Analysis Tools in the VMAF Open-Source Package

Zhi Li, Christos G. Bampis
Netflix

Video Quality Expert Group (VQEG) Meeting
Mountain View, CA, 11/13/2018
I have developed a machine-learning model to predict video quality, can I trust it?

How should I evaluate the performance of the model?

For a particular video, how much can I trust the score predicted by the model?

Which features / elementary metrics contributed the most to the prediction?
Overtime, we’ve incorporated some helper tools into the VMAF package...

VMAF - Video Multi-Method Assessment Fusion

VMAF is a perceptual video quality assessment algorithm developed by Netflix. VMAF Development Kit (VDK) is a software package that contains the VMAF algorithm implementation, as well as a set of tools that allows a user to train and test a custom VMAF model. For an overview, read this tech blog post, or this slide deck.

News

- (10/25/18) We have published our second techblog on VMAF, with recommendations on best practices.
- (9/13/18) SUREAL is no longer a submodule to VMAF.
- (6/19/18) Each VMAF prediction score now comes with a 95% confidence interval (CI), which quantifies the level of confidence that the prediction lies within the interval.
- (6/19/18) Added a 4K VMAF model under model/vmaf_4k_v0.6.1.pkl, which predicts the subjective quality of video displayed on a 4K TV and viewed from the distance of 1.5X the display height.
- (6/5/18) Speed optimization to vmafossexec : 1) support multi-threading (e.g. use ---thread 0 to use all cores), 2) support frame sampling (e.g. use ---subsample 5 to calculate VMAF on one of every 5 frames).
Tools in the Repo besides VMAF

- Metrics implementation - elementary metrics & benchmark
  - SSIM & MS-SSIM (Wang et al.)
  - BRISQUE & NIQE (Mittal et al.)
  - ST-MAD (Chandler et al.)
  - ST-RRED (Soundararajan et al.)
  - SpEED-QA (Bampis et al.)

- Subjective data clean up tools (Li & Bampis) — moved to SUREAL repo

- BD-rate calculator

- Performance Metrics beyond Pearson and Spearman
  - Resolving Power (Pinson & Wolf)
  - AUC - Area Under the RoC Curve (Krasula et al.)

- Local explainer (based on LIME by Ribeiro et al.)

- Confidence intervals via bootstrapping (Li & Bampis, work in progress)
Topics of This Talk

- Performance metrics beyond Pearson and Spearman
- Local explainer
- Confidence intervals via bootstrapping
PLCC and SROCC

- PLCC: Pearson Linear Correlation Coefficient
- SROCC: Spearman Rank Order Correlation Coefficient
- Limitations
  - Not consider variability in the raw subjective scores - only MOS
  - Do not give interpretation that is intuitive enough
  - Range-dependent
Resolving Power (Pinson & Wolf)

- Consider raw subjective scores’ variability
- Put scores in pairs; for each pair, pose as a detection problem
- Ask the question: how much score difference is required to determine if one video is significantly better than the other, with a 95% confidence?
  - e.g. R.P. 1.53 out of [1,5] — score difference required to claim video A is better than B with 95% confidence
- Report score difference in two scales
  - Subjective score scale [1, 5]
  - Quality metric scale e.g. [0, 100]

AUC - Area Under the ROC Curve (Lukas et al.)

- Consider raw subjective scores’ variability
- Put scores in pairs; for each pair, pose as a detection problem
- Characterize performance by area under the ROC curve (ROC AUC)
- Two steps
  - Different/similar analysis
  - Better/worse analysis

- ResPow
  - 23.379 - resolv. power in VMAF score scale (0 - 100)
  - 1.373 - resolv. power in subjective scale (1 - 5)
- AUC
  - 0.878 - different/similar (DS) AUC analysis
  - 0.992 - better/worse (BW) AUC analysis
Topics of This Talk

- Performance metrics beyond Pearson and Spearman
- Local explainer
- Confidence intervals via bootstrapping
Local Explainer - Motivation

- VMAF predicts video quality by fusing elementary metrics using a nonlinear regression (e.g. SVM)
- It is helpful to be able to interpret each elementary metric’s contribution to the final VMAF score
  - Something similar to a linear regressor will be nice, where the “weight” represents the importance
LIME - Local Interpretable Model-Agnostic Explanation

“Why Should I Trust You?”
Explaining the Predictions of Any Classifier

Marco Tulio Ribeiro
University of Washington
Seattle, WA 98105, USA
marcolcr@cs.uw.edu

Sameer Singh
University of Washington
Seattle, WA 98105, USA
sameer@cs.uw.edu

Carlos Guestrin
University of Washington
Seattle, WA 98105, USA
guestrin@cs.uw.edu

Model
Data and Prediction
Explaner (LIME)
Explanation
Human makes decision
Local Explainer - Intuitions

- Idea in a nutshell
  - “Linearize” a nonlinear classifier (C) / regressor (R) at a local instance
  - The coefficients of the linear C / R serves as the weight for each features

- In more detail
  - For a local instance (i.e. feature vector), sample in its neighborhood (Gaussian kernel), run the nonlinear C / R to get the labels of the samples
  - Train a linear C / R using the samples and their labels
Local Explainer - Applying to VMAF

- Explain default VMAF model v0.6.1 on an OldTownCross video

```
```
Local Explainer - Applying to BRISQUE

- Explain BRISQUE features

```
./run_vmaf yuv420p 1920 1080 NFLX_dataset_public/ref/OldTownCross_25fps.yuv
NFLX_dataset_public/dis/OldTownCross_90_1080_4300.yuv --local-explain --model
model/vmaf_brisque_all_v0.0rc.pkl
```
Topics of This Talk

● Performance metrics beyond Pearson and Spearman
● Local explainer
● Confidence intervals via bootstrapping
The Need for Bootstrapping

Pixel Neighborhood
- spatial feature extraction (VIF, DLM)
- temporal feature extraction (TI)

Frame Level
- within-frame spatial pooling

SVM prediction
- trained model
- "Fusion"

temporal pooling
- per-frame score
Bootstrapping - “Resampling with Replacement”

```python
import numpy as np

pop_size = 100000
sample_size = 1000
trials = 100

pop_mean = 5
pop_std = 11
population = np.random.randn(pop_size) * pop_std + pop_mean
sample = population[:sample_size]

means_pop = [np.mean(np.random.choice(population, size=sample_size, replace=True)) for _ in range(trials)]
means_bootstrap = [np.mean(np.random.choice(sample, size=sample_size, replace=True)) for _ in range(trials)]

stds_pop = [np.std(np.random.choice(population, size=sample_size, replace=True)) for _ in range(trials)]
stds_bootstrap = [np.std(np.random.choice(sample, size=sample_size, replace=True)) for _ in range(trials)]

print('std of sample mean: {} (ground truth)'.format(np.std(means_pop)))
print('std of sample mean: {} (bootstrapped)'.format(np.std(means_bootstrap)))

print('std of sample stdev: {} (ground truth)'.format(np.std(stds_pop)))
print('std of sample stdev: {} (bootstrapped)'.format(np.std(stds_bootstrap)))
print('Done.')
```

Bootstrapping on Training Videos

Train Data → Bootstrap → Train → Predict → Test Data → Var → C.I.

N: # train videos, X: N x 6 feature matrix, y: N x 1 label vector
Bootstrapping on Training Videos (Cont’d)

BOOTSTRAP_VMAF
(SRCC: 0.943, PCC: 0.940, RMSE: 12.733)

* 95% C.I., Bootstrapping based on 20 models
Subjective Bootstrapping

- Training videos can be different; but subjects can be as well
- How can we capture this subjective variability in VMAF predictions?
- Let Ns be the number of subjective bootstrap models
- For each bootstrap iteration:
  - Sample subjects (allow repetition)
  - For each train video, eliminate scores from subjects not selected
  - For each train video, repeat scores for subjects that were selected more than once
Toy Example

4 videos and 3 subjects: Tom, Jerry and Anna

3 example bootstrap sets: [Tom, Jerry, Tom], [Anna, Anna, Anna] and [Jerry, Anna, Jerry]

<table>
<thead>
<tr>
<th></th>
<th>Tom</th>
<th>Jerry</th>
<th>Anna</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0</td>
<td>5</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>#1</td>
<td>2</td>
<td>1</td>
<td>4</td>
</tr>
<tr>
<td>#2</td>
<td>3</td>
<td>5</td>
<td>1</td>
</tr>
<tr>
<td>#3</td>
<td>4</td>
<td>3</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>MOS</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>3.67</td>
</tr>
<tr>
<td></td>
<td>2.33</td>
</tr>
<tr>
<td></td>
<td>3.00</td>
</tr>
<tr>
<td></td>
<td>3.00</td>
</tr>
</tbody>
</table>
Toy Example - cont’d

● For each bootstrap set, determine the new MOS vector (labels)

<table>
<thead>
<tr>
<th></th>
<th>Tom</th>
<th>Jerry</th>
<th>Tom</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>#0</td>
<td>5</td>
<td>3</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>#1</td>
<td>2</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>#2</td>
<td>3</td>
<td>5</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>#3</td>
<td>4</td>
<td>3</td>
<td>4</td>
<td></td>
</tr>
</tbody>
</table>

● Retrain VMAF using the new labels

<table>
<thead>
<tr>
<th></th>
<th>MOS</th>
</tr>
</thead>
<tbody>
<tr>
<td>#0</td>
<td>4.33</td>
</tr>
<tr>
<td>#1</td>
<td>1.67</td>
</tr>
<tr>
<td>#2</td>
<td>3.67</td>
</tr>
<tr>
<td>#3</td>
<td>3.67</td>
</tr>
</tbody>
</table>
Subjective Bootstrapping Results

Subjective variability tends to produce a lower CI than training video variability.
Coupled Bootstrapping

- Combine the two bootstrapping approaches
- Account for both training video and subjective variability
The combined effect of training video and subjective variability increases the CI.
Final Remarks

- We want to have better understanding of ML models trained to predict quality

- We have incorporated a set of helper tools to develop such understanding
  - Performance metrics: resolving power and AUC
  - Local explainer
  - Bootstrapping for prediction confidence interval

- We invite researchers to use our tools and also contribute new tools
New Techblog on VMAF

VMAF: The Journey Continues

by Zhi Li, Christos Bampis, Julie Novak, Anne Aaron, Kyle Swanson, Anush Moorthy and Jan De Cock

How will Netflix members rate the quality of this video—poor, average or excellent?
Which video clip looks better—encoded with Codec A or Codec B?
For this episode, at 1000 kbps, is it better to encode with HD resolution, with some blockiness, or will SD look better?
- Adaptive media streaming, content storage, and content delivery
- Novel technologies for interactive audiovisual communications
- Next-generation/future video coding, point cloud compression
- Cloud and P2P based multimedia
- Video streaming over software-defined networks
- Multimedia communications over future networks, such as information-centric networks next-generation 802.11ax networks and 5G wireless
- Coding and streaming of immersive media, including virtual reality (VR), augmented reality (AR), 360° video and multi-sensory systems
- Machine learning in media coding and streaming systems
- Standardization: DASH, MMT, CMAF, OMAF, MiAF, WebRTC, HTTP/2, QUIC, MPTCP, MSE, EME, WebXR, Hybrid Media, WAVE, etc.
- Emerging applications: social media, game streaming, personal broadcast, healthcare, industry 4.0, multi-camera surveillance, smart transportation, etc.

Submission deadline: February 10, 2019
Acceptance notification: March 22, 2019
Camera-ready deadline: April 7, 2019

https://2019.packet.video

PACKET VIDEO WORKSHOP 2019

June 18, 2019, Amherst, MA, USA (co-located with ACM MMSys’19)
Questions ?