
13. Contrast,
14. Flickering [8],
15. Blackout.

References next to the above-mentioned metrics lead to experimental setups providing concept verification. As an addition, one can refer to the work of Søgaard *et al.* [10], which uses some of those indicators to objectively measure the quality of a video sequence with variable bitrate, utilized to test subjective scores for HTTP Adaptive Streaming technology and its influence on user experience.

It is worth mentioning that all quality indicators presented here were developed either by the authors themselves or by other members of a team, which the authors are part of.

3. Measurement Software Package

As was already mentioned, the presented software package performs a remote NR quality assessment. The main goal accompanying its design and implementation was the idea to create an application that is platform-independent and does not include proprietary software. Consequently, the source code of the program was written entirely in C programming language and none of the metrics utilized any external libraries. This approach resulted in a longer development timeframe but at the same time allowed us to create a versatile, portable and stable measurement system.

3.1. Input and Output Interfaces

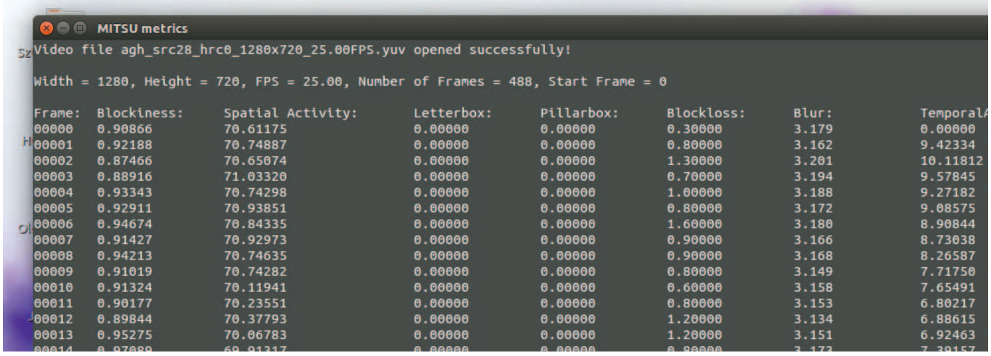
The presented software package operates within the NR model, meaning that the measurement is performed without any knowledge of the original sequence. Therefore, input material must be analyzed in pixel-by-pixel fashion. This in turn imposes the necessity of decompression of the video file or stream, before any computation may be performed. Because the algorithms used operate solely on the luminance channel (Y), YUV420p format is utilized to store the input files for the application. It makes it possible to save memory by omitting part of the information related to colors, further referred to as chrominance channels. Data stored in this manner incorporates complete information about the grayscale representation, but allocates only one value of chrominance channels (U and V) for each 4 pixels of the original material.

An additional advantage of using the previously mentioned format is contiguous alignment of image data, which constitutes a basic optimization strategy. Most hardware platforms perform best when operated on linearly stored information. Reading out sequentially ordered memory blocks yields the lowest possible access times and thus leaves more headroom for the actual computation.

In addition to the uncompressed video sequence, the application also expects the parameters describing width, height and number of frames per second of the tested material. Supplementary input arguments result from the specification of YUV420p format. It does not contain any header for storing detailed information about the included material. In most cases, however, this is not a problem, since data used for processing exists in compressed form, which along with the video material, contains all the information essential for further processing.

The application generates a detailed report concerning each frame of the input material. Alongside frame number, one can also see the result of each previously described metrics. Presentation of the output information is twofold:

- Standard output (stdout) – results get printed in the terminal session used to invoke the software (see Fig. 1),
- Comma-Separated Values (CSV) file – outcome stored in the form suitable for usage in spreadsheets and automated calculation scripts (see Fig. 2).

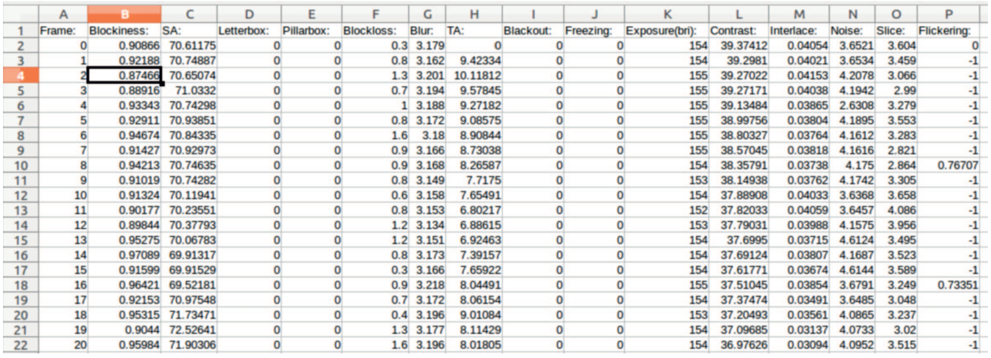


```
Video file agh_src28_hrc0_1280x720_25.00FPS.yuv opened successfully!

Width = 1280, Height = 720, FPS = 25.00, Number of Frames = 488, Start Frame = 0

Frame:  Blockiness:     Spatial Activity:       Letterbox:      Pillarbox:      Blockloss:      Blur:           TemporalA
00000   0.90866         70.61175                0.00000         0.00000         0.30000         3.179           0.00000
00001   0.92188         70.74887                0.00000         0.00000         0.80000         3.162           9.42334
00002   0.87466         70.65074                0.00000         0.00000         1.30000         3.201           10.11812
00003   0.88916         71.03320                0.00000         0.00000         0.70000         3.194           9.57845
00004   0.93343         70.74298                0.00000         0.00000         1.00000         3.188           9.27182
00005   0.92911         70.93851                0.00000         0.00000         0.80000         3.172           9.08575
00006   0.94674         70.84335                0.00000         0.00000         1.60000         3.180           8.90844
00007   0.91427         70.92973                0.00000         0.00000         0.90000         3.166           8.73038
00008   0.94213         70.74635                0.00000         0.00000         0.90000         3.168           8.26587
00009   0.91019         70.74282                0.00000         0.00000         0.80000         3.149           7.71750
00010   0.91324         70.11941                0.00000         0.00000         0.60000         3.158           7.65491
00011   0.90177         70.23551                0.00000         0.00000         0.80000         3.153           6.80217
00012   0.89844         70.37793                0.00000         0.00000         1.20000         3.134           6.88615
00013   0.95275         70.06783                0.00000         0.00000         1.20000         3.151           6.92463
00014   0.97089         69.91317                0.00000         0.00000         0.80000         3.173           7.39157
```

Fig. 1. Exemplary standard output generated by QoE software package.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
1	Frame:	Blockiness:	SA:	Letterbox:	Pillarbox:	Blockloss:	Blur:	TA:	Blackout:	Freezing:	Exposure(bri):	Contrast:	Interlace:	Noise:	Slice:	Flickering:
2	0	0.90866	70.61175	0	0	0.3	3.179	0	0	0	154	39.37412	0.04054	3.6521	3.604	0
3	1	0.92188	70.74887	0	0	0.8	3.162	9.42334	0	0	154	39.2981	0.04021	3.6534	3.459	-1
4	2	0.87466	70.65074	0	0	1.3	3.201	10.11812	0	0	155	39.27022	0.04153	4.2078	3.066	-1
5	3	0.88916	71.0332	0	0	0.7	3.194	9.57845	0	0	155	39.27171	0.04038	4.1942	2.99	-1
6	4	0.93343	70.74298	0	0	1	3.188	9.27182	0	0	155	39.13484	0.03865	2.6308	3.279	-1
7	5	0.92911	70.93851	0	0	0.8	3.172	9.08575	0	0	155	38.99756	0.03804	4.1895	3.553	-1
8	6	0.94674	70.84335	0	0	1.6	3.18	8.90844	0	0	155	38.80327	0.03764	4.1612	3.283	-1
9	7	0.91427	70.92973	0	0	0.9	3.166	8.73038	0	0	155	38.57045	0.03818	4.1616	2.821	-1
10	8	0.94213	70.74635	0	0	0.9	3.168	8.26587	0	0	154	38.35791	0.03738	4.175	2.864	0.76707
11	9	0.91019	70.74282	0	0	0.8	3.149	7.7175	0	0	153	38.14938	0.03762	4.1742	3.305	-1
12	10	0.91324	70.11941	0	0	0.6	3.158	7.65491	0	0	154	37.88908	0.04033	3.6368	3.658	-1
13	11	0.90177	70.23551	0	0	0.8	3.153	6.80217	0	0	152	37.82033	0.04059	3.6457	4.086	-1
14	12	0.89844	70.37793	0	0	1.2	3.134	6.88615	0	0	153	37.79031	0.03988	4.1575	3.956	-1
15	13	0.95275	70.06783	0	0	1.2	3.151	6.92463	0	0	154	37.6995	0.03715	4.6124	3.495	-1
16	14	0.97089	69.91317	0	0	0.8	3.173	7.39157	0	0	154	37.69124	0.03807	4.1687	3.523	-1
17	15	0.91599	69.91529	0	0	0.3	3.166	7.65922	0	0	154	37.61771	0.03674	4.6144	3.589	-1
18	16	0.96421	69.52181	0	0	0.9	3.218	8.04491	0	0	155	37.51045	0.03854	3.6791	3.249	0.73351
19	17	0.92153	70.97548	0	0	0.7	3.172	8.06154	0	0	154	37.37474	0.03491	3.6485	3.048	-1
20	18	0.95315	71.73471	0	0	0.4	3.196	9.01084	0	0	153	37.20493	0.03561	4.0865	3.237	-1
21	19	0.9044	72.52641	0	0	1.3	3.177	8.11429	0	0	154	37.09685	0.03137	4.0733	3.02	-1
22	20	0.95984	71.90306	0	0	1.6	3.196	8.01805	0	0	154	36.97626	0.03094	4.0952	3.515	-1

Fig. 2. Exemplary CSV file generated by QoE software package.

3.2. Planned and Applied Optimization Schemes

The careful reader may notice that operations performed on uncompressed video sequences require large memory bandwidth, as well as high computational power. This kind of restriction becomes especially important when operating in real-time or nearly real-time scenarios. Average computation time for 1920×1080 material oscillates around 119 ms. At this point it is worth mentioning that this test was conducted using a single thread version of the application on the machine featuring a Intel Core i7 CPU 950 3.07 GHz ×8 processor.

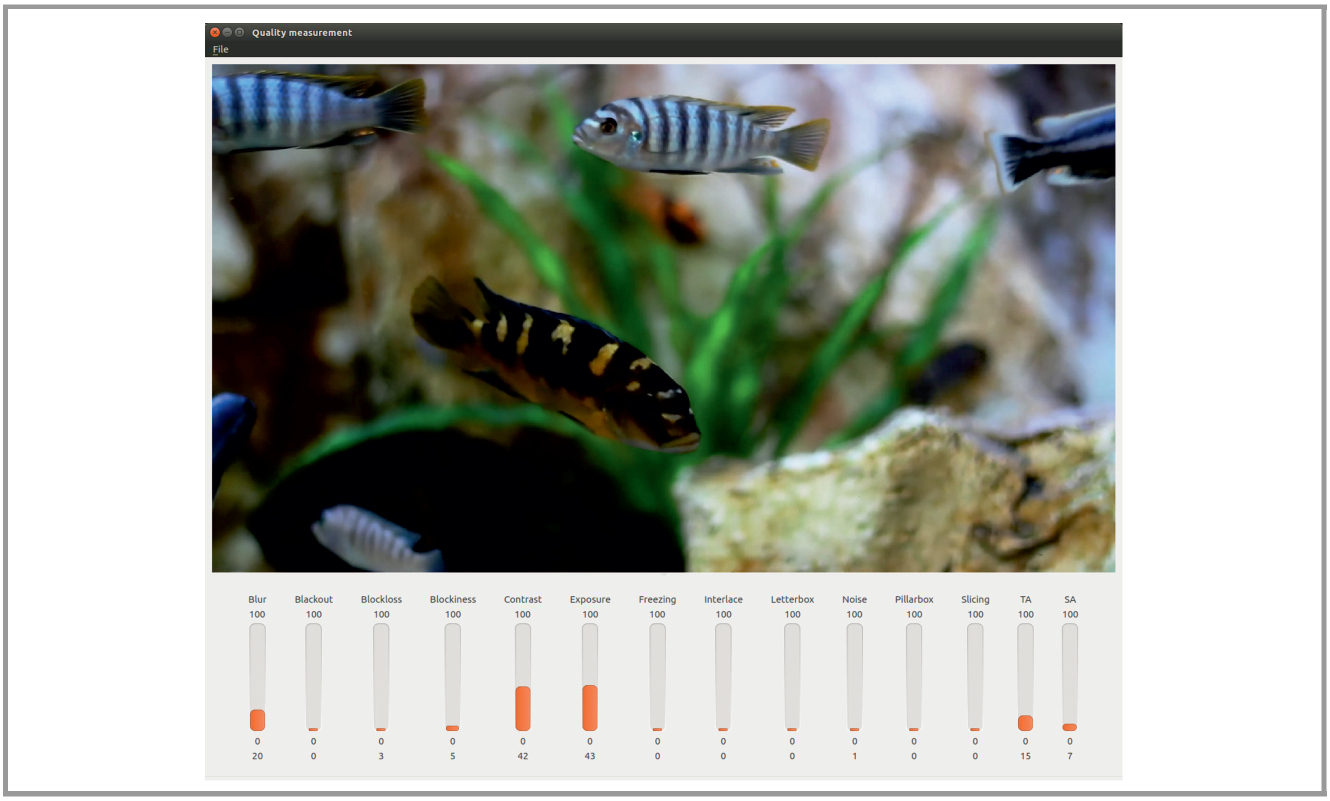
The average processing time indicates the necessity of further optimization if one requires real-time execution of the software. Assuming the video sequence gets refreshed 30 times per second, fetching image data and performing computations must not exceed 33 ms. Should dropping any of the provided indicators prove impossible, another optimization technique would be to utilize a multiprocessor and thus, multithread architecture of contemporary platforms. Performing the test once again – this time employing a multithread version of the application – allowed us to reduce the time needed for calculations to 59 ms. Even though it does not guarantee real-time operation, there is still more optimization strategies to be implemented.

If, on the other hand, eliminating some of the indicators proves to be possible, ruling out blur and block-loss metrics yields an execution time below 33 ms (provided that multithread version of the software is used).

It is worth mentioning that many image processing algorithms use precisely defined, and more importantly, a finite set of operations, which may be performed on the image. Therefore, once processed, an image or parameter may be stored and used again in other metrics. This strategy works best if the amount of data to be stored does not exceed some threshold value, which defines the balance point for a trade-off between memory usage and computational complexity.

Yet another possible optimization scenario is to move as much computations as possible into the domain of integer numbers. This is justified only if one plans to use the central processing unit (CPU) exclusively. Due to its internal topology, it performs best when used with this kind of data.

All optimization methods described operate in the software layer of the system design. Apart from those, one can always try to port the code to another hardware platform like the Graphics Processing Unit (GPU) or Field Programmable Gate Array (FGPA). Both solutions allow us to massively parallelize the execution and thus reduce the time needed for processing. However, advantageous features of both these solutions come at a price of thorough source code rebuilding that is necessary to gain maximum performance boost.



Blur	Blackout	Blockloss	Blockiness	Contrast	Exposure	Freezing	Interlace	Letterbox	Noise	Pillarbox	Slicing	TA	SA
100	100	100	100	100	100	100	100	100	100	100	100	100	100
0	0	0	0	0	0	0	0	0	0	0	0	0	0
20	0	3	5	42	43	0	0	0	1	0	0	15	7

Fig. 3. The graphical user interface of application measuring QoE.

3.3. Additional Scripts

As an addition, several automated calculation scripts are provided. To achieve a high level of portability, all of the scripts were written both for Unix-like and Microsoft Windows systems. Obtaining this extent of versatility required the creation of two separate implementations. One written in Bash (Linux, Mac OS) and one in Batch (Windows). Utilization of FFmpeg tools allowed reducing the input interface to a single parameter, namely the path to video sequence or folder containing video materials to process. Automation scripts are based on the assumption that all input data is stored in the form including detailed information about its content. This mechanization allows one to seamlessly apply the presented measurement techniques to a large set of input data, be it images or videos.

3.4. Versions

One of the most important aspects accompanying the development process was the assumption that if possible, the application should be platform independent. As a result, the software package was released for all the most popular operating systems: Linux, Mac OS and Windows. Though multi-sided, the software's implementation remains consistent, meaning that a single source code may be used to compile into all supported binaries. Minute changes in the configuration file is enough to quickly switch between the desired operating system (OS) and architecture type (32- or 64-bit).

The described software is provided free of charge (for non-commercial usage) and may be downloaded from the web page [11].

3.5. Graphical User Interface

Keeping in mind that presentation of the software is of key importance, the authors decided to additionally implement a graphical user interface. Its main advantage is the possibility of simultaneous observation of results and the currently processed video sequence. Figure 3 shows an example of the described software. The graphical version of the measurement system is capable of processing any video stream, provided its content is made available in a shared memory. Thus, it is necessary to introduce a thin integration layer decompressing video stream and uploading raw frames into memory shared with QoE application. This kind of solution was developed and tested inside the MITSU project. Connecting transcoding software with the QoE measurement system allowed us to create dynamically changing video streams that aim to maximize user experience in terms of QoE.

4. Predicting VQM

To obtain the global quality indicator we used VQM metric as a grand truth. In order to create such global metric we considered a specific case of adaptive streaming. Note

Table 1
Compression parameters used in the experiment

	2	4	5	6	7	8	9	10
Bitrate	100	300	500	1000	2500	5000	7000	10000
Resolution	256×144	424×240	424×240	640×360	854×480	1280×720	1920×1080	1920×1080
Profile	Base	Base	Main	Main	Main	Main	Main	Main
No. of slices	1	1	1	2	2	2	2	2

that such assumption limited the obtained result since some distortion, like packet losses, were not considered. On the other hand, the adaptive streaming is the growing market right now. In addition, the FR metrics, like VQM, works better for compression only sequences than for packet loss sequences. Therefore, our reference is more precise.

Adaptive streaming is mainly compression with resolution change. Therefore, after analyzing the information about typical adaptive streaming compression parameters settings presented in Table 1 were used.

We also considered the frame rate change but the obtained full reference values look unrealistic and we decided to drop those cases. This is the reason why the compression cases are numbered 2, 4, 5, etc. leaving one and four.

Compression parameters are one important factor of the differences in the quality. The second one is content. Obviously, the quality of animation sequence will be very different comparing with the quality of fast are reach in details sequence. Therefore, diverse content was the key factor to obtain reasonable results.

All 1080p sequences were downloaded from the CDVL library (www.cdvl.org). After the analysis, we chose 44 different source sequences. Each sequence was divided into two seconds long chunks, which simulated adaptive streaming chunks. Therefore, we obtained 361 source chunks. Those chunks represented very different, from coding complexity, conditions. It is especially important since a typical subjective experiment contains only few source sequences.

We created the processed video sequences (PVSs) for all source video sequences (SRC). The VQM was calculated by copying the same two seconds three times to increase the length of the sequence to six seconds needed by VQM algorithm. This process could influence the correctness of the VQM metric but we are not able to calculate the magnitude of that influence.

The next step was calculating values of the presented metrics. Since all the metrics are image based it was possible to calculate them for the short, two seconds long, sequences. The single value for the PVS chunk was obtained by calculating the mean over the all values. We also added temporal and spatial activity/information metrics called SA and TA or SI and TI [12]. Those metrics helps in the context characterization even if they are calculated on the distorted sequences, which is out of scope of the original use-case. We used those metrics in on compressed sequences previously obtaining reasonable results [13].

The final data set contains 3242 rows and 9 different metrics. We did not consider all metrics since some of them are very fare from the adaptive streaming use-case, like letterboxing or pillarboxing. The final metrics were: blockiness, blockloss, blur, brightness, contrast, exposure, noise, SA, and TA. Those metrics were considered as potentially useful but we limited the list further after the results were obtained.

Knowing that the source sequences are very different and it would be difficult to characterize them we decided to use a machine learning algorithm Support Vector Machine (SVM) to predict VQM. To avoid over fitting we used 5-Fold Cross Validation method [14]. The method is based on dividing the learning set in to five different sets, running the learning process on the combined four sets and using for the verification the last set. It is done for each possible combination giving five different results. More details can be found in [14].

The 5-Fold Cross Validation assumes that each point is independent and can be used for training or testing. It is not the case for our data since the content is one of the key factors. Therefore, dividing the data set to the subsets had to be done carefully. First, all PVSs and chunks generated from the same SRC have to be in the same set. Thanks to that restriction we never train and test with the same content.

After dividing the data set we were able to run the SVM algorithm. In order to find the best model we need to decide, which variables are the most meaning full and we have to find the best pair of the two SVM fitting parameters cost and γ. With relatively small amount of metrics, we were able to test all possible SVM models for limited cost-to-γ pairs. For the models showing the best performance, we run more precise search for optimal cost-to-γ pairs. This method let us find the final model which is based on four metrics: blockiness, blur, noise, spatial activity (SA) with blur being the most significant factor.

The stability of the obtained model is strongly depending on the exact source sequence distribution. It proves that more contents and more content characteristics are needed to obtain better results. Just as an example, two different scatter plots, showing the SVM model precision, are shown in Figs. 4 and 5.

The obtained scatter plots show the strong influence of the source sequences. If very specific sequences were only in the validation set, the obtained results are much worst. It

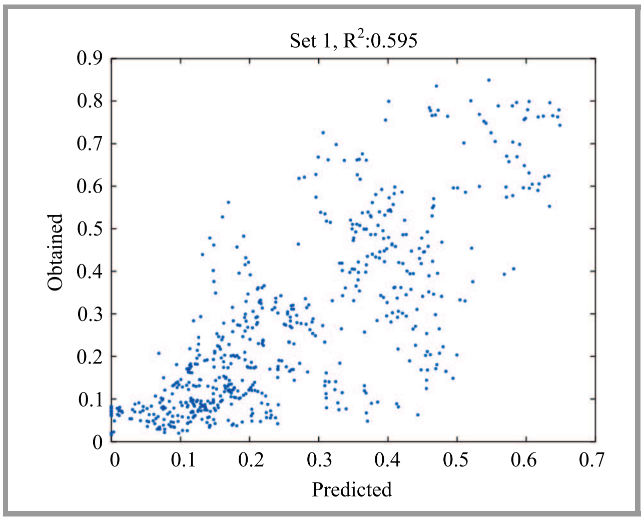


Fig. 4. The scatter plot of the worst performing validation set.

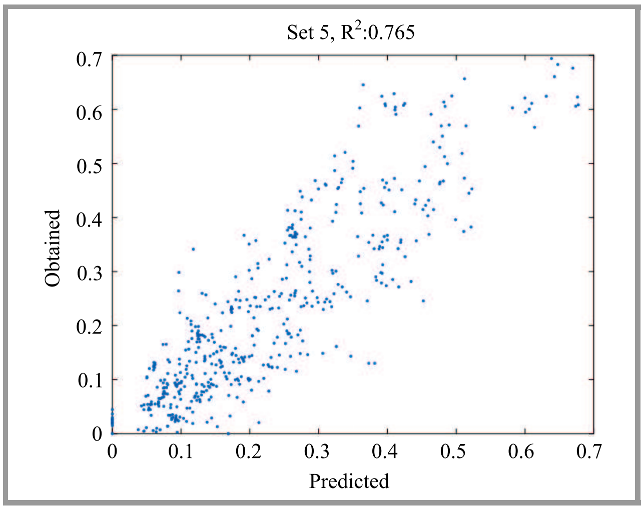


Fig. 5. The scatter plot of the best performing validation set.

shows clearly that to create a correct model very broad range of sources has to be used.

After choosing the best SVM model, we also analyzed linear models. In this case, we focused on the four metrics which we already have in the model, but we considered not only linear terms but also all possible interactions and the square terms. The same as for SVM the variables have to be normalized in order to obtain stable results. The normalization we used is $(-1,1)$ interval by equation $xn = 2\frac{x-a}{b-a} - 1$. The coefficients used for the normalization are shown in Table 2.

Table 2
The normalization coefficients used for the linear model

Indicator	Min (a)	Max (b)
Blockiness	0.553	1.123
SA	4.401	150.410
Blur	2.354	33.944
Noise	0.000	1.422

The obtained model has couple of statistically insignificant terms. Reducing them one by one, we obtained model presented in Table 3.

Table 3
The smallest linear model with all coefficients statistically significant

Indicator	Coefficient
(Intercept)	–0.097
Blockiness	2.058
Blockiness^2	0.151
Blur	–0.349
Blur^2	–0.144
Blockiness: SA	1.460
Blockiness: blur	2.471
Blockiness: noise	0.254
SA: noise	–0.069
Blur: noise	0.184
Blockiness: SA: blur	1.504

For the linear model the obtained $R^2 = 0.69$, which is comparable with the SVM model.

5. Conclusions

QoE metrics have been successfully developed as a result of the work. All together constitute a single, universal and multi-platform measurement system, which runs entirely on the receiving side. This ability makes it especially suitable for content providers operating on a massive scale. The opportunity to remotely sense quality of experience at each user-node guarantees better system control and gives solid input for various resource utilization algorithms. Moreover, measurement performed on two ends of the system allows one to quantitatively measure its impact on the content being transmitted.

A related point to consider is the fact that the software provides information regarding all indicators separately. Establishing trustworthy mapping between those KPIs and final subjective quality is a challenging task requiring more experimental data nevertheless such mapping is also proposed. More studies are needed to perform better correlation with the subjective results or objective FR metric.

Due to clear and comprehensive presentation of results, the user alone may choose the meaning and importance of certain metrics, not only focusing on the global score. The global score can be used as an error indicator and the deep metrics analysis can show the exact reason of the error.

6. Acknowledgments

This work was (partially) supported by the Polish Ministry of Science and Higher Education under European Regional Development Fund, Grant POIG.01.01.02-00-045/09-00, Future Internet Engineering.

References

[1] M. Leszczuk, M. Hanusiak, M. C. Q. Farias, E. Wyckens, and G. Heston, "Recent developments in visual quality monitoring by key performance indicators", *Multim. Tools and Appl.*, vol. 75, no. 17, pp. 10745–10767, 2014 (doi: 10.1007/s11042-014-2229-2).

[2] M. H. Pinson, L. K. Choi, and A. C. Bovik, "Temporal video quality model accounting for variable frame delay distortions", *IEEE Trans. on Broadcast.*, vol. 60, no. 4, pp. 637–649, 2014 (doi: 10.1109/TBC.2014.2365260).

[3] M. Leszczuk *et al.*, "Key indicators for monitoring of audiovisual quality", in *Proc. 22nd Sig. Process. and Communications Appl. Conf. SIU 2014*, Trabzon, Turkey, 2014, pp. 2301–2305 (doi: 10.1109/SIU.2014.6830724).

[4] K. Zhu, C. Li, V. Asari, and D. Saupe, "No-reference video quality assessment based on artifact measurement and statistical analysis", *IEEE Trans. on Circ. and Sys. for Video Technol.*, vol. 25, no. 4, pp. 533–546, 2015 (doi: 10.1109/TCSVT.2014.2363737).

[5] X. Li, Q. Guo, and X. Lu, "Spatiotemporal statistics for video quality assessment", *IEEE Trans. on Image Process.*, vol. 25, no. 7, pp. 3329–3342, 2016 (doi: 10.1109/TIP.2016.2568752).

[6] M. Leszczuk, "Assessing task-based video quality – a journey from subjective psycho-physical experiments to objective quality models", in *Multimedia Communications, Services and Security. 4th International Conference, MCSS 2011, Krakow, Poland, June 2-3, 2011. Proceedings*, A. Dziech and A. Czyżewski, Eds. *CCIS*, vol. 149, pp. 91–99. Springer, 2011 (doi: 10.1007/978-3-642-21512-4_11).

[7] L. Janowski and Z. Papir, "Modeling subjective tests of quality of experience with a generalized linear model", in *Proc. Quality of Int. Worksh. on Multim. Exper. QoMEx 2009*, San Diego, CA, USA, 2009, pp. 35–40 (doi: 10.1109/QOMEX.2009.5246979).

[8] P. Romaniak, L. Janowski, M. Leszczuk, and Z. Papir, "Perceptual quality assessment for H.264/AVC compression", in *Proc. IEEE Consumer Commun. and Netw. Conf. CCNC 2012*, Las Vegas, NV, USA, 2012, pp. 597–602 (doi: 10.1109/CCNC.2012.6181021).

[9] M. Mu, P. Romaniak, A. Mauthe, M. Leszczuk, L. Janowski, and E. Cerqueira, "Framework for the integrated video quality assessment", Multim. Tools and Appl., vol. 61, no. 3, pp. 787–817, 2012 (doi: 10.1007/s11042-011-0946-3).

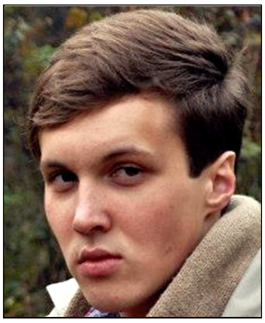
[10] J. Søgaard, S. Tavakoli, K. Brunnström, and N. García, "Subjective analysis and objective characterization of adaptive bitrate videos", in *Proc. IS&T Int. Symp. on Elec. Imaging 2016: Image Quality and System Performance XIII*, San Francisco, CA, USA, 2016 (doi: 10.2352/ISSN.2470-1173.2016.16HVEI-105).

[11] Video quality [Online]. Available: http://vq.kt.agh.edu.pl (2016).

[12] C. Fenimore, J. Libert, and S. Wolf, "Perceptual effects of noise in digital video compression", in *in Proc. 140th SMPTE Tech. Conf. and Exhibit*, Pasadena, CA, USA, 1998 (doi: 10.5594/M00301).

[13] L. Janowski and P. Romaniak, "QoE as a Function of Frame Rate and Resolution Changes", in *Future Multimedia Networking Third International Workshop, FMN 2010, Kraków, Poland, June 17-18, 2010. Proceedings*, S. Zeadally, E. Cerqueira, M. Curado, and M. Leszczuk, Eds. *LNCS*, vol. 6157, pp. 34–45. Springer, 2010 (doi: 10.1007/978-3-642-13789-1_4).

[14] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R.* Springer, 2014.



Jakub Nawała is a student and beginner researcher at AGH University of Science and Technology. He works at Department of Electronics and Department of Telecommunications. His research concerns computer vision, video quality assessment and embedded systems programming.

E-mail: jakub.tadeusz.nawala@gmail.com
AGH University of Science and Technology
Department of Telecommunications
Mickiewicza av. 30
30-059 Krakow, Poland

Mikołaj Leszczuk started his professional career in 1996 at Comarch SA as manager of the Multimedia Technology Department, and then at Comarch Multimedia as the CEO. Since 1999 has been employed at the AGH Department of Telecommunications. In 2000, he moved to Spain for a four-month scholarship at the Universidad Carlos III de Madrid. After returning to Poland, he was employed at the Department of Telecommunications as a Research and Teaching Assistant, and in 2006, he did his Ph.D. His current research interests are focused on multimedia data analysis and processing systems, with emphasis on Quality of Experience. He (co-)authored approximately 130 scientific publications of which 23 are publications in journals of the JCR database.

E-mail: leszczuk@agh.edu.pl
AGH University of Science and Technology
Department of Telecommunications
Mickiewicza av. 30
30-059 Krakow, Poland

Lucjan Janowski is an Assistant Professor with the Department of Telecommunications, AGH University of Science and Technology. He received his Ph.D. degree in Telecommunications in 2006 from the AGH. In 2007, he worked in a postdoctoral position at the Centre National de la Recherche Scientifique (CNRS), LAAS (Laboratory for Analysis and Architecture of Systems of CNRS) in France, where he prepared both malicious traffic analysis and anomaly detection algorithms. In 2010–2011, he spent half a year in a postdoctoral position at the University of Geneva, working on quality of experience (QoE) for health applications. In 2014–2015, he spent half a year in a postdoctoral position at The Telecommunications Research Centre Vienna (FTW), working on quality of experience for IPTV customers. His main interests are statistics and probabilistic modeling of subjects and subjective rates used in QoE evaluation.

E-mail: janowski@kt.agh.edu.pl
AGH University of Science and Technology
Department of Telecommunications
Mickiewicza av. 30
30-059 Krakow, Poland