(Paper ID: 86)

25th INTERNATIONAL CONFERENCE ON PATTERN RECOGNITION

ICPR 2020

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

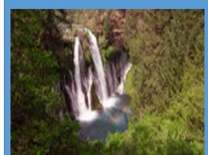
Zongyi (Joe) Liu, Bruce Ferry, Simon Lacasse

Finance org, Amazon.com

Summary

- It is important to measure image quality for the Prime Video team
  - Improve their live streaming and movie on demand services
- Image quality metrics type
  - Full Reference (FR): require a master video/image, e.g., PSNR, MSE, VMAF
  - Reduced Reference (RR): require some samples from the master
  - Non Reference (NR): no master required, e.g., BRISQUE, NIQE, VIDMAP
- Advantages of using non-reference metrics
  - Expense to spatially and temporally align a master video and a captured video
    - A movie of 2 hours captured at 30fps: 2 * 3600 * 30 = 216,000 frames
  - No master video: live streaming
- Contributions of this work
  - A novel DNN that has better accuracy than state-of-art algorithms and is scalable
  - Measure 5 commonly seen NR artifacts: upscaling, interlacing, h264 hits, mpeg2 hits and compression

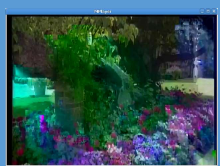
Sample Images Impaired By Artifacts



Upscaling

Interlacing

Compression



H264 Hits

Mpeg2 Hits

Algorithm Part 1: Image Preprocessing

Input Image (I_{in}) → RGB →

Color Transform
Y = 0.299R + 0.587G + 0.114B
U = -0.147R + 0.289G + 0.436B
V = 0.615R + 0.515G + 0.100B

→ Intensity Channel Y →

MSCN Transform

$$\mu(Y)(x,y) = mean\left(I_{in}(Y)(x_i, yi)\right) \forall xi, yi \in W$$

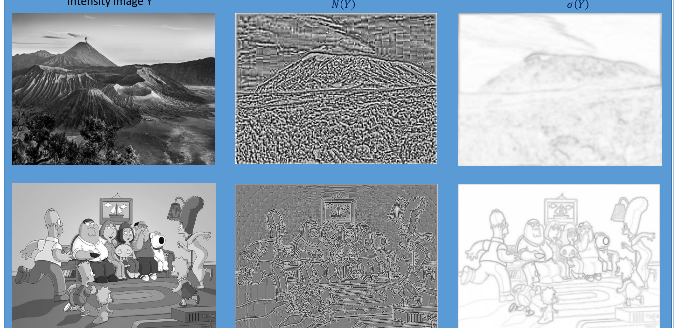
$$\sigma(Y)(x,y) = stdev\left(I_{in}(Y)(x_i, yi)\right) \forall xi, yi \in W$$

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

Intensity Image Y

$N(Y)$

$\sigma(Y)$



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

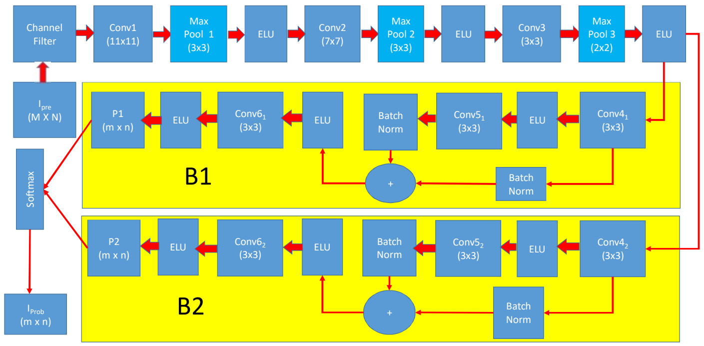
- Why use MSCN transform:

It has been observed that Gaussianize and de-correlate pixels can better separate the high quality images from impaired images.

- Why use color (uv):

It is our observation that color helps some artifacts such as upscaling and compression

Algorithm Part 2: Image Impairment Artifact Detection Model



Channel Selection

Upscaling and compression: $[N(Y), I_{in}(U), I_{in}(V)]$

Interlacing and hits: $[N(Y), \sigma(Y)]$

Probability: Softmax

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

Image Level Score

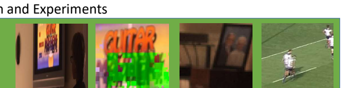
$$P_{AR} = \frac{Area(I_{prob} > 0.5)}{Area(I_{prob})}$$

Input Image | Detected Positive Regions

Performance Evaluation and Experiments

Dataset: video clips
- UHD: 3840 x 2160
- HD: 1920 x 1080
- non-HD: <1920 x 1080
- 5-60 seconds

Training Data Generation
- Crop 256 x 256 image patches
- Use clean images as class 0
- Use synthesized images as class 1



Clean | H264 hit | Upscaling | Interlacing

Testing Data Generation
- Input the full image into the model
- For Upscaling and interlacing, we use the images identified by the video specialists from the PV lab data
- For the h264 hits, mpeg2 hits and compression, we use the synthesized images.

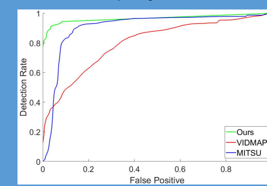
The training dataset we created for each artifact detection. One video clip has a duration ranging from 5 seconds to 60 seconds, and we built I_{patch} with a fixed size of 256×256 as training samples. Here we built an equal number of positive and negative training images using over-sampling.

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

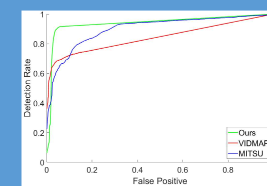
The testing dataset with HD and non-HD images for each artifact detection. Note that there are two types of sources: the **L** means the data are from our lab and are labelled by a video specialist, and the **S** means the data are the synthesized images as described in II-B.

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

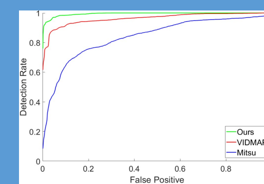
Upscaling



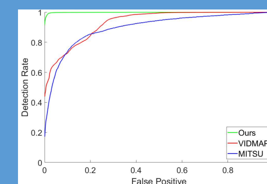
Interlacing



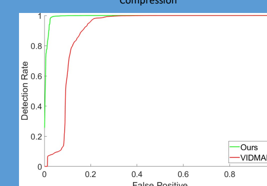
H264 hits



Mpeg2 hits



Compression



The ROC curves of our algorithm, the state of art DNN algorithm (VIDMAP) and traditional MITSU algorithm

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.

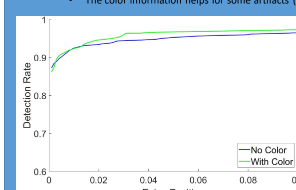
Artifact	FP	Ours	VIDMAP	MITSU
Up-scaling	5%	**92.0**	38	46.4
	10%	**94.4**	48.7	83
Combing	5%	**90.9**	68.5	63.1
	10%	**91.5**	72.0	71.4
H.264 hits	5%	**98.3**	86.3	75.6
	10%	**99.6**	88.47	80.9
MPEG-2 hits	5%	**99.2**	66.2	60.1
	10%	**99.8**	72.8	73.7
Compression	5%	**99.6**	9.4	N/A
	10%	**99.9**	62.4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

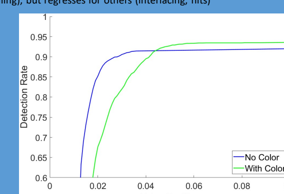
The running time in terms of milliseconds for our algorithm and the state of the art VIDMAP algorithm [13] when processing images with size of 1920×1080. The test machine is an AWS $P3\ 2X$ large instance with one NVIDIA Tesla V100-SXM2 GPU and an Intel(R) Xeon(R) CPU @2.30GHz.

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

Conclusions

- We presented a DNN framework to detect the NR metrics from an input image
- For training, we use 256x256 image patches.
- For testing, we use the full image.
- Our algorithm output performs the state of art of VIDMAP in terms of both accuracy and speed
- The color information helps for some artifacts (upscaling), but regresses for others (interlacing, hits)





Future Study

- Change the binary classifier into multiple classifier: one pass to predict multiple artifacts
- Change the super-pixel level output to pixel level output: FPN backbone

Contact us for more information at (joeliu, bferry)@amazon.com