Analysis and development of quality assessment methods of 360 videos

Mohammad Saleh Dehqanpour

Supervisor: Dr.Sharifkhani

Feb. 2021

Overview

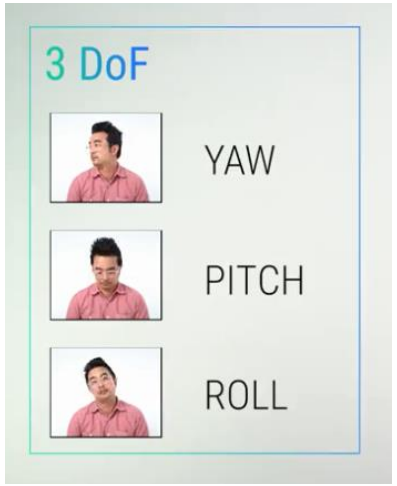
- Introduction
- Previous works for 360° Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

Overview

- Introduction
- Previous works for 360° Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

360° Videos

- 360 degree horizontal (longitude) and 180 degree vertical (latitude)
- 3-degrees of freedom
- Processing Pipeline



Video Quality Assessment (VQA)

- Goal
  - Report a number indicating how well a video looks to a typical user
- Applications
  - Video coding
  - Video recovery
  - User experience assessment
  - Manufacturing
  - etc.

Video Quality Assessment (VQA)

- Two approaches:
  - Subjective assessment
    - Conduct a standard experiment with users
    - Process the reported scores
    - Report the result as “MOS” or “DMOS”
  - Objective assessment
    - Categories
      - **Full-Reference** ←
      - Reduced-Reference
      - No-Reference
    - Given a test video and its reference one, predict users’ scores
    - Some classic methods: PSNR, SSIM
    - Very diverse & advanced methods (VQM, MOVIE, …)

Overview

- Introduction
- Previous works for 360 Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

Previous works for 360° Video VQA: Approaches & Ideas

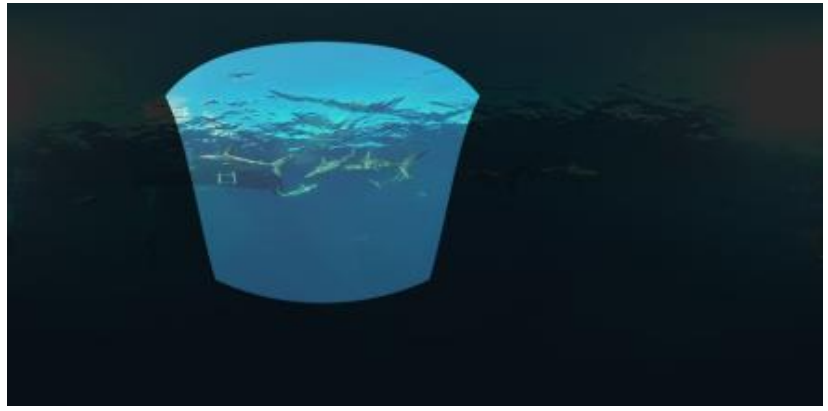
- PSNR-based methods
  - Based on assigning weight to pixels in different ways
  - Examples:
    - WS-PSNR
    - S-PSNR-I / S-PSNR-NN
    - CPP-PSNR
    - NZ-SPSNR
    - VASW-PSNR
    - NCP-PSNR / CP-PSNR
    - etc.

Previous works for 360° Video VQA: Approaches & Ideas (Cont.)

- Using other 2D metrics along with assigning weights
  - If calculation is not on pixel level, apply them on small blocks or patches
  - Examples:
    - NCP-SSIM / CP-SSIM
- Weight maps using a saliency prediction model
- CNN/DNN methods on extracted patches
- Adversarial learning networks

Inaccurate methods

Require large scale datasets



Recent Methods for 360° Video VQA

- Pano
  - Perception-based by considering 3 new factors
    - Relative head-target speed
    - Relative luminance changes
    - Depth of field difference of target to its vicinity
- VMAF on ERP
  - Applying VMAF method on entire ERP plane
- Viewport-Driven Multi-Metric Fusion Approach
  - Patch-based (40 degree field of view chosen)
  - Not Perceptual
  - Computationally complex

Overview

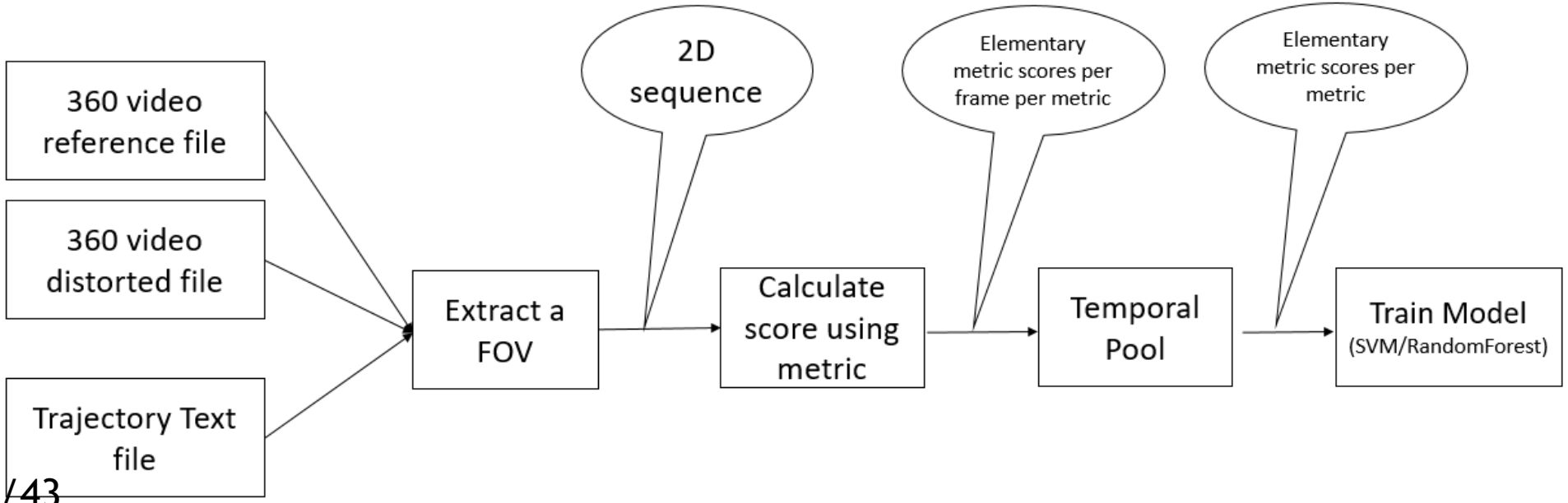
- Introduction
- Previous works for 360° Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

Proposed VQA Method

- Goal: design and assess a full-reference video quality assessment method for 3-DoF 360-degree video contents
- General ideas:
  - Fusion of several base metrics
    - The same idea which have been used successfully in other domains
  - Reduce the video from 360 plane to a traditional video. So:
    - Our inputs are the same as what users have seen
    - Unlock the richness of traditional video metrics
  - Take subject behaviors into account
    - To be “perception-based”
    - Achieve high correlation

Train Phase (1/3)

1. Inputs
    1. Reference and test panoramic videos
    2. Viewers' watching trajectories
2. Extract a rough field of view according to trajectory of each viewer
    - Using mapping calculations (360Lib)
    - Using OpenGL (setting camera)



Viewport Extraction: Reducing Subjects

- We have 48 subjects (so trajectories) for each 360 video.
- Large number => impractical calculations (specially in-the-field)
- There are at least two ways to select a subset:
  - Randomly select from them.
  - Devise an algorithm to choose most useful subset from them. ✓
- We will report the results for both cases but the latter is preferred.

Subject Selection Method

- Define the distance of two subjects as the MSE of their trajectories
- For each 360 video, select a subset of 20 subjects from a total of 48 ones by choosing the farthest ones from each other.
  - So we avoid useless calculations for those subjects who probably watched video like the selected ones.
- The problem known as “discrete p-dispersion” :
  - Select a subset of p nodes from n nodes and maximize the distance between them
- A rough solution called 2-factor approximation was used.
- After determining the selected subjects, we take the effect of removed ones into account by:
  - Assigning a weight to each selected subject. It is equal to number of subjects it represent (i.e. those who watched video similar to it)
  - Example:
    [1,1,1,0,0,0,0,12,0,0,0,1,0,1,0,0,1,0,1,5,0,1,6,0,1,0,0,1,0,0,1,1,0,0,0,0,3,0,1,0,0,0,0,0,
    1,4,4]

Viewport Extraction: Choosing FoV Size

- We should choose a proper FoV degree
  - It determines viewport resolution
- Ideally should be equal to viewer's headset FoV
- But the more FoV we choose, the more file size, metric calculation time, …

- By conducting a simple experiment and allowing an error margin, we found:
  - No significant difference between 80 and 110 degrees

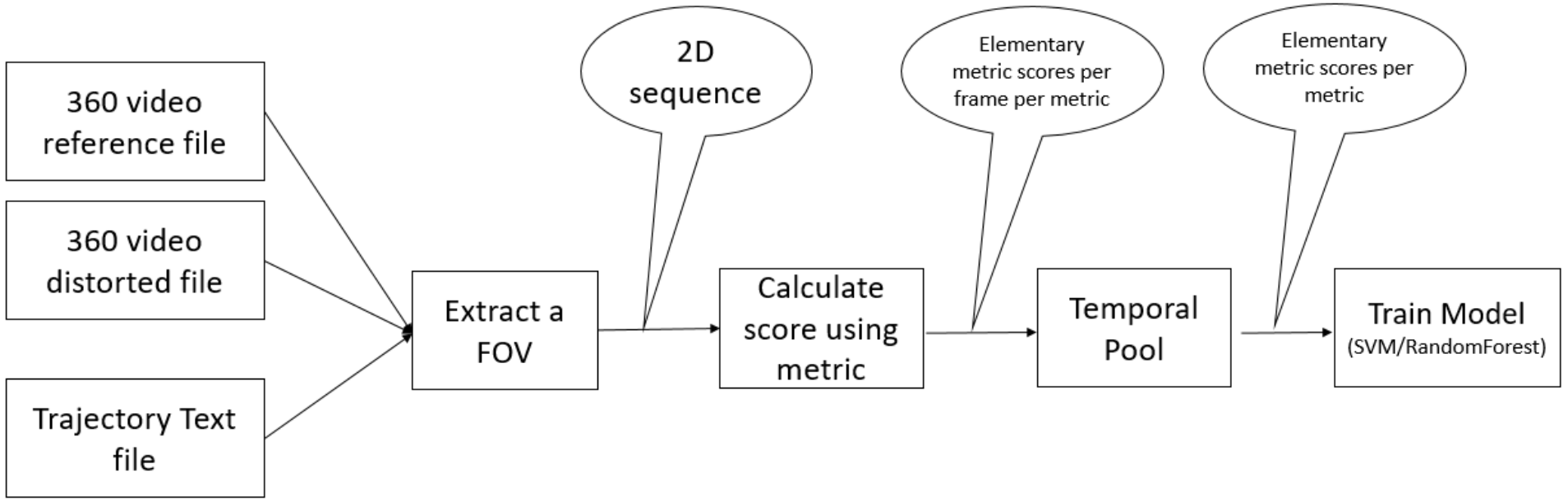
- Therefore extracted viewports has a FoV of 80 degree and a resolution of 910x910

Viewport Extraction: Choosing test or reference trajectory

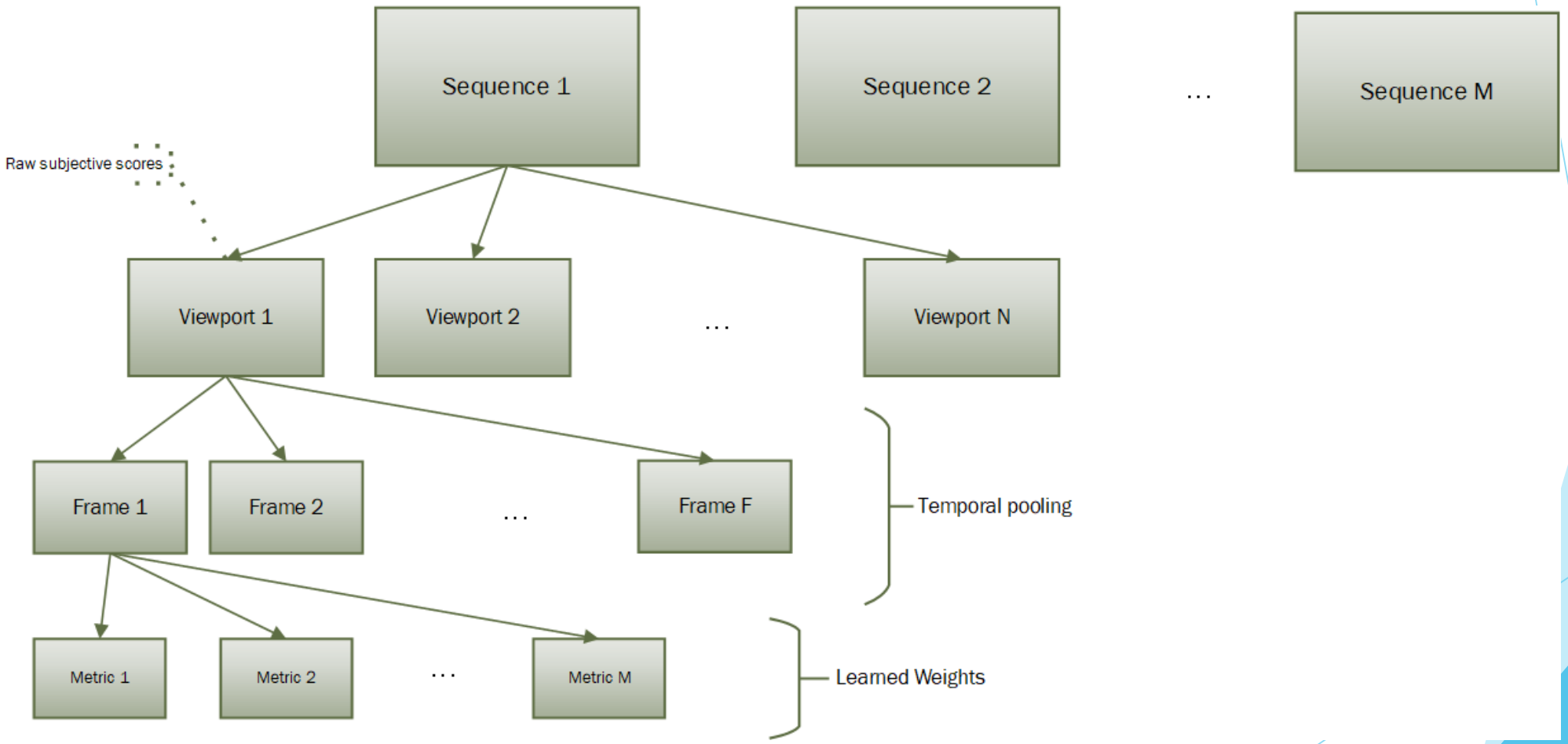
- For each subject and each 360 video, we should choose either reference or test video trajectory to extract both videos according to it.
  - Otherwise the two videos will not match to same region
- If we choose test video:
  - Advantage: More reasonable and valid
    - because we assume the reference video is a perfect one and reported score deficiency was due to the test video
  - Disadvantage: More processing
    - We should extract a new trajectory from reference video for each test video

Train Phase (2/3)

3. Calculate each elementary metric frame-by-frame on extracted videos
   - Just as we do in a traditional setup
4. Temporal pool each metric to achieve a single number for each 2D sequence
   - We simply used arithmetic mean.
   - Thus we have a score for each video and each metric.

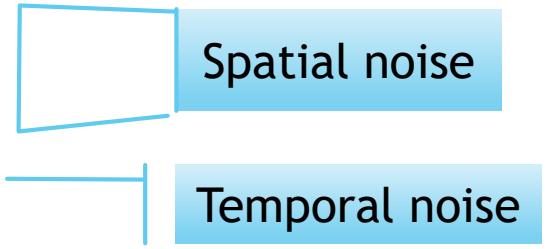


Metric Calculation Hierarchy



Elementary Quality Metrics

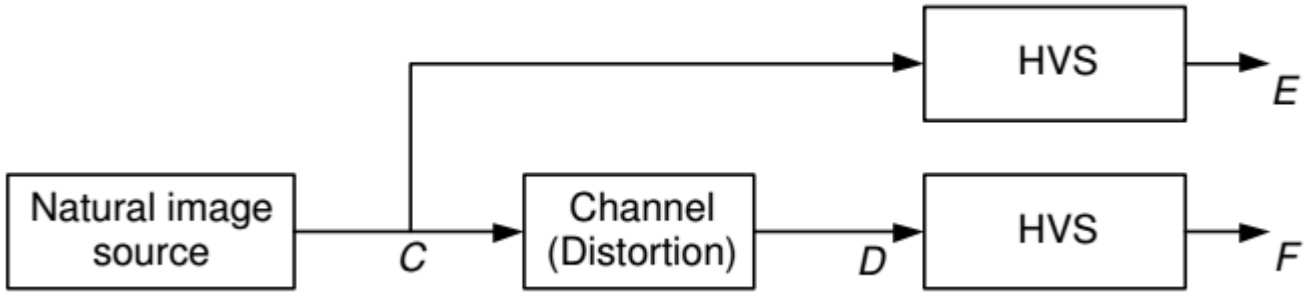
- These metrics was calculated as part of our VQA:
  - Visual Information Fidelity (VIF) at four scales
  - Detail Loss Metric (DLM)
  - Mean Co-Located Pixel Difference (MCPD)



- Also these metrics was calculated for the sake of comparison:
  - PSNR (For each of 3 color planes: YCbCr)
  - SSIM
  - MS-SSIM (at 5 scales)

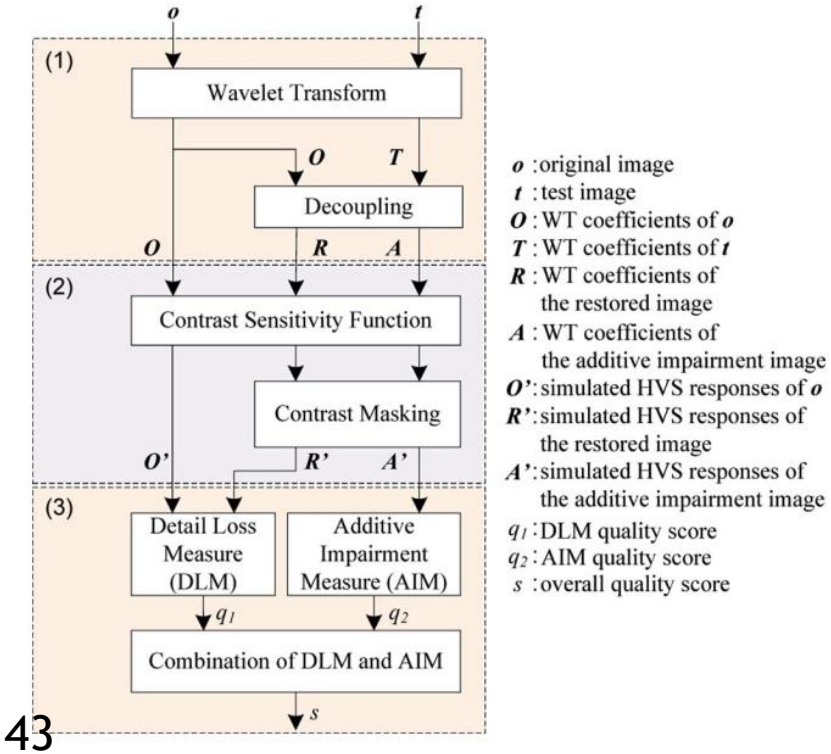
Visual Information Fidelity (VIF)

- An information fidelity criterion
- Assumes all reference videos signals follow Natural Scene Statistics (NSS) modelling.
- It quantifies shared Shannon information between the reference and the distorted images *relative* to the information contained in the reference image itself.
- Roughly calculated as $\frac{I(C;F)}{I(C;E)}$ considering this figure's labels:

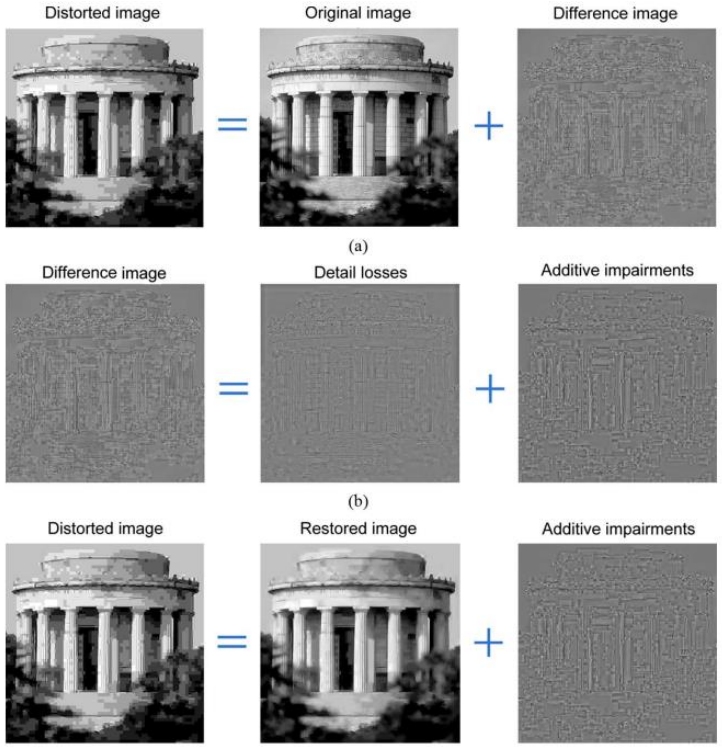


Detail Loss Metric (DLM)

- General Idea: Two types of loss which cause degradation to user:
  - Loss of details which affects the content visibility: DLM
  - Redundant impairment which distracts viewer attention: addItive Impairment Measure (AIM)



o : original image
t : test image
O : WT coefficients of *o*
T : WT coefficients of *t*
R : WT coefficients of the restored image
A : WT coefficients of the additive impairment image
O' : simulated HVS responses of *o*
R' : simulated HVS responses of the restored image
A' : simulated HVS responses of the additive impairment image
q_1 : DLM quality score
q_2 : AIM quality score
s : overall quality score

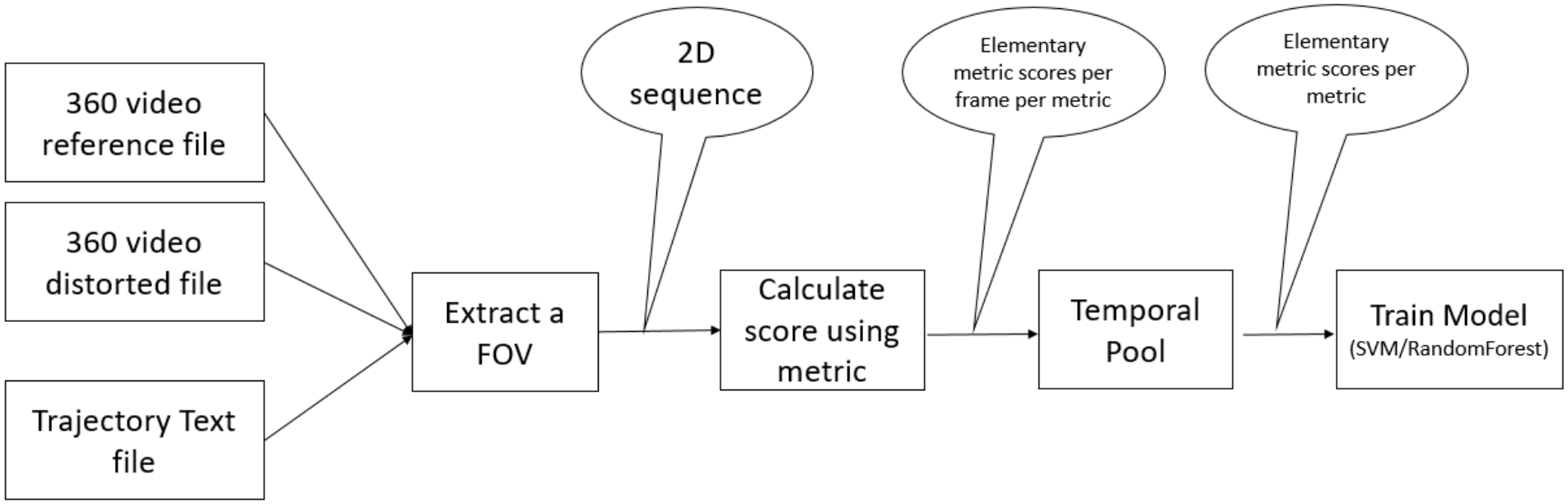


Mean Co-Located Pixel Difference (MCPD)

- Average absolute pixel difference for the luminance component
- A simple measure of the temporal difference between adjacent frames
- To take temporal noises into account

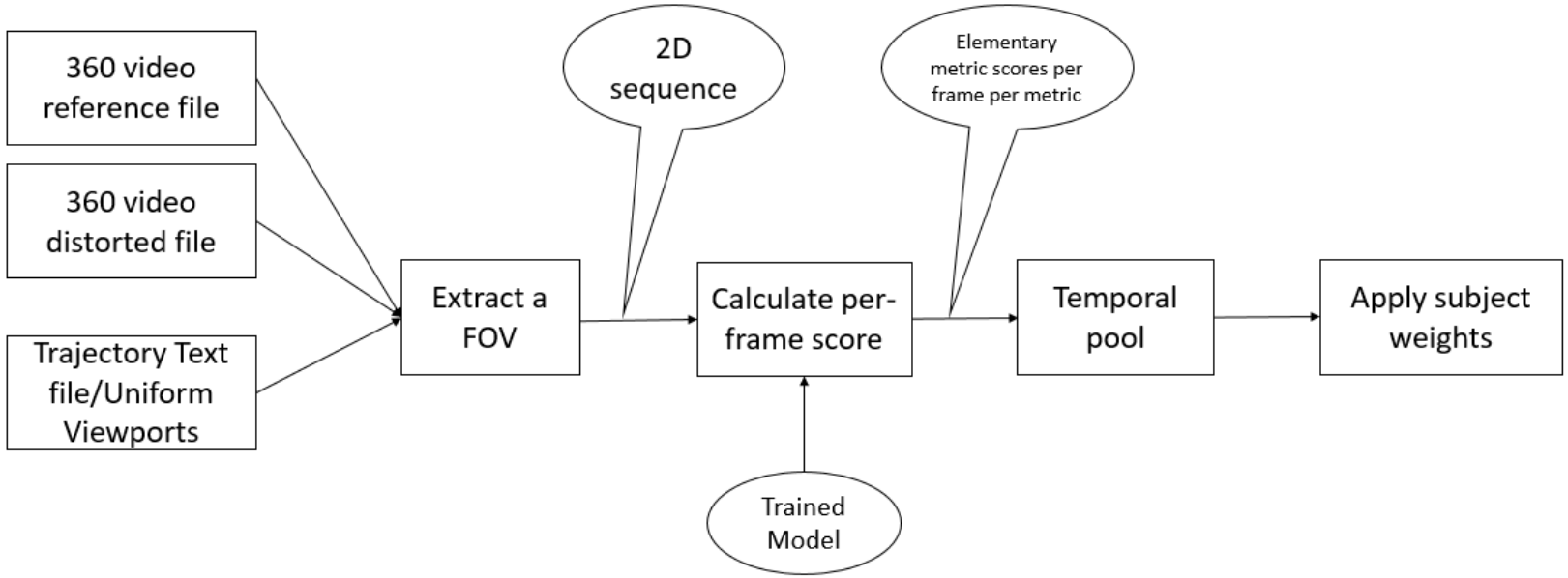
Train Phase (3/3)

5. Train model using computed values
   - Using mean DMOS of each 360 video to label its derived 2D videos
   - Apply the subjects' weights by repeating the data (proportional to the weight)
   - We used random forest with 10 estimators
     - Also tried SVR but does not yield better results
   - 75/25 train test split ratio



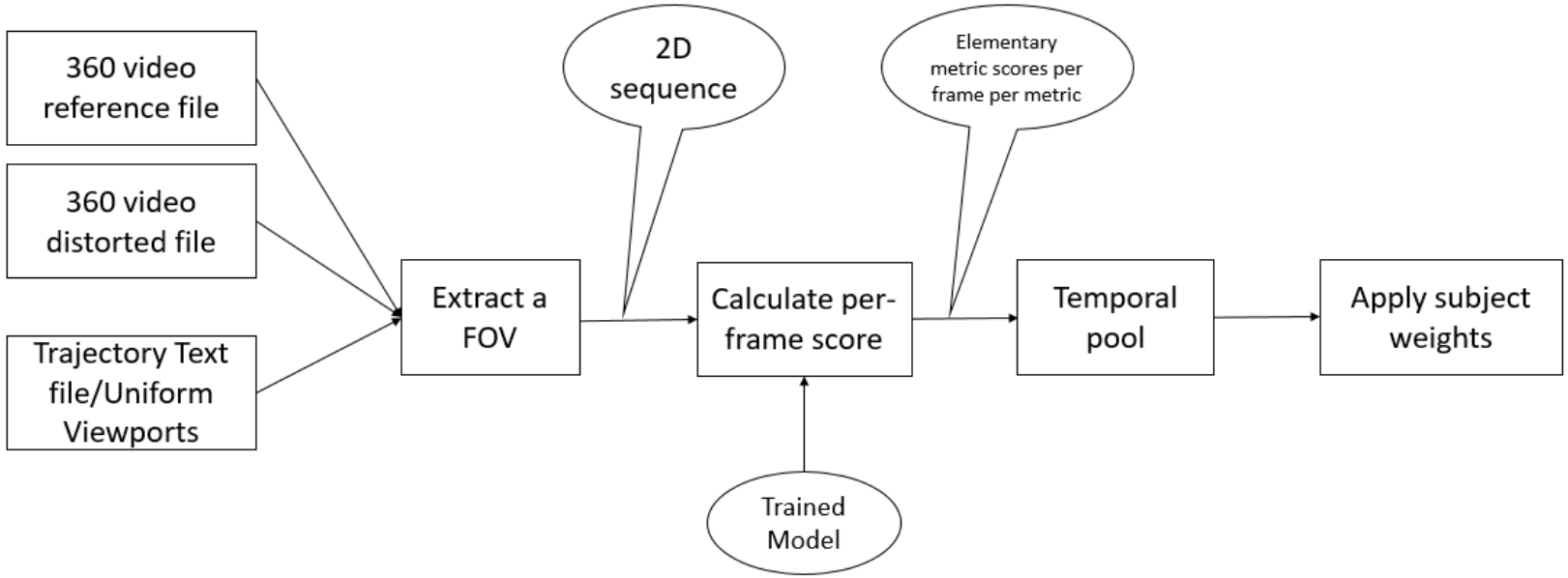
Test Phase (1/2)

1. Inputs:
    1. A pair of reference/test video
    2. Subjects' trajectory
        - Use uniform viewports if not available
2. Extract FoV (just as we do in train phase)



Test Phase (2/2)

3. Apply the model on a **per-frame** basis
   - For each frame, compute all metrics and aggregate using the trained model
4. Do temporal pooling so we have a single number for each extracted viewport of 2D video
5. Finally calculate a weighted sum among all derived 2D videos of the 360 video to achieve the quality score.



Overview

- Introduction
- Previous works for 360° Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

Dataset

- Extra requirement for dataset:
  - Viewers' head trajectory
- We use dataset from this paper
  - Xu, Mai, Chen Li, Zhenzhong Chen, Zulin Wang, and Zhenyu Guan. "Assessing visual quality of omnidirectional videos." IEEE Transactions on Circuits and Systems for Video Technology 29, no. 12 (2018): 3516-3530.
- Available at: https://github.com/Archer-Tatsu/head-tracking
- Properties
  - 48 video from 12 contents
  - 3 distorted version for each reference video (coded with QP values of 27,37,42)
  - Resolution 4096x2048 / 300 frames
  - 48 subjects
  - Each subject watched each video

Implementation

1. Reference videos was in YUV format. Distorted ones was encoded with H.264 and stored in MP4 container.
2. For distorted videos, first we converted from MP4 to YUV.
3. Then we chose 20 subjects and extracted test and reference videos for each of them. This resulted in 2*36*20=1440 2D videos (~600GB of storage)
4. Then we ran the libvmaf on all of them to calculate base metrics and store the results into text files.
5. With above results, the training process (with several settings like learners, …) was performed.
6. The test process was done and the graphs obtained.

On server

On laptop

- The above cycle was done two times.
  - First with 20 random subjects.
  - Second with selected subjects using aforementioned algorithm.

Overview

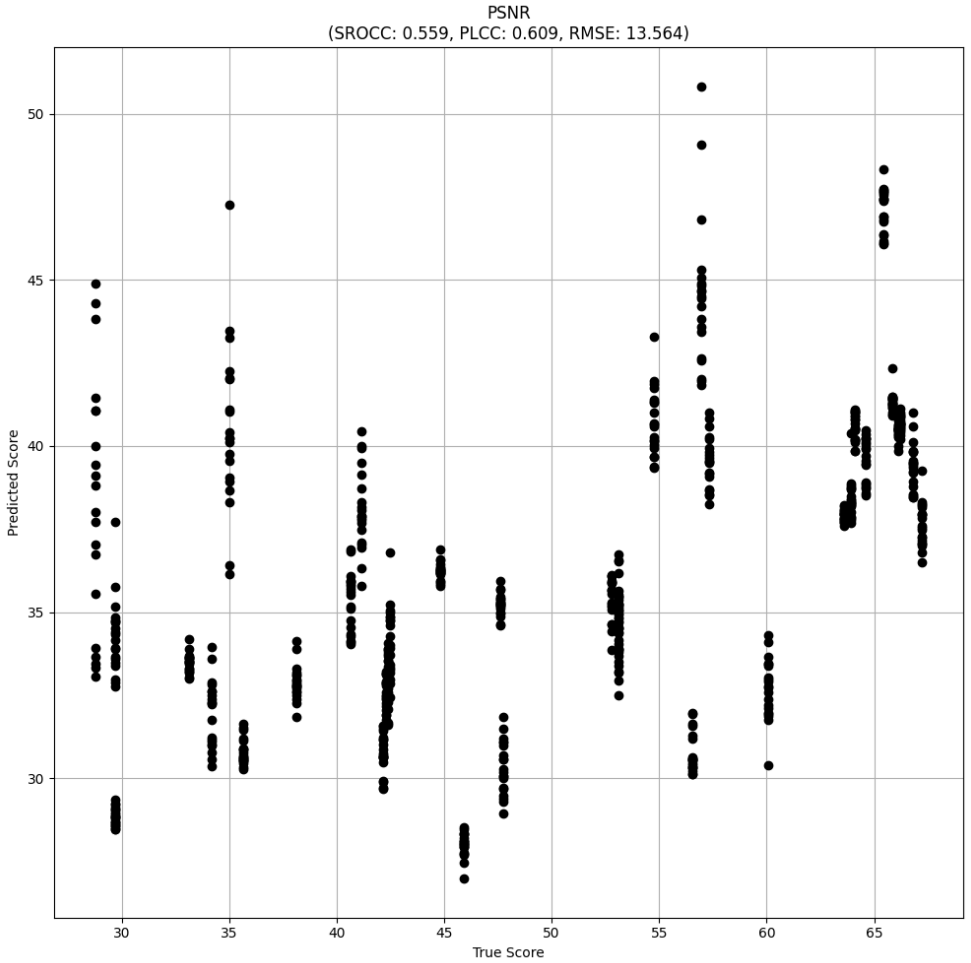
- Introduction
- Previous works for 360 Videos Quality Assessment
- Proposed VQA Method
- Implementation
- Achieved Results

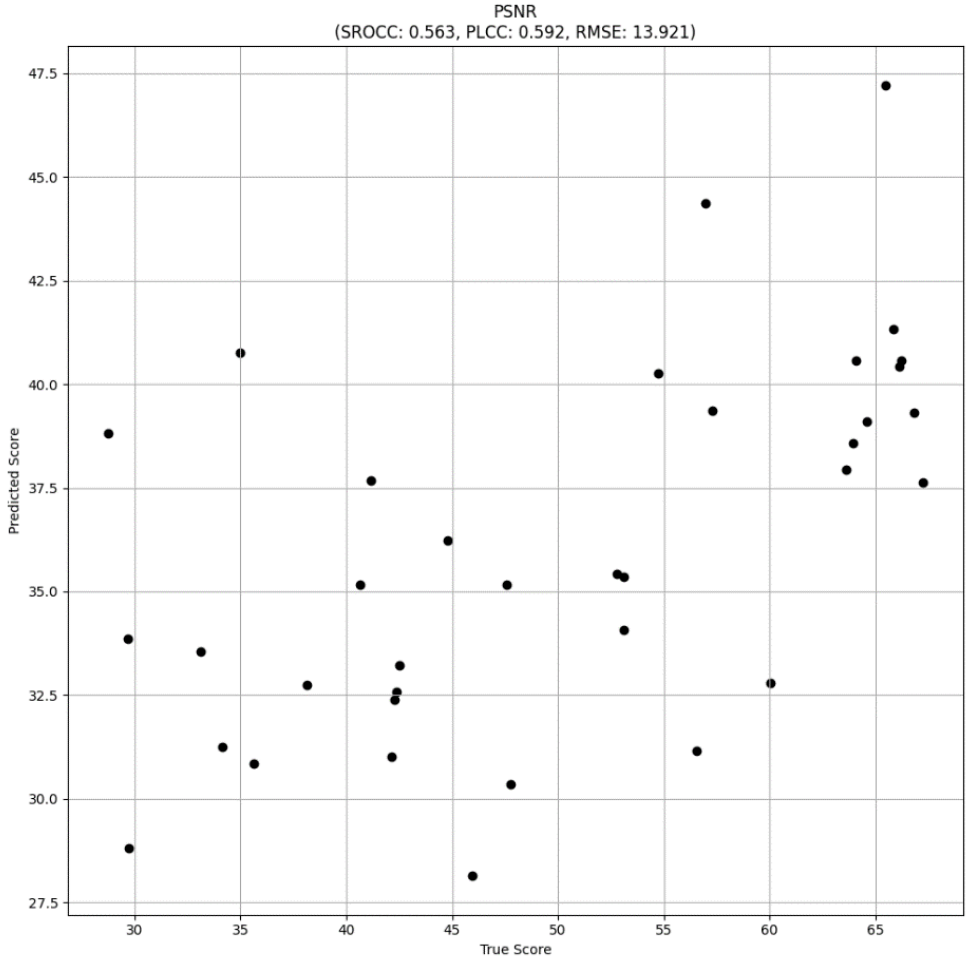
Comparison Considerations

- Dataset inconsistency problem
  - It is not valid to compare scores of two methods if the underlying datasets were not identical.
- We'll compare our results versus:
  - The referenced paper method which the dataset was picked from.
    - It has proposed 4 VQA methods: NCP-PSNR/CP-PSNR/NCP-SSIM/CP-SSIM
  - Applying classic metrics (PSNR, SSIM, MS-SSIM)

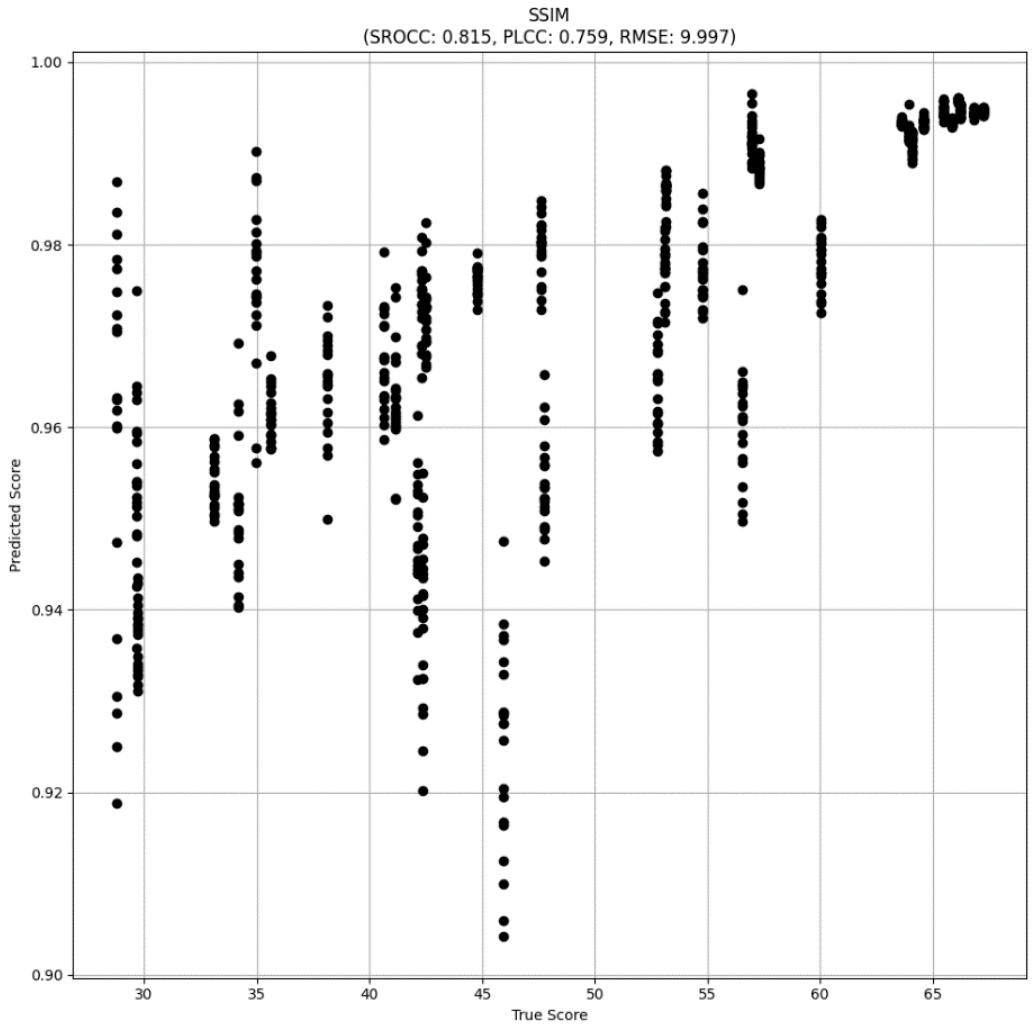
- In the following slides, we present the results when applying subject weights.

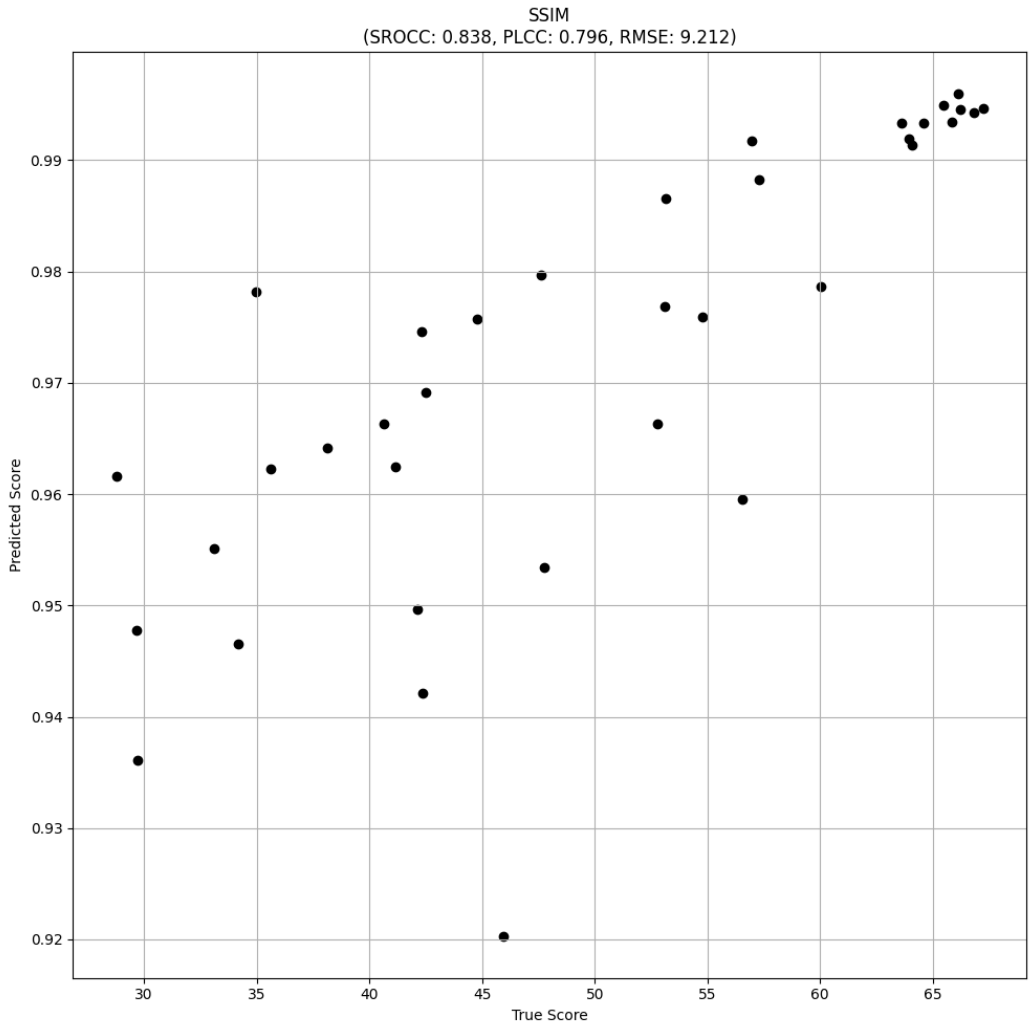
PSNR


PSNR
(SROCC: 0.559, PLCC: 0.609, RMSE: 13.564)
Predicted Score
True Score
50
45
40
35
30
30
35
40
45
50
55
60
65

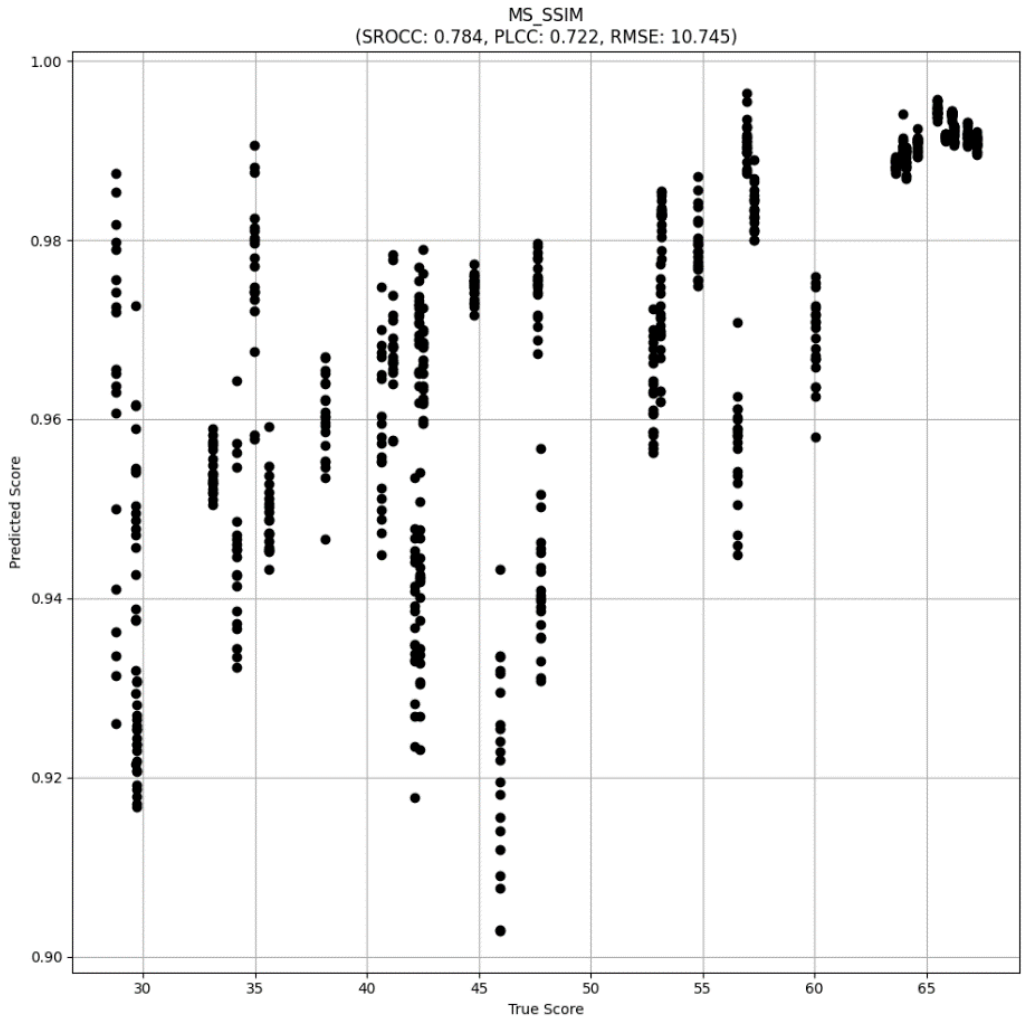

PSNR
(SROCC: 0.563, PLCC: 0.592, RMSE: 13.921)
Predicted Score
True Score
47.5
45.0
42.5
40.0
37.5
35.0
32.5
30.0
27.5
30
35
40
45
50
55
60
65

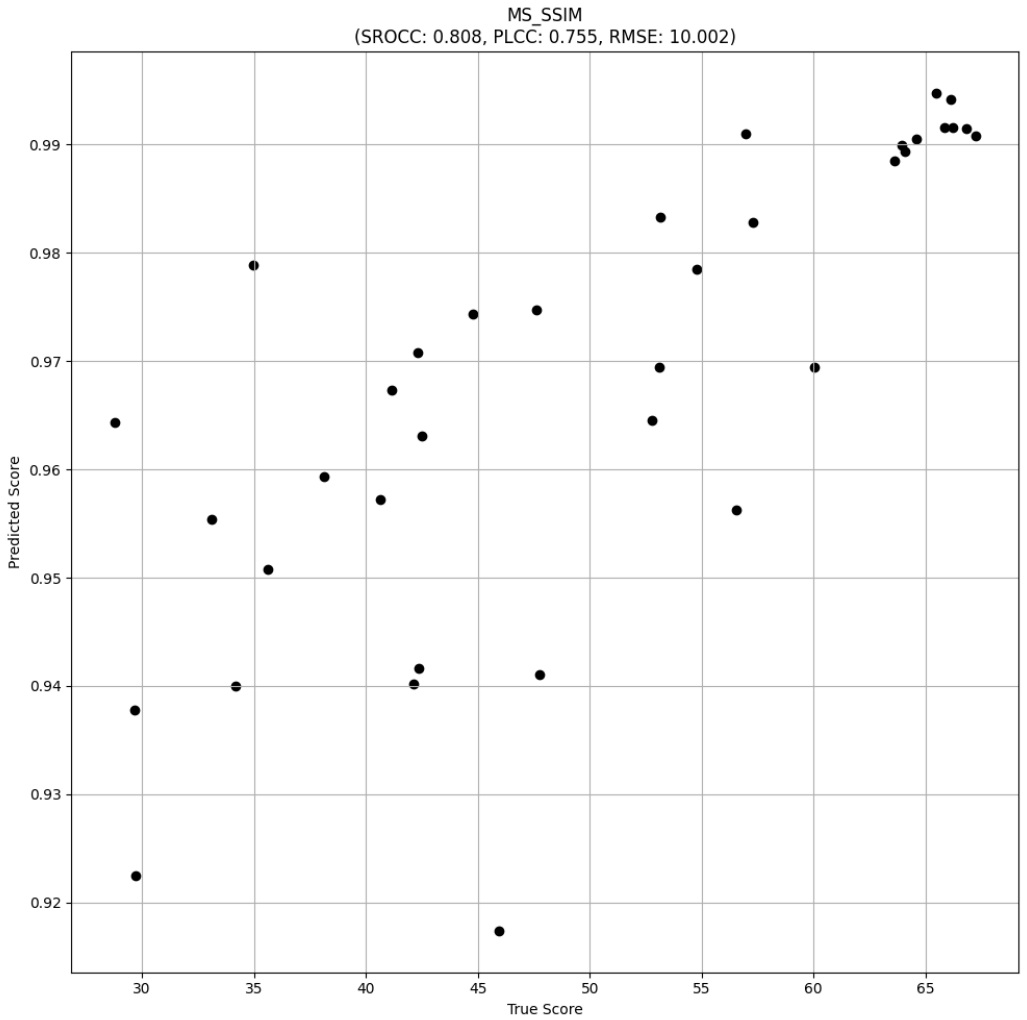
SSIM


SSIM
(SROCC: 0.815, PLCC: 0.759, RMSE: 9.997)
Predicted Score
True Score
1.00
0.98
0.96
0.94
0.92
0.90
30
35
40
45
50
55
60
65


SSIM
(SROCC: 0.838, PLCC: 0.796, RMSE: 9.212)
Predicted Score
True Score
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
30
35
40
45
50
55
60
65

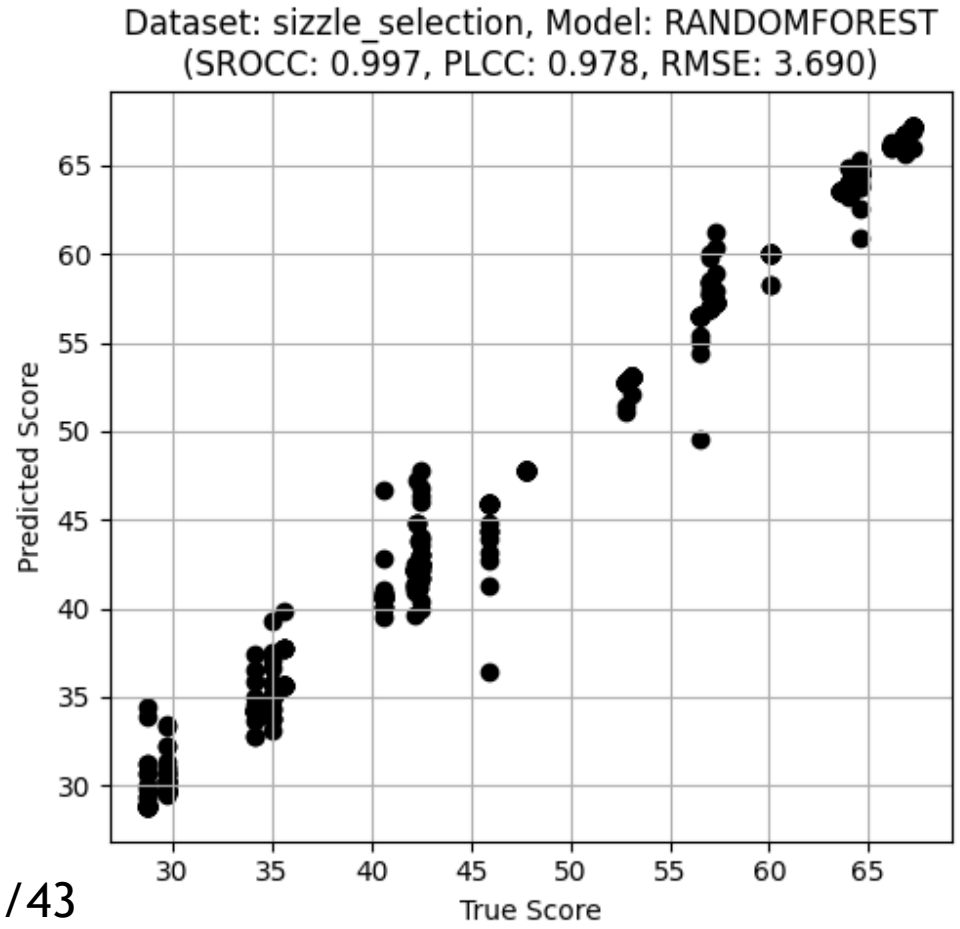
MS-SSIM

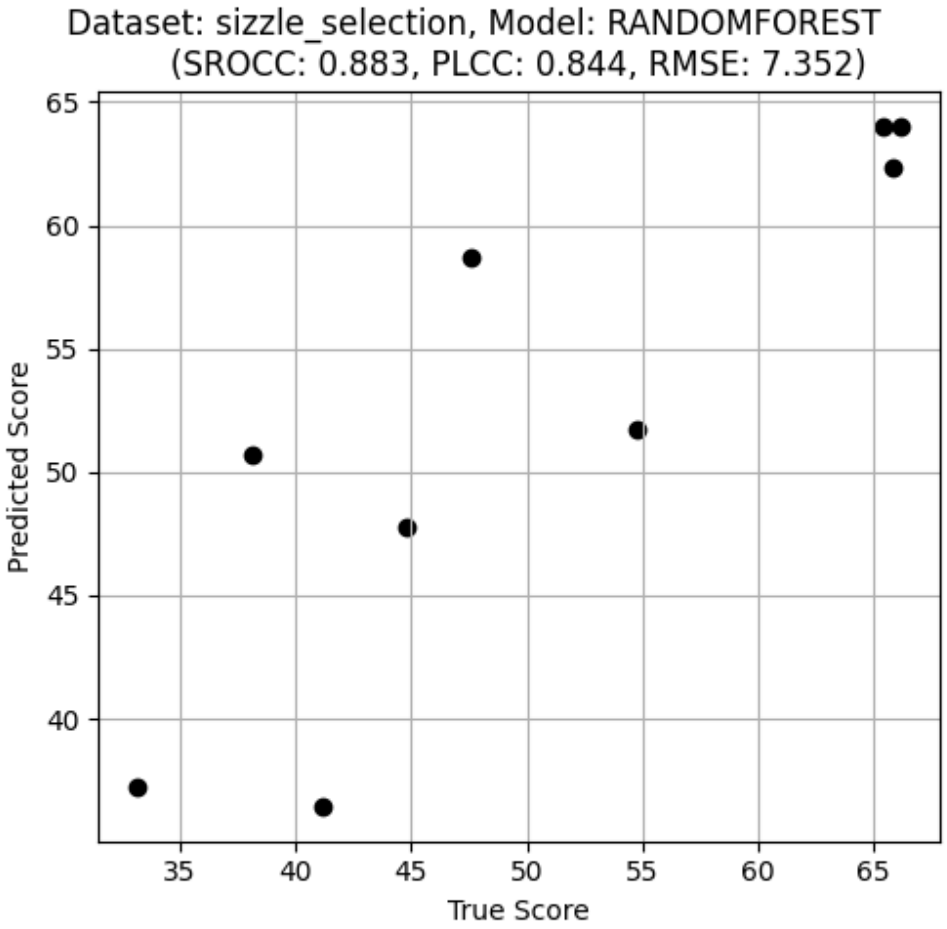

MS_SSIM
(SROCC: 0.784, PLCC: 0.722, RMSE: 10.745)
Predicted Score
True Score
1.00
0.98
0.96
0.94
0.92
0.90
30
35
40
45
50
55
60
65


MS_SSIM
(SROCC: 0.808, PLCC: 0.755, RMSE: 10.002)
Predicted Score
True Score
0.99
0.98
0.97
0.96
0.95
0.94
0.93
0.92
30
35
40
45
50
55
60
65

Composite Method Using 3 Base Metrics

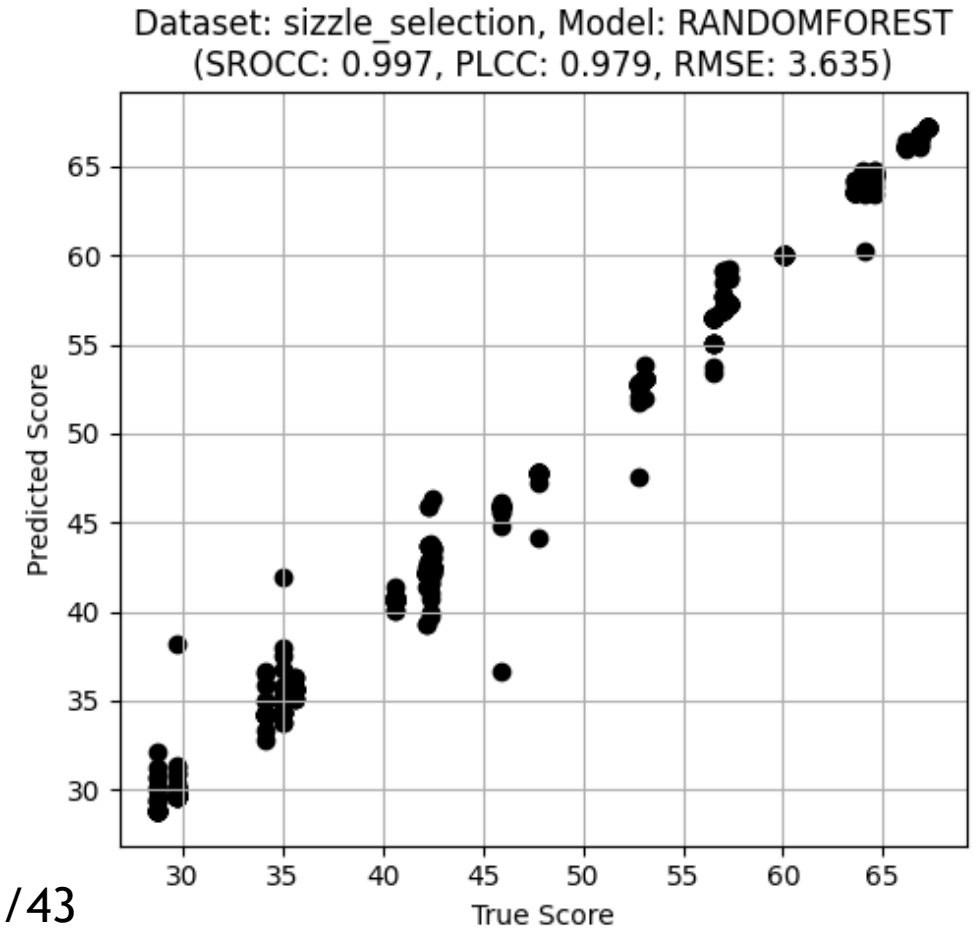
- 3 Metrics was used (VIF, DLM, MCPD)

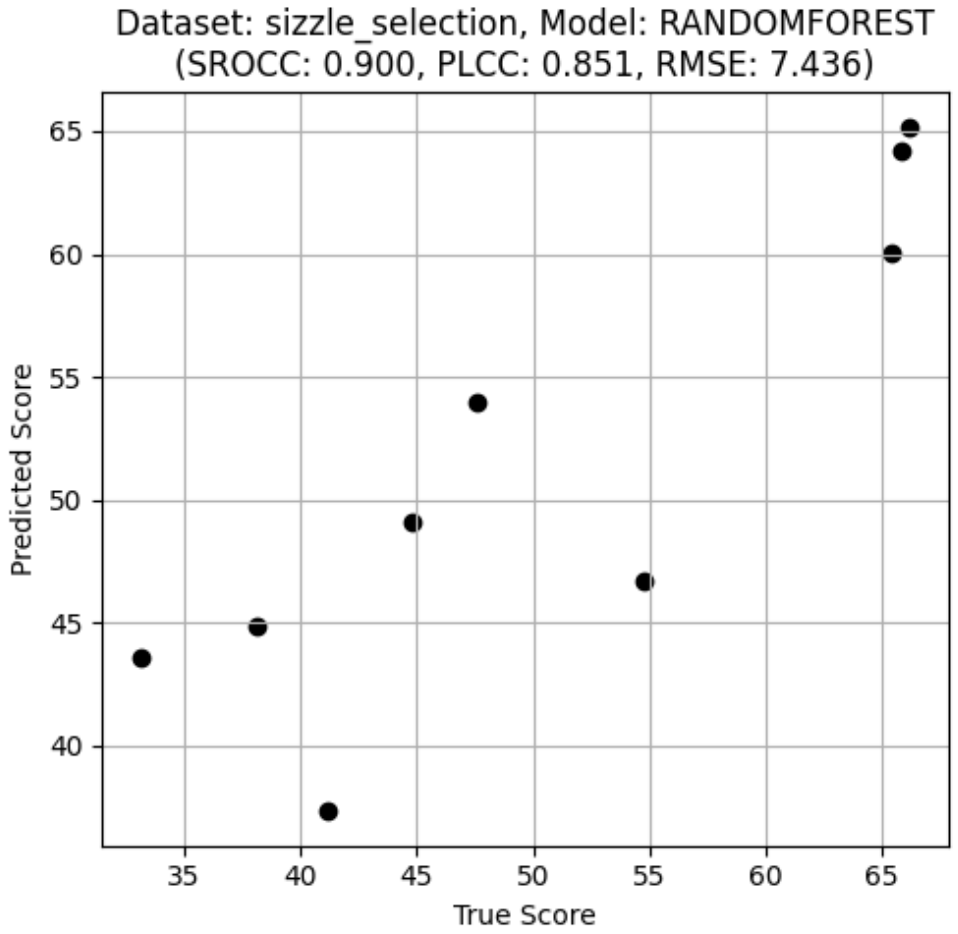




Composite Method Using 5 Base Metrics

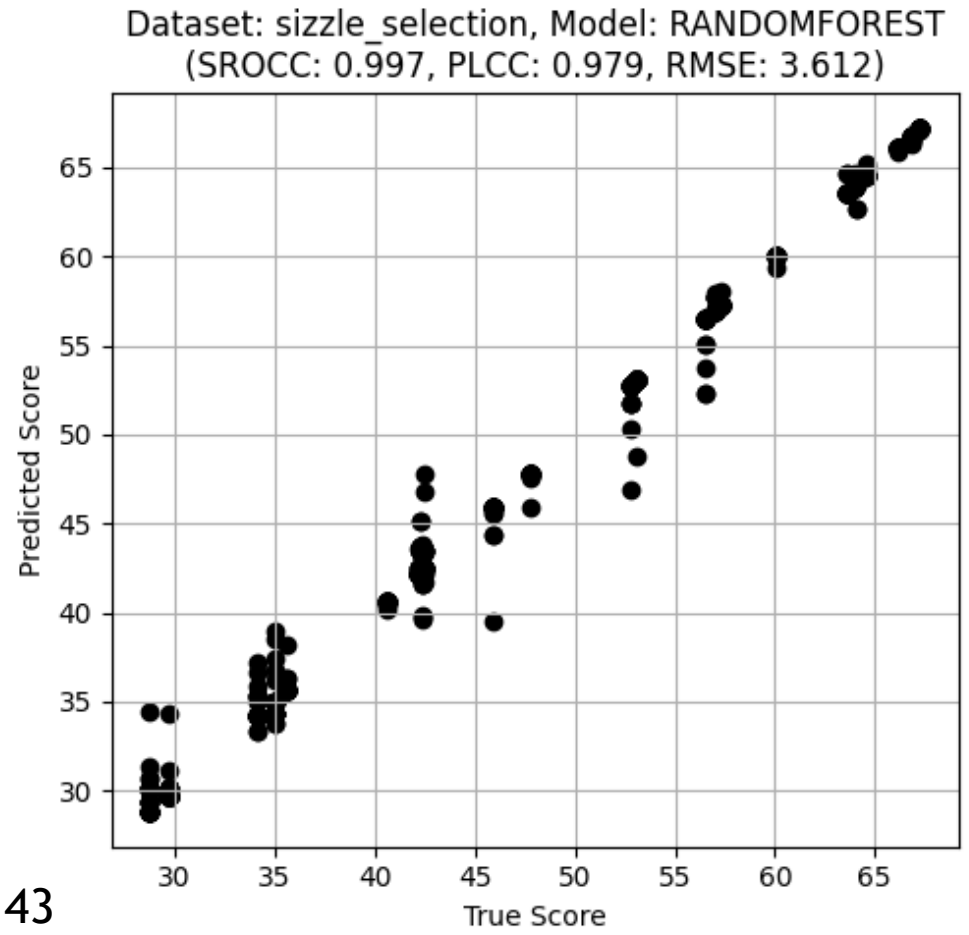
- 5 Metrics was used (VIF, DLM, MCPD, PSNR-Y, SSIM)

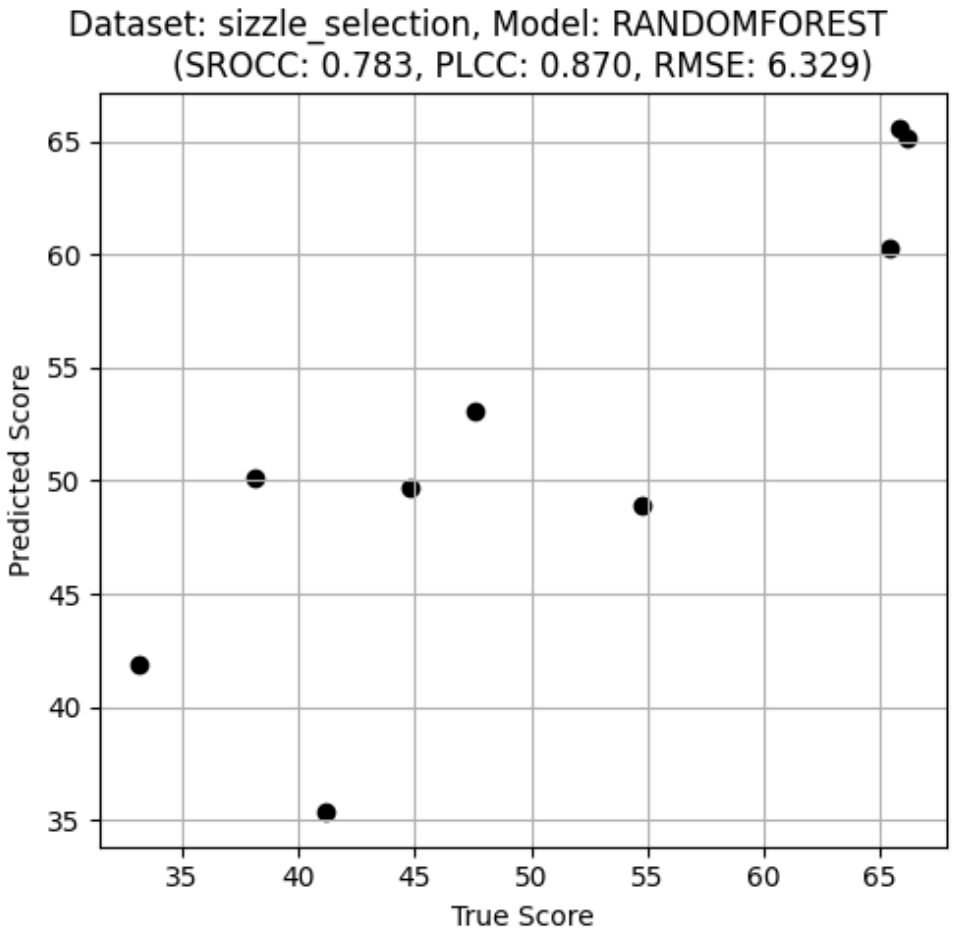




Composite Method Using 6 Base Metrics

- 6 Metrics was used (VIF, DLM, MCPD, PSNR-Y, SSIM, MS-SSIM)





Summary

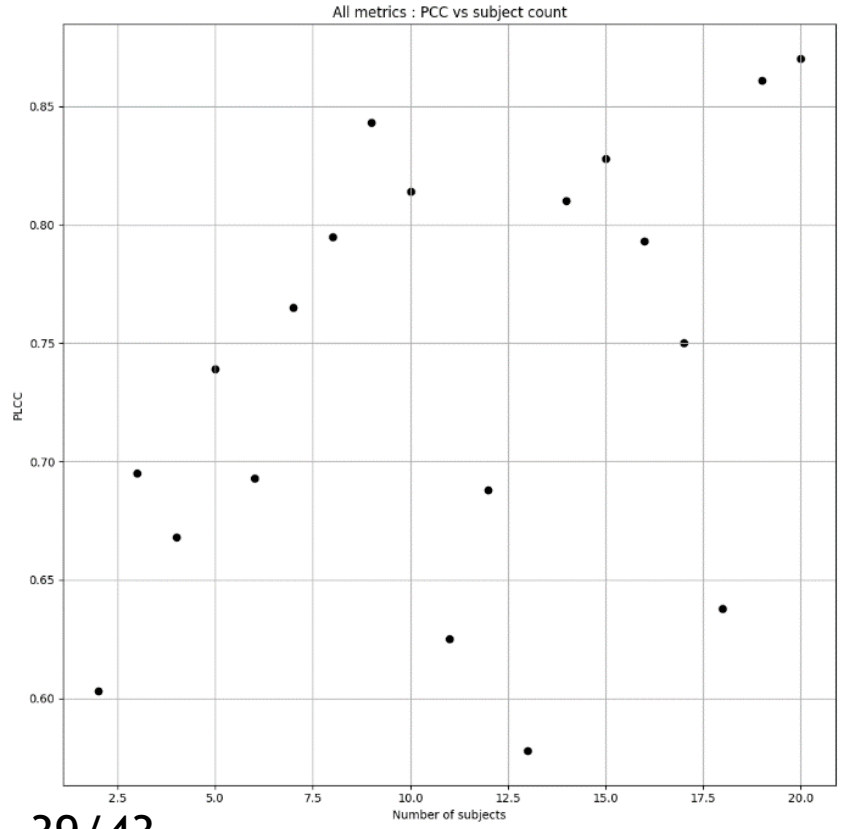
Metric	PLCC	SROCC	RMSE	Time(s)
PSNR	0.609	0.559	13.564	1.774
SSIM	0.759	0.815	9.997	2.484
MS SSIM	0.722	0.784	10.745	19.918
Composite (3 metrics)	0.844	0.883	7.352	5.095
Composite (5 metrics)	0.851	**0.900**	7.436	9.353
Composite (6 metrics)	**0.870**	0.783	**6.329**	29.271

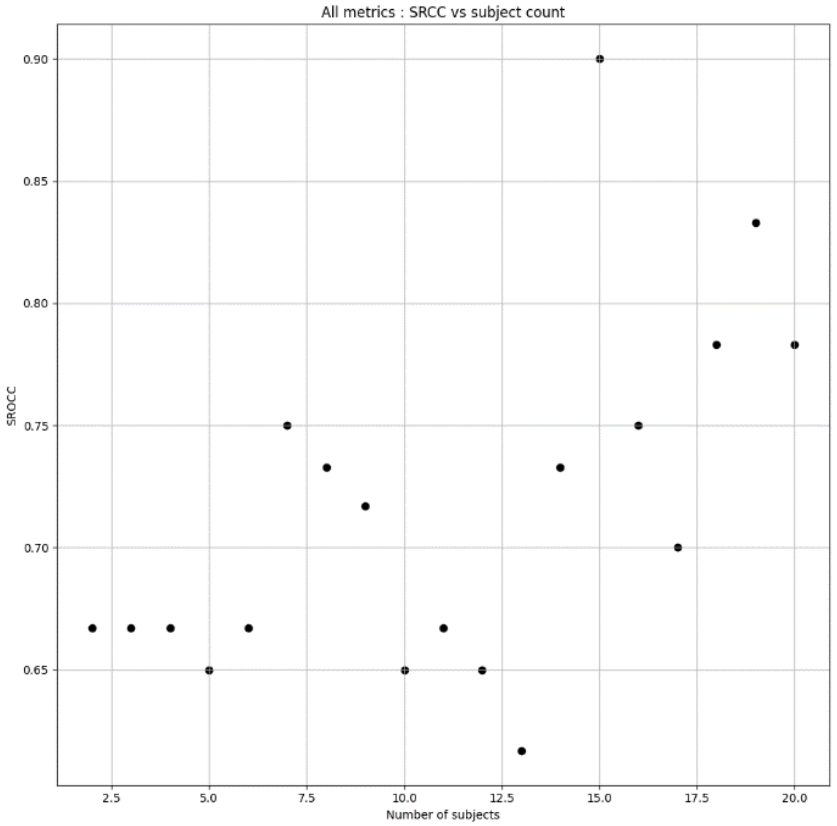
Time Complexity

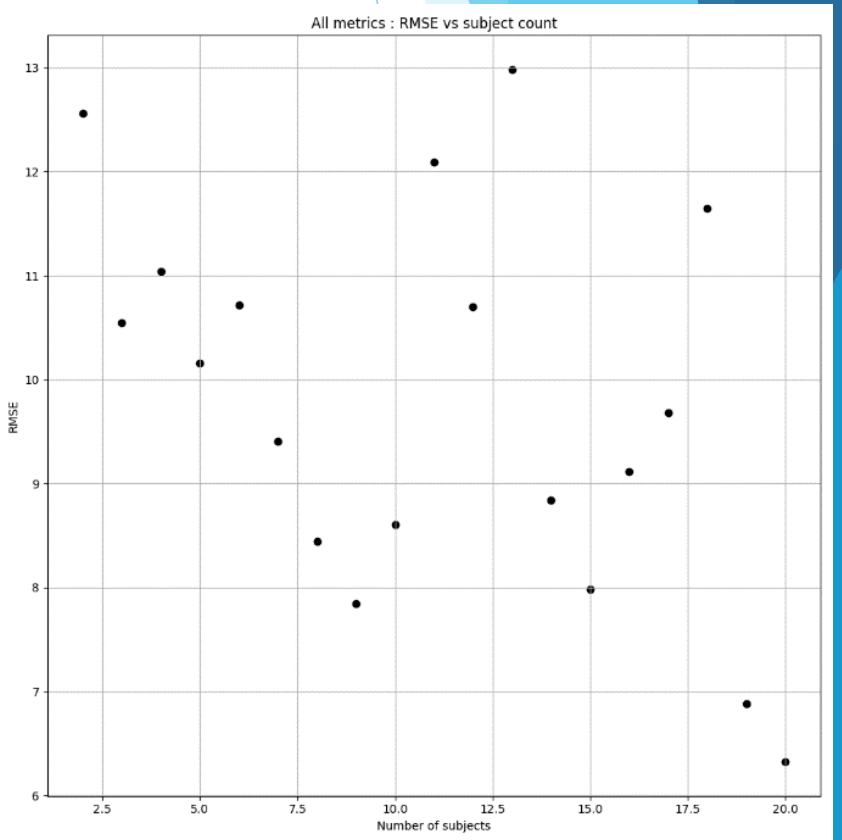
- Factors which contribute to execution time
  - Viewport extraction
    - If using GPU method, might be done in real time and so neglected
    - In our setup, it is not so useful to report this time. Big YUV files & slow HDD storage will corrupt the deduction!
  - Base Metric Calculation
    - Presented in the table
  - Aggregation
    - Almost can be neglected with respect to metric calculation
  - Temporal pooling
    - In our method it was just a simple mean so was negligible.
- It seems the most important factor to optimize time is number of viewports which should be extracted.

Amendments #1

- Effect of number of subjects (**with applying subject weights**)

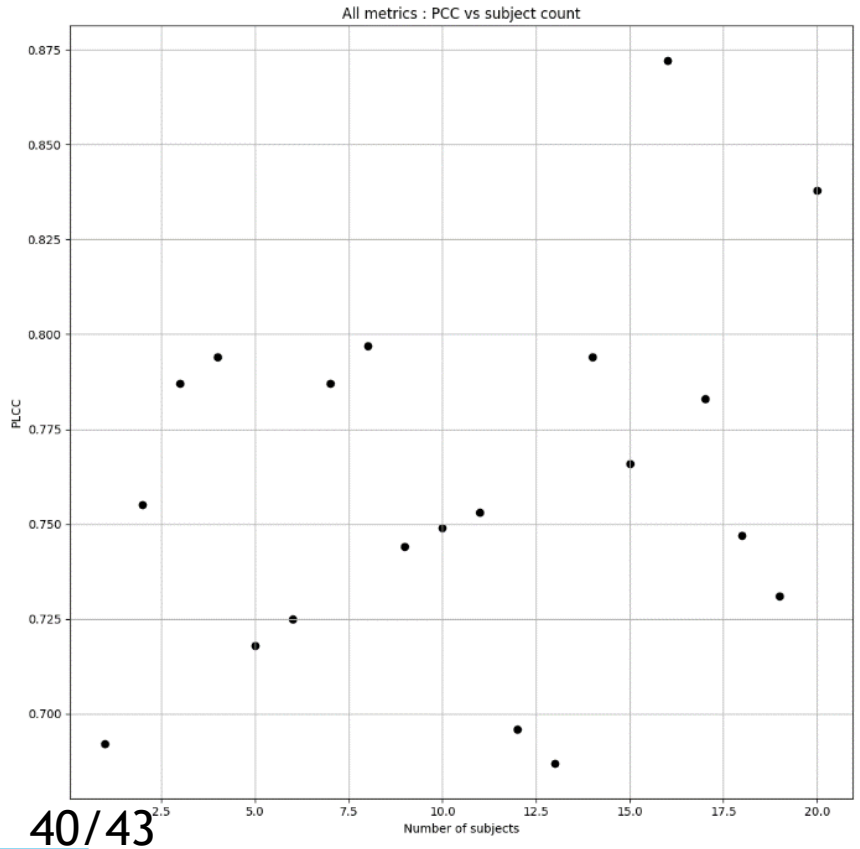


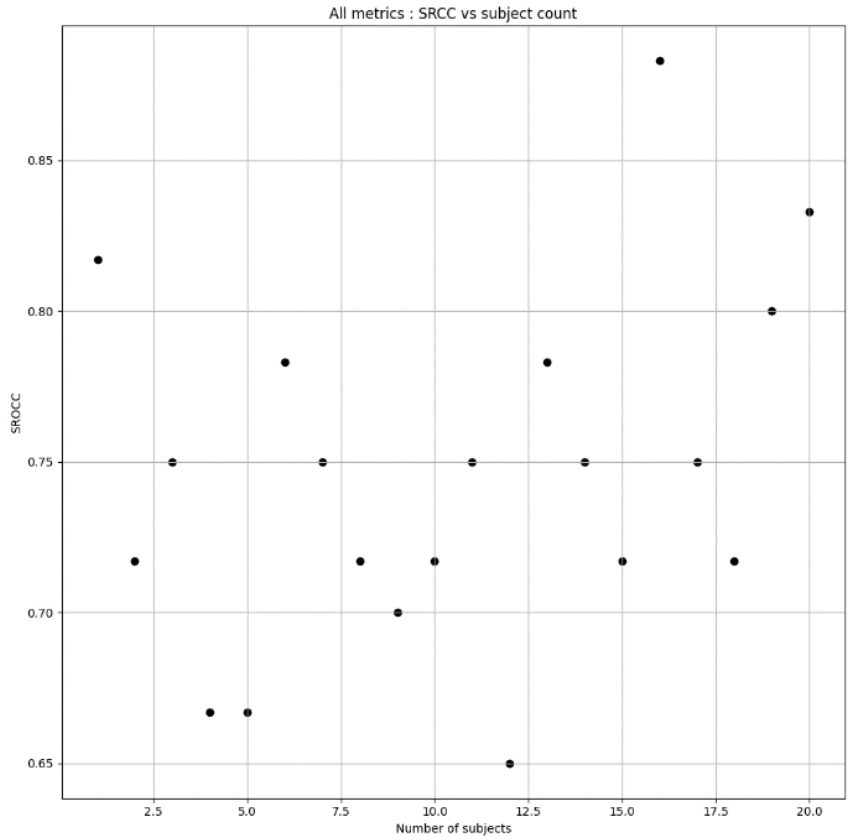


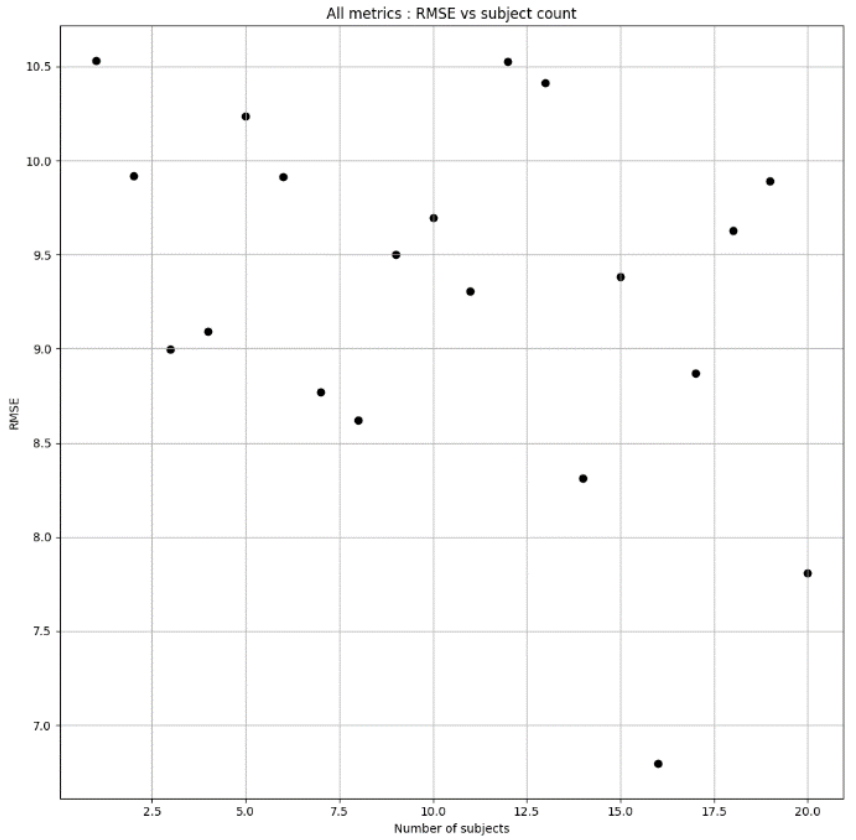


Amendments #2

- Effect of number of subjects (**without applying subject weights**)







Final Note

- Ideally we could compute a lot of metrics in this framework and use methods like PCA to decide on most useful metrics.
- Building a composite metric this way (maybe not limited to 360 videos and even for regular videos) will result hopefully in a more robust metric.
- One can increase or decrease the complexity of that by adding/removing metrics/features. (According to what PCA tells us)

Thank you