A Scalable Deep Neural Network to Detect Low Quality Images Without a Reference

Zongyi (Joe) Liu*, Bruce Ferry, Simon Lacasse
Amazon.com
Seattle, Washington, USA 98121
Email: joeliu*, bferry, lacasse@amazon.com

***Abstract*—Online streaming services have been growing at a fast pace. To provide the best user experience, it is needed to detect low quality images from videos so that we can repair or improve them before showing to customers. For example, for movie and TV-show streaming services, it is important to check if an original (master) video produced from a studio has low quality images that contain artifacts such as up-scaling or interlacing; for live streaming services, it is important to detect if a streamed video have hits due to encoding such as H.264 or MPEG-2. The impairment detection is usually measured by the non-reference (NR) metrics because it is often difficult and sometimes impossible to get the original (master) videos. On the other hand, today researches in the image quality area, such as super-resolution, are mainly focused on the full reference (FR) metrics like PSNR or VMAF. In this paper, we present an algorithm that is able to reliably compute five types of spatial NR metrics that are commonly used in the online streaming industries. The algorithm consists of two components: a pre-processing step that spatially de-correlates pixel intensity values and a novel deep neural network (DNN) that is able to quantify the NR metrics at the image region level. We show that our algorithm achieves better performance than state-of-art algorithms in this area.**

I. Introduction

Automatically measuring image quality has been actively studied in recent years. Generally speaking, these metrics can be categorized into three types: (i) full-reference (FR) metrics that require a complete noise-free signal (master). For example, peak signal noise ratio (PSNR), mean square error (MSE), or more complicated perceptual visual quality metrics such as SSIM [1], VDP [2], PEVQ [3], VMAF [4]. (ii) Reduced-reference (RR) metrics that require some samples from the master. For this type, people usually build a model using the limited reference data. For example, M. Tagliasacchi *et. al* [5] modeled the noise in the testing images, and J. A. Redi *et. al* [6] studied the human visual impact using color distortion. (iii) Non-reference (NR) metrics that do not require a master. For example, the Blind/Referenceless Image Spatial Quality Evaluator (BRISQUE) [7] and the Naturalness Image Quality Evaluator (NIQE) [8]. Some of the recent works include using periodicity analysis [9], frequency-based analysis [10], Singular Value Decomposition (SVD) based analysis [11], and combined wavelet-Fourier transforms based analysis [12]. In addition to the traditional signal processing and computer vision technologies, recently some researches have also applied the DNN in this area. For example, T. R. Goodall and A. C. Bovik [13] presented a video impairment mapper (VIDMAP) that is able to compute eight types of NR metrics.

Compared with the full-reference and reduced-reference metrics, the NR metrics have the following advantages that are particularly useful for detecting low quality images for online streaming services: (i) we don't need to perform spatial or temporal alignment between a captured video and its master video which is very computationally expensive; in addition, depending on the distribution agreements between a studio and a service provider, comparing a video captured in a mobile app such as the Prime Video with its master video can be legally prohibited; (ii) for some services such as live event streaming, there is no master video to compare with; and (iii) the master videos need to be inspected too because they may also contain low quality images. However, today the researches in this area, such as super-resolution, are mainly focused on the FR metrics like PSNR or VMAF instead of NR metrics.

In this paper, we present an algorithm that can reliably quantify five spatial NR metrics that have been used to detect impairments during source video inspection or live streaming video quality analysis: up-scaling, combing (interlacing), H.264 hits, MPEG-2 hits and compression. Fig. 1 shows publicly available web crawled sample images. The algorithm starts with a pre-processing step that performs the Mean-Subtracted Contrast Normalization (MSCN), then runs a novel DNN model that computes the region level impaired probability for an input image. We show that our algorithm not only achieves better accuracy than state-of-art algorithms, but also is much faster than the other DNN-based algorithms so that it can be scaled up to work for the live streaming product line.

The remainder of the paper is organized as follows. In Sec. II we describe the algorithm in detail, in Sec. III we evaluate the algorithm's performance using a video dataset, and in Sec. IV we conclude the paper and explore future studies.

II. Algorithm Description

Our algorithm consists of two steps. The first step is an image pre-processing step that runs the Mean-Subtracted Contrast Normalization (MSCN) in the intensity channel, and the second step is an end-to-end trainable DNN that inputs a pre-processed image and outputs the probability whether an artifact is in the image region level.

Fig. 1. The publicly available web crawled sample images with bad quality due to (a) up-scaling, (b) combing/interlacing, (c) H.264 hits, (d) MPEG-2 hits and (e) compression.

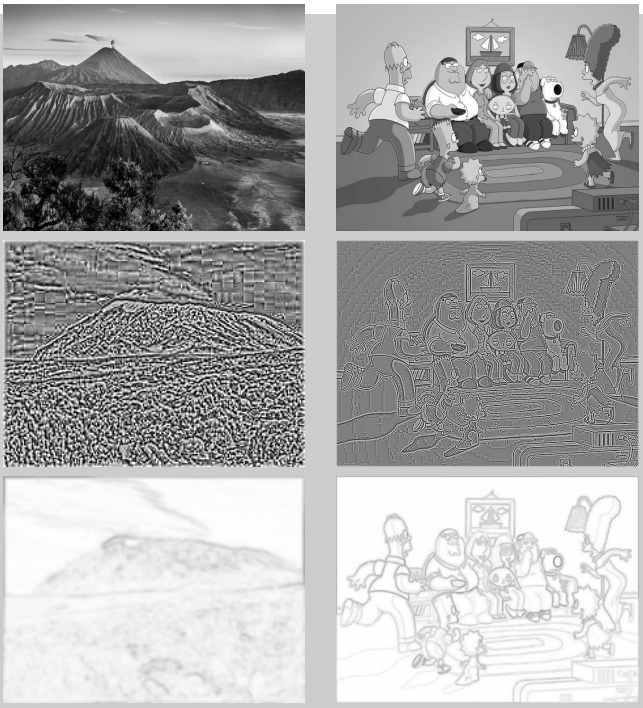

Fig. 2. The MSCN normalization on two publicly available web crawled sample images. Here the top row is the intensity channel of the original image, the middle row is the $N(Y)$ as defined in Eq. 3, and the bottom row is the $\sigma(Y)$ as defined in Eq. 2. For both images, we set the neighboring window W to 13. For better visualization, we have the $N(Y)$ (middle row) values normalized and added a constant offset (127) to show both the negative and positive parts; we also have $\sigma(Y)$ (bottom row) values inverted.

A. *Image Pre-processing*

$$\begin{aligned} Y &= 0.299R + 0.587G + 0.114B \\ U &= -0.147R - 0.289G + 0.436B \\ V &= 0.615R - 0.515G - 0.100B \end{aligned} \tag{1}$$

Given an input color (RGB values that are linearized and gamma-decompressed) image ($I_{in}(RGB)$), we first convert from RGB channels into YUV channels as Eq. 1 defines to separate the intensity and the chrominance components. Then we apply the MSCN transform that has shown success in many image quality measurement algorithms because it has been observed that Gaussianize and de-correlate pixels can better separate the high quality images from impaired images. [13], [8]. Specifically, we apply the MSCN to the intensity component $I_{in}(Y)$ as defined in Eq. 2 and 3.

$$\begin{aligned} \mu(Y)(x,y) &= mean(I_{in}(Y)(x_i,y_i)) \quad \forall x_i, y_i \in W \\ \sigma(Y)(x,y) &= stdev(I_{in}(Y)(x_i,y_i)) \quad \forall x_i, y_i \in W \end{aligned} \tag{2}$$

$$N(Y)(x,y) = \frac{I_{in}(Y)(x,y) - \mu(Y)(x,y)}{\sigma(Y)(x,y)} \tag{3}$$

Here the W is the neighbor window size for normalization, which is set to 13 as suggested in [13]. Fig. 2 shows some publicly available sample images after the MSCN normalization. For better visualization, we have the $N(Y)$ (middle row) values normalized and added a constant offset (127) to show both the negative and positive parts; we also have $\sigma(Y)$ (bottom row) values inverted.

The output of the pre-processed image (I_{pre}) contains $N(Y)(x,y)$ and $\sigma(Y)(x,y)$ as defined in 2 and 3. Here we also include the two color channels (UV) of I_{in}. As a result, the I_{pre} is a four-channel image as defined in Eq. 4.

$$I_{pre}(x,y) = [N(Y), \sigma(Y), I_{in}(U), I_{in}(V)](x,y) \tag{4}$$

B. *Image Impairment Artifact Detection Model*

The second part of the algorithm is an end-to-end trainable DNN model. Fig. 3 shows its structure. We started with a channel selection from the four channel I_{pre} produced in the pre-process step. Currently, we select $[N(Y), \sigma(Y)]$ for the combing, H.264 hits and MPEG-2 hits detection model and $[N(Y), I_{in}(U), I_{in}(V)]$ for the other two artifact detection models. This is based on our observation that combing and video hits artifacts can be modeled by the edge cue of gray level images, where up-scaling and compression artifacts are more obvious in color images. Next, we extract the features from an input image by building three convolution layers ($Conv1$, $Conv2$, and $Conv3$), each followed by a max pooling layer to reduce the dimension and then followed by an exponential rectify layer (ELU) to prevent gradient

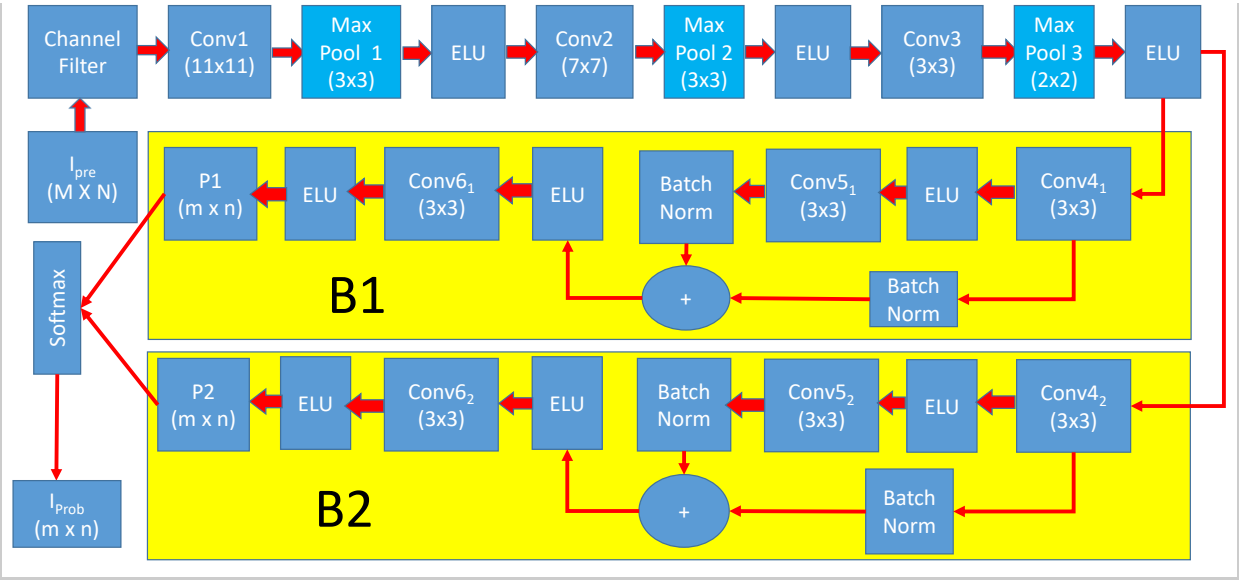

Fig. 3. The neural network structure of the image impairment artifact detection model. Here the input I_{pre} is a four-channel image after pre-processing step as defined in Eq. 4, and the output I_{prob} is the region level artifact detected probability map as defined in Eq. 5. Here M, N equal to the input image dimension, and $m = \frac{M}{18}, n = \frac{N}{18}$. We picked this two-branch architecture following [13] because it achieved better accuracy than a single branch without adding more convolution layers.

from exploding. The output of this process are the region level (REG) features. In our algorithm, the impairment detection is modeled as a binary classification problem: each REG is classified as either negative (high quality) or positive (impairment detected). So we build a dual-path (parallel) network: $B1$ and $B2$, that take the REG features as input and employ three additional convolution layers to compute response for the positive class and negative class, respectively. We picked this two-branch architecture because it achieved better accuracy than a single branch without adding more convolution layers, which in turn helps our network easier to train and do inference at faster speed. In our network, $B1$ and $B2$ are implemented using the residual network structure. For example, layer $Conv5_1$ and $Conv5_2$ are skipped over activation of $Conv4_1$ and $Conv4_2$, in order to tackle the gradient vanishing problem. Finally, the output of $B1$ and $B2$ are sent into a softmax layer to generate a binary class probability map matrix I_{prob}, as defined in Eq. 5. Like most classification DNNs, we use the cross-entropy loss function to back propagate the network during the training process.

$$I_{prob}(x,y) = \frac{e^{P2(x,y)}}{e^{P1(x,y)} + e^{P2(x,y)}} \tag{5}$$

Compared with VIDMAP [13], the main contribution of our model is that it first builds a shallow network with only three convolution layers to extract the REG level features from an input image, and then builds the dual-path network with discriminative neurons to effectively separate high quality images from impaired images. According to our study, this process has substantially improved the algorithm accuracy. In addition, these REG features have much lower dimension. Specifically, it is 324 $(9 \times 9 \times 4)$ times smaller than the input image. The low complexity of the network and the dimension reduction have dramatically reduced the computational time. Another difference is that our network uses both intensity and color channels $([N(Y), \sigma(Y), I_{in}(U), I_{in}(V)])$ but VIDMAP only uses the intensity channel $([N(Y), \sigma(Y))$. And we have observed that color helps detect some types of impairments such as the upscaling artifact. We show these improvements in Sec. III.

To detect artifacts for an input image, we first pre-process it and then run through the DNN model to get an image region level probability matrix I_{prob} as defined in Eq. 5. Fig. 4 shows publicly available sample images that are impaired by different artifacts and the output artifact detected area: I_{prob} with probability values greater than 0.5. Next, we compute the positive area ratio (P_{AR}) as defined in Eq. 6. Then we compare P_{AR} with a pre-defined threshold T_P, where we label I_{full} as positive (artifact detected) if P_{AR} is greater than T_P.

$$P_{AR} = \frac{Area(I_{prob} > 0.5)}{Area(I_{prob})} \tag{6}$$

III. Performance Evaluation and Experiments

We evaluated our model on five types of artifacts: upscaling, combing (interlacing), H.264 hits, MPEG-2 hits and compression. The dataset used in this paper were collected from the PV lab. Specifically, we first collected several hundred PV videos of different genres such as sport events, movies, cartoons, talk shows, etc. These videos have different resolution including UHD (3840×2160), HD (1920×1080) and non-HD ($< 1920 \times 1080$). Then for each video, we randomly sample 5 seconds to 60 seconds of frames as one

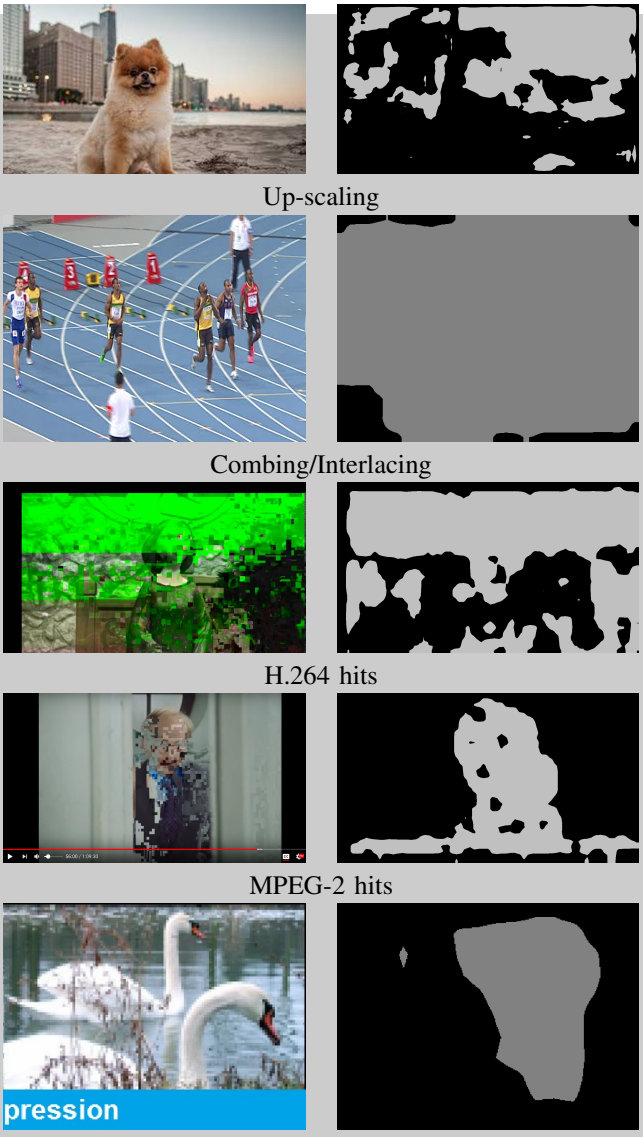

Fig. 4. The sample images (publicly available web crawled) that are impaired by different artifact (left), and the output region level detection results (right) from corresponding models. Here the bright pixels are the positive area with the probability values greater than 0.5, i.e., artifacts detected.

video clip, and then have a video specialist to manually watch it and label it as negative (pristine) or positive (with artifact).

During the data collection process, we found that the positive data (images with impairments) are usually much more difficult to acquire. So in our study, in addition to the PV lab data, we also synthesized positive frames by artificially adding artifacts into the pristine images. For example, to synthesize images with up-scaling artifact, we scaled up a clean image $4-6$ times using one of the common resizing technologies such as bilinear interpolation or nearest neighboring; to synthesize images with combing artifact, we replaced the even rows of a frame at number t with the rows in the frame at number $t+2$. For the video hits and compression artifacts, we

TABLE I
THE TRAINING DATASET WE CREATED FOR EACH ARTIFACT DETECTION. ONE VIDEO CLIP HAS A DURATION RANGING FROM 5 SECONDS TO 60 SECONDS, AND WE BUILT I_{patch} WITH A FIXED SIZE OF 256×256 AS TRAINING SAMPLES. HERE WE BUILT AN EQUAL NUMBER OF POSITIVE AND NEGATIVE TRAINING IMAGES USING OVER-SAMPLING.

Artifact	No. of video clips	No. of positive or negative I_{patch}
Up-scaling	127	177,096
Combing	82	506,524
H.264 hits	83	261,396
MPEG-2 hits	84	460,084
Compression	93	151,584

used the $FFMPEG$ tool [14] to generate positive images.

Similar to T. R. Goodall and A. C. Bovik's work [13], we trained one DNN model to detect one type of impairment. For a given artifact model, we split the videos into non-overlapping training and testing set. For the training data creation, we first sampled no more than 300 frames from each video clip with a fixed step size in order to prevent data from being biased toward long clips. Then for each frame (I_{full}), we created $20-25$ sub-images (I_{patch}) with a fixed size of (256×256) by random spatial sampling. The advantages of using I_{patch} include that (i) it increases the total number of training samples, (ii) it improves the training speed because I_{patch} is much smaller than I_{full}, and (iii) all I_{patch} images have the same dimension so that they are easy to be batched into a deep learning framework such as MxNet.

In our experiments, the negative training data is a collection of $I_{patches}$ built using the negative images ($I_{pristine}$). The positive training data, on the other hand, is mainly a collection of $I_{patches}$ built from the synthesized images as described in Sec. II-B. But we also include I_{patch} built from a small subset of positive video clips collected from the PV lab. This is because that the artifact generation algorithm can not fully simulate the variations for some artifacts. For example, the upscaling artifact generation algorithm applies one of the four image resizing algorithms provided by OpenCV. But there are positive video clips from our lab that were upscaled using other types of algorithms. So it is necessary to include some of these images into the training dataset in order to build a robust classifier. Table I lists the details of our training dataset for each artifact. Note that we used the over-sampling to build a equal number of positive and negative training image samples.

For the testing data creation, instead of building I_{patch}, we directly used the I_{full} from the video clips in the testing dataset. Similar to the training data creation, the testing dataset also used the $I_{pristine}$ as the negative samples, and used the images from the positive videos plus the synthesized images as the positive samples. For the test dataset, we only include the HD and non-HD images. This is because (i) we do not have positive UHD videos from our lab, where a test set only consists of synthesized data is not convincing enough; and (ii) the literature algorithms we used to compare performance with are only trained using HD and non-HD images so that

TABLE II
THE TESTING DATASET WITH HD AND NON-HD IMAGES FOR EACH ARTIFACT DETECTION. NOTE THAT THERE ARE TWO TYPES OF SOURCES: THE **L** MEANS THE DATA ARE FROM OUR LAB AND ARE LABELLED BY A VIDEO SPECIALIST, AND THE **S** MEANS THE DATA ARE THE SYNTHESIZED IMAGES AS DESCRIBED IN II-B.

Artifact	No. of pos. videos	No. of pos. images	Pos. image source	No. of neg. videos	No. of neg. images	Neg. image source
Up-scaling	41	15346	L	44	15575	L
Combing	51	7337	L	60	8806	L
H.264 hits	55	7733	S + L	58	10097	L
MPEG-2 hits	53	9369	S	58	9976	L
Compression	43	9747	S	52	5125	L

this helps to have a fair comparison. Table II lists the details of our testing dataset.

Given a testing I_{full}, our algorithm labels it as positive (artifact detected) if the P_{AR} as defined in Eq. 6 is greater than T_P. For comparison, we also ran the state of the art VIDMAP algorithm using their models pre-trained for these artifacts [15]. In their paper [13], T. R. Goodall and A. C. Bovik claimed that an image is classified as bad quality if one or more pixels in the output map matrix have a positive probability greater than 0.5. However, when testing on I_{full} in our test dataset, we found that this method creates a lot of false alarms. So we modified the classification process to be similar to our algorithm: we computed P_{AR} as defined in Eq. 6 from the output probability map matrix of VIDMAP, and then plotted the ROC using different T_P values. In addition to the neural network based approaches, we also ran MITSU algorithm proposed by AGH University [16] that uses the traditional computer vision technologies to compute the non-reference video qualities. For example, to compute the blockiness metric, absolute differences in pixel luminance were calculated separately for intra-pairs, represented by neighbouring pixels from a single coding block, and inter-pairs, represented by pixels from neighbouring blocks. A ratio between the total values of intra- and inter-differences is calculated over the entire video frame [17]. Based on their papers, we used its blur indicator [17], [18] for the up-scaling testing dataset, the interlacing indicator [19] for the combing testing dataset, and blockloss indicator [20] for the H.264 hits and MPEG-2 hits testing dataset.

Fig. 5 plots the Receiver Operating Characteristic (ROC) curves using different values of T_P for our algorithm. In addition to the ROC curves, we also compared the detection rate of these three algorithms at the false positive rate of 5% and 10% in Table. III. It showed that our algorithm achieves better performances than the other two in terms of all five artifacts detections.

In addition to accuracy, we also measured the running time of our algorithm and compared it with VIDMAP algorithm that

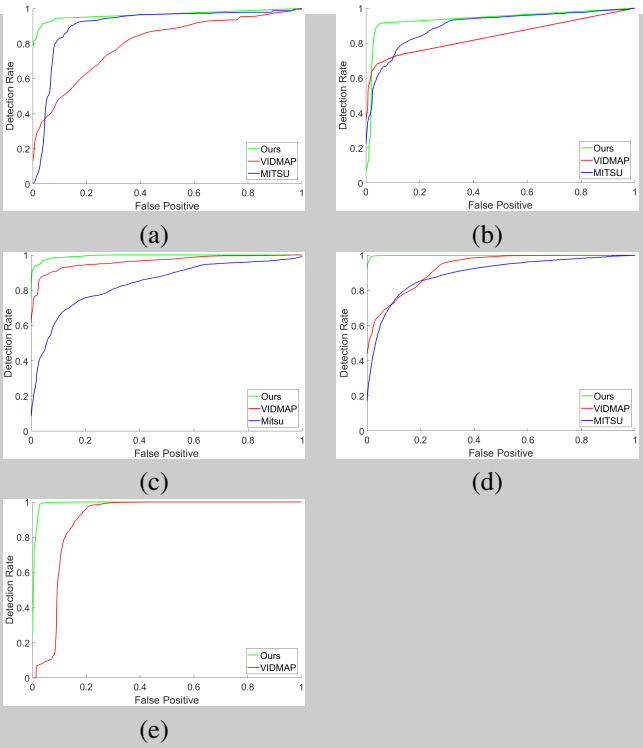

Fig. 5. The ROC curves computed using our algorithm, VIDMAP algorithm [13] and MITSU algorithm [17], [18], [19] for the artifact of (a) up-scaling, (b) combing/interlacing, (c) H.264 hits, (d) MPEG-2 hits, and (e) compression.

TABLE III
THE ALGORITHM PERFORMANCE COMPARISON BETWEEN OUR ALGORITHM, VIDMAP ALGORITHM [13] AND MITSU ALGORITHM [17], [18], [19] IN TERMS OF DETECTION RATE AT FALSE POSITIVE RATE (FP) OF 5% AND 10%, RESPECTIVELY.

Artifact	FP	Ours	VIDMAP	MITSU
Up-scaling	5%	**92.0**	38	46.4
	10%	**94.4**	48.7	83
Combing	5%	**90.9**	68.5	63.1
	10%	**91.5**	72.0	71.4
H.264 hits	5%	**98.3**	86.3	75.6
	10%	**99.6**	88.47	80.9
MPEG-2 hits	5%	**99.2**	66.2	60.1
	10%	**99.8**	72.8	73.7
Compression	5%	**99.6**	9.4	N/A
	10%	**99.9**	62.4	N/A

is based on the neural network as well. Table. IV listed the running time of both algorithms for images of size 1920×1080 on an AWS $P3$ $2\times$ large instance with one NVIDIA Tesla V100-SXM2 GPU and an Intel(R) Xeon(R) CPU @2.30GHz. We can see that both algorithms take ~ 180 milliseconds in the image pre-processing part. But for the DNN part, VIDMAP algorithm takes about $1,200$ milliseconds where our algorithm only takes about 40 milliseconds which is about 30 times faster. This suggests that our algorithm is much more scalable because the DNN model is expensive to run as it requires GPU hosts where the pre-processing step can be distributed to run on CPU only hosts.

TABLE IV
THE RUNNING TIME IN TERMS OF MILLISECONDS FOR OUR ALGORITHM AND THE STATE OF THE ART VIDMAP ALGORITHM [13] WHEN PROCESSING IMAGES WITH SIZE OF 1920×1080. THE TEST MACHINE IS AN AWS $P3\,2X$ LARGE INSTANCE WITH ONE NVIDIA TESLA V100-SXM2 GPU AND AN INTEL(R) XEON(R) CPU @2.30GHZ.

Algorithm	Pre-processing (CPU Only)	DNN (GPU Required)
Ours	180	40
VIDMAP	180	1150

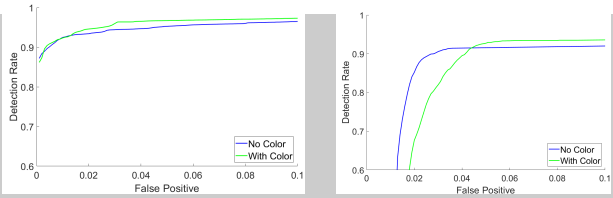

Fig. 6. The performance comparison between the models using intensity only (no color) input and color input. The testing dataset has included UHD, HD and non-HD images.

IV. CONCLUSIONS AND FUTURE STUDY

In this paper, we presented an algorithm to detect the spatial image impairment without a reference. The algorithm consists of two steps: a pre-processing step that performs the MSCN in the intensity channel, and a DNN model that computes the region-level probability whether an artifact is detected. We built models for five types of artifacts: up-scaling, combing, H.264 hits , MPEG-2 hits and compression. The training dataset are the 256×256 I_{patch} cropped from images collected from the PV lab. The negative samples are built on the $I_{pristine}$ and the positive samples are synthesized by artificially adding impairing artifacts into $I_{pristine}$. We tested the algorithm using I_{full} from a separated video clips collected from our lab. The results showed that our algorithm outperformed both the state of the art DNN algorithm VIDMAP and the traditional computer vision technology based MITSU algorithm. In addition to the accuracy, our algorithm is also more scalable as the DNN part only takes 40 milliseconds for a HD image with dimension of 1920×1080 that is 30 times faster than VIDMAP.

For the input signal as defined in Eq. 4, we picked the intensity value $[N(Y), \sigma(Y)]$ for the combing and hits artifact detection models, and added color $[I_{in}(U), I_{in}(V)]$ for the up-scaling and compress model. To have an insight of how the color input impacts on the recognition performance, we computed the ROC curves of using intensity only vs. adding color input for the upscaling and combing detection DNN model. Fig. 6 showed the plots for the false positives from 0 to 10%. Note here we also added a few UHD pristine images. We can see that the added color input has boosted the accuracy of upscaling model for about $2-3\%$. But for the combing model, the added color input has worse performance when the false positive is less than 4%, and then outperforms the intensity only input for about 2%. This suggests that color input has different impacts for different artifact detection, and it will be interesting to investigate the underlying reasons.

In terms of future study, one direction is that in addition to the spatial image, we also add temporal information into the train process, so that we can pick video level impairments such as frame dropping and flickering. Another direction is that instead of having a binary DNN model for each artifact, we train a DNN model that can detect multiple artifacts in a single pass. However, we need to be careful because one image may be impaired by multiple artifacts. Also, different artifacts may have similar visual impacts. For example, it can be difficult to separate the aliasing artifact from the combing artifact when the input is a single image. Another direction is that out algorithm outputs the detection at the region level today. This is sufficient for applications that only need to detect impairment at image level. But for applications requiring pixel level detection, we should then improve the algorithm to have better precision.

REFERENCES

[1] Z. Wang, A. C. Bovik, H. R. Sheikh, and E. P. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, April 2004.

[2] Daly and Scott, "Digital images and human vision," A. B. Watson, Ed. Cambridge, MA, USA: MIT Press, 1993, ch. The Visible Differences Predictor: An Algorithm for the Assessment of Image Fidelity, pp. 179–206.

[3] "Objective perceptual multimedia video quality measurement in the presence of a full reference," p. 108, August 2008.

[4] Z. Li, A. Aaron, I. Katsavounidis, A. Moorthy, and M. Manohara, "Toward a practical perceptual video quality metric," June 2016.

[5] M. Tagliasacchi, G. Valenzise, and M. e. a. Naccari, "A reduced-reference structural similarity approximation for videos corrupted by channel errors," in *Multimed Tools Appl.* Springer US, July 2010, vol. 48, pp. 471–492.

[6] J. A. Redi, P. Gastaldo, I. Heynderickx, and R. Zunino, "Color distribution information for the reduced-reference assessment of perceived image quality," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 20, no. 12, pp. 1757–1769, Dec 2010.

[7] A. Mittal, A. K. Moorthy, and A. C. Bovik, "No-reference image quality assessment in the spatial domain," *IEEE Transactions on Image Processing*, vol. 21, no. 12, pp. 4695–4708, Dec 2012.

[8] A. Mittal, R. Soundararajan, and A. C. Bovik, "Making a completely blind image quality analyzer," *IEEE Signal Processing Letters*, vol. 20, no. 3, pp. 209–212, March 2013.

[9] S.-J. Ryu and H.-K. Lee, "Estimation of linear transformation by analyzing the periodicity of interpolation," *Pattern Recognition Letters*, vol. 36, pp. 89–99, Jan. 2014.

[10] I. Katsavounidis, A. Aaron, and D. Ronca, "Native resolution detection of video sequences," in *SMPTE 2015 Annual Technical Conference and Exhibition*, Oct 2015, pp. 1–20.

[11] D. Vzquez-Padn, P. Comesaa, and F. Prez-Gonzlez, "An svd approach to forensic image resampling detection," in *2015 23rd European Signal Processing Conference (EUSIPCO)*, Aug 2015, pp. 2067–2071.

[12] E. Chae, W. Kang, E. Lee, S. Kim, and J. Paik, "Spatially adaptive antialiasing for enhancement of mobile imaging systems using combined wavelet-fourier transforms," in *2013 IEEE International Conference on Consumer Electronics (ICCE)*, Jan 2013, pp. 86–87.

[13] T. R. Goodall and A. C. Bovik, "Detecting and mapping video impairments," *IEEE Transactions on Image Processing*, vol. 28, no. 6, pp. 2680–2691, June 2019.

[14] "Ffmpeg: a complete, cross-platform solution to record, convert and stream audio and video." [Online]. Available: https://ffmpeg.org/

[15] "Vidmap source code and pre-trained model," [Online]. [Online]. Available: http://live.ece.utexas.edu/research/quality/VIDMAP_release.zip
[16] "Mitsu video quality indicators." [Online]. Available: http://vq.kt.agh.edu.pl/metrics.html
[17] P. Romaniak, L. Janowski, M. Leszczuk, and Z. Papir, "Perceptual quality assessment for h.264/avc compression," in *2012 IEEE Consumer Communications and Networking Conference (CCNC)*, Jan 2012, pp. 597–602.
[18] M. Mu, P. Romaniak, A. Mauthe, M. Leszczuk, L. Janowski, and E. Cerqueira, "Framework for the integrated video quality assessment," in *Multimedia Tools and Applications*, vol. 61, Dec 2011, p. 787817.
[19] L. Janowski and Z. Papir, "Modeling subjective tests of quality of experience with a generalized linear model," in *2009 International Workshop on Quality of Multimedia Experience*, July 2009, pp. 35–40.
[20] M. Leszczuk, M. Hanusiak, M. C. Q. Farias, E. Wyckens, and G. Heston, "Recent developments in visual quality monitoring by key performance indicators," in *Multimedia Tools and Applications*, vol. 75, Sep 2016, pp. 10 745–10 767.