AIS 2024 Challenge on Video Quality Assessment of User-Generated Content: Methods and Results

Marcos V. Conde [*][†] Saman Zadtootaghaj [*] Nabajeet Barman [*] Radu Timofte [*]
Chenlong He Qi Zheng Ruoxi Zhu Zhengzhong Tu Haiqiang Wang
Xiangguang Chen Wenhui Meng Xiang Pan Huiying Shi Han Zhu
Xiaozhong Xu Lei Sun Zhenzhong Chen Shan Liu Zicheng Zhang
Haoning Wu Yingjie Zhou Chunyi Li Xiaohong Liu Weisi Lin Guangtao Zhai
Wei Sun Yuqin Cao Yanwei Jiang Jun Jia Zhichao Zhang Zijian Chen
Weixia Zhang Xiongkuo Min Steve Göring Zihao Qi Chen Feng

Abstract

This paper reviews the AIS 2024 Video Quality Assessment (VQA) Challenge, focused on User-Generated Content (UGC). The aim of this challenge is to gather deep learning-based methods capable of estimating the perceptual quality of UGC videos. The user-generated videos from the YouTube UGC Dataset include diverse content (sports, games, lyrics, anime, etc.), quality and resolutions. The proposed methods must process 30 FHD frames under 1 second. In the challenge, a total of 102 participants registered, and 15 submitted code and models. The performance of the top-5 submissions is reviewed and provided here as a survey of diverse deep models for efficient video quality assessment of user-generated content.

1. Introduction

Past two decades have seen a massive increase in popularity and demand for online video streaming applications such as Netflix and YouTube [24]. This has been made possible due to improvements in network capacity, improved end-user devices, and increased computational efficiency, allowing users to stream and watch content for hours over the internet everyday [25]. In order to optimize the end-user experience and provide them with an improved quality of experience, the service provider must measure the perceptual quality of the videos being delivered to them.

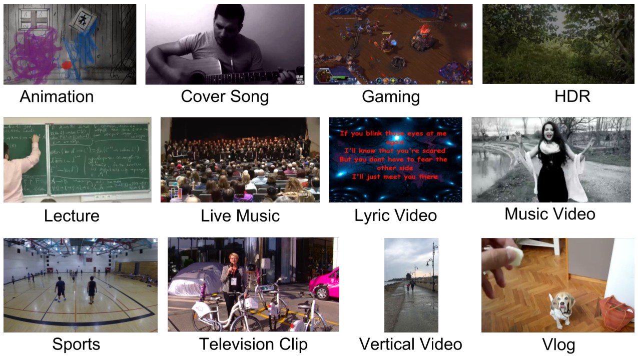


Figure 1. Samples from the videos in the YT-UGC Dataset [33].

Image quality assessment (IQA) or video quality assessment (VQA) can be assessed either subjectively or objectively. In subjective quality assessments, the users directly assess the image/video quality and provide a rating for that [8, 10, 23, 33]. However, such assessment processes are time consuming, costly, and often not realistic in real-world applications. Objective quality models help bridge this gap by using mathematical/statistical models to predict the quality as would be subjectively judged by human observers [19]. In recent years, deep learning techniques have enabled us to learn objective quality metrics from visual content and the corresponding ratings. Depending on the availability of a reference, QA models can broadly be classified into Full-Reference and No-Reference (Blind) [1].

This challenge deals with the design of deep learning-based methods for blind video quality metrics, targeting user-generated content. Given a short video of an arbitrary resolution, the method will predict the overall quality.

In this context, user-generated content refers to content

[*] Marcos V. Conde ([†] corresponding author, project lead) and Radu Timofte are the challenge organizers, while the other authors participated in the challenge and survey.
Marcos V. Conde and Radu Timofte are with University of Würzburg, CAIDAS & IFI, Computer Vision Lab.
Saman Zadtootaghaj, Marcos V. Conde and Nabajeet Barman are with Sony Interactive Entertainment, FTG.
AIS 2024 webpage: https://ai4streaming-workshop.github.io/. Code: https://github.com/mv-lab/VideoAI-Speedrun

that is captured by users using consumer-grade devices, such as (primarily) smartphones, tablets, GoPros, etc. (see Fig. 1), and often shared via platforms such as Instagram, YouTube, TikTok, etc [23, 33, 34]. Unlike professionally generated content, they are usually captured under very challenging conditions, and hence, these can suffer from many artifacts (camera capture impairments, lightning conditions, formats (resolution, fps), etc.).

Recent works focus on designing NR models using deep learning approaches on large-scale datasets to tackle this problem [34, 36, 39]. Deep learning methods are better able to capture and model various factors such as content, distortion, compression, and blur artifacts while also taking into account the temporal aspect for video quality prediction. However, these demand large amounts of annotated data, this has led to the creation of larger, more realistic datasets such as KonVid-1k [10], YouTube YT-UGC [33], and more recently, KonViD-150k VQA Database [7].

2. UGC Video Quality Challenge

2.1. Dataset

The challenge uses the YouTube User-Generated-Content (YT-UGC) dataset [33] that consists of around 1000 video clips with a duration of 20 seconds.

The dataset includes several perceptual artifacts such as blockiness, blur, banding, noise, and jerkiness. In addition, the dataset has a wide range of content types with 15 distinct categories, including animation, gaming, cover songs, music videos, and vlogs, among others.

Moreover, a wide range of resolutions is considered in the dataset, including 360p, 480p, 720p, 1080p, and 2160p.

The clips are annotated with subjective ratings in the 5-point categorical Absolute Rating (ACR) Scale. All videos were rated by more than 100 subjects using crowdsourcing. The Mean Opinion Score (MOS) is obtained on a rating scale of 1 to 5, where 1 is the lowest perceived quality (bad) and 5 is the highest perceived quality (excellent).

For this AIS UGC Video Quality Assessment Challenge, the dataset is split into two sets, training and test. The larger portion of the dataset consisting of 900 clips is used for training, while the test set includes 126 clips, selected carefully considering a balanced range of resolution, content type, and distortion. We show samples in Fig. 1.

2.2. Model Design Rules

- The VQA models should be able to process FHD and HD clips of 30 frames under 1 second. Frame sampling is allowed, as long as the runtime per frame is still ≤ 33 ms. This was measured on an NVIDIA A100 GPU (or similar modern GPUs *e.g.* RTX 3090 Ti).
- We use standard correlation metrics of model scores with subjective (MOS) ratings (SRCC, PCC, KCC).
- Participants were allowed to use any pre-trained and existing solutions.
- The organizers validate the efficiency and reproducibility of the methods.

3. Challenge Results

3.1. Baselines

We consider two Baseline models for benchmarking which are discussed next.

NDNetGaming [31] is a CNN-based quality metric that is designed to assess gaming video quality. NDNetGaming is designed to predict quality in an interpretable range of one to five, where one is the lowest quality, and five is the highest quality score. NDNetGaming uses DenseNet-121 as the backbone and is pre-trained on a large-scale gaming video dataset annotated with VMAF and fine-tuned by a public gaming video dataset. Since NDNetGaming was tailored for images, we used a sampling rate of 5 frames per second and averaged the resultant quality estimation.

We additionally used MobileNet v2 as the second baseline model, which allows us to compare the efficiency of proposed models with a lightweight CNN image encoder architecture. We first process each frame using MobileNet [22]. Next, we average the encoded features for all the frames obtaining a single deep encoded representation, and finally, we predict the quality using a single linear layer. Thus, no frame sampling is applied to the MobileNet result. This represents a naive solution for benchmarking purposes. The baselines are highlighted in blue in Tab. 1.

3.2. Architectures and main ideas

1. **Frame Sampling:** Given a clip of N frames, most methods apply temporal (down)sampling *i.e.* process 1 (or 2) frames of every 30. This allows to increase efficiency without harming performance. Note that this is the reason why we report clip-based metrics instead of frame-based metrics. For instance, a model can virtually process a 30-frame clip in 100 ms, yet it does not imply a 330 FPS performance.
2. **Spatial Downsampling:** Besides pooling in the temporal domain, most approaches resize the frames to lower resolutions (*e.g.* 512px) to reduce memory requirements and operations.
3. **Ensembles:** The best solutions such as COVER [9] and TVQE use multiple image processing models to extract diverse features [34]. Each model is trained to focus on predicting specific properties such as aesthetics or compression. Although combining multiple models might increase training and inference complexity, this approach provides the best performance while being surprisingly efficient.

Team	Method	SROCC	KROCC	PLCC	# Params. [M]	Runtime [ms]		MACs [G]	
						30-FHD	60-HD	30-FHD	60-HD
FudanVIP	COVER [9]	0.914	0.741	0.912	61.02	79.37	78.66	NA	NA
TVQE	TVQE	0.915	0.741	0.918	8254	299.18	294.93	1127.35	1263.53
Q-Align	Q-Align [40]	0.908	0.734	0.912	8198	526.55	429.4	991.17	991.17
SJTU MMLab	SimpleVQA+ [27]	0.906	0.728	0.911	207.7	222.96	394.51	140.17	280.35
AVT	AVT	0.877	0.690	0.878	168	90.57	81.90	NA	NA
BVI-VQA	FasterVQA [38]	0.817	0.638	0.751	28.13	52.49	55.87	NA	NA
Baseline	NDNet [31]	0.718	0.502	0.715	6.95	52.95	24.21	597.47	265.99
Baseline	MobNet	NA	NA	NA	2.22	157.74	138.65	397.31	353.60

Table 1. **AIS 2024 UGC Video Quality Assessment Challenge Benchmark.** We report runtime and MACs operations for a complete 30-frame FHD clip, and 60-frame HD clip. “NA” indicates the results are not available or could not be calculated.

Method	SROCC	KROCC	PLCC	RMSE
BRISQUE [17]	0.4398	0.2934	0.4525	0.5608
GM-LOG [41]	0.3501	0.2336	0.3424	0.5904
VIDEVAL [28]	0.7946	0.5959	0.7691	0.4024
RAPIQUE [29]	0.7483	0.5556	0.7482	0.4177
FAVER [45]	0.7897	0.5832	0.7898	0.3861
NIQE [18]	0.2479	0.1689	0.3146	0.5976
HIGRADE [13]	0.7639	0.5524	0.7507	0.4156
FRIQUEE [5]	0.7182	0.5268	0.7091	0.4439
CORNIA [42]	0.5988	0.4113	0.5905	0.5064
TLVQM [12]	0.6690	0.4833	0.6412	0.4831
CLIPIQA+ [32]	0.5374	0.3734	0.5801	0.5128
FasterVQA [38]	0.5345	0.3716	0.5438	0.5284
FASTVQA [37]	0.6493	0.4676	0.6792	0.4621
DOVER [39]	0.7359	0.5391	0.7653	0.4053
FasterVQA*	0.6937	0.4965	0.6909	0.4552
FASTVQA*	0.8617	0.6716	0.8669	0.3139
DOVER*	0.8761	0.6865	0.8753	0.3144
FasterVQA* (Sec. 4.6)	0.8170	0.6380	0.7510	-
AVT (Sec. 4.5)	0.8775	0.6909	0.8785	-
SimpleVQA+ [27]	0.9060	0.7280	0.9110	-
Q-Align [40]	0.9080	0.7340	0.9120	-
TVQE (Sec. 4.2)	0.9150	0.7410	0.9182	-
COVER [9]	0.9143	0.7413	0.9122	0.2519

Table 2. Extended comparison with classical and previous *state-of-the-art* methods. We report some numbers from [9]. “*” indicates models were fine-tuned using the AIS Challenge dataset.

3.3. Efficiency Study

In Tab. 1 we present the summary of quantitative results and efficiency metrics for each method. The efficiency metrics are calculated using: `https://github.com/mv-lab/VideoAI-Speedrun`. The runtime is the average of 10 independent runs (after GPU warmup).

TVQE and Q-Align [40] use novel LLM-based VQA approaches, thus the number of parameters is considerably high (8 Billion). These approaches leverage video descriptions and visual information to provide accurate ratings. Although the number of parameters and operations is considerably high, the models can process 30 frames under a second, even at high resolution (FHD, 4K).

Team	Method	# Params. [M]	Runtime [ms]	MACs [G]
FudanVIP	COVER [9]	61.02	79.37	NA
TVQE	TVQE	8254	705.30	1127.35
Q-Align	Q-Align [40]	8198	1707.06	991.17
SJTU MMLab	SimpleVQA+ [27]	207.7	245.512	140.175
Baseline	NDNet [31]	6.95	209.43	479.06
Baseline	MobNet	2.22	347.51	1585.32

Table 3. **High-Resolution Efficiency study** using as input a clip of 30 frames of 4K resolution 3840 × 2160. We report the runtime and MACs for the complete clip of 30 frames.

As we show in Tab. 1 and Tab. 3, all the proposed methods can process 30 FHD frames in under 1 second, and 60 HD frames in under 0.5 seconds. Moreover, most approaches can process 30 4K frames under 1 second.

Discussion on frame-wise metrics We report clip-based metrics. Since each method uses different frame sampling techniques, it is difficult to calculate FPS or frame-wise metrics. We instead focus on 30-frame and 60-frame clips.

We can appreciate in Tab. 1 that COVER [9], TVQE and Q-Align [40] have almost constant runtime (or operations) independently of the input resolution or number of frames. The reason is the constant temporal-spatial downsampling on the input video *i.e.* FHD, HD, and 4K frames are always downsampled to the same resolution and fed into the model.

Related Challenges This challenge is one of the AIS 2024 Workshop associated challenges on: Event-based Eye-Tracking [35], Video Quality Assessment of user-generated content [3], Real-time compressed image super-resolution [2], Mobile Video SR, and Depth Upscaling.

4. Challenge Methods and Teams

In the following sections we describe the best challenge solutions. Note that the method descriptions were provided by each team as their contribution to this survey.

4.1. A Comprehensive Video Quality Evaluator

Team FudanVIP

Chenlong He [1], *Qi Zheng* [1], *Ruoxi Zhu* [1], *Zhengzhong Tu* [2]

[1] *State Key Laboratory of Integrated Chips and Systems,*
Fudan University, China
[2] *University of Texas at Austin, America*

Contact: zhengzhong.tu@utexas.edu

The team introduces COVER [9], a comprehensive video quality evaluator, a novel framework designed to evaluate video quality holistically — from a technical, aesthetic, and semantic perspective. Specifically, COVER leverages three parallel branches: (1) a Swin Transformer [15] backbone implemented on spatially sampled crops to predict technical quality; (2) a ConvNet [16] employed on subsampled frames to derive aesthetic quality; (3) a CLIP[21] image encoder executed on resized frames to obtain semantic quality. We further propose a simplified cross-gating block to interact with the three branches before feeding into the predicting head. The final quality score is attained using a weighted sum of each sub-score, making a multi-faceted, explainable metric. Our experimental results demonstrate that COVER exceeds the state-of-the-art models in multiple UGC video quality datasets while it is capable of processing 1080p videos in real-time.

4.1.1 Method

The network architecture of our proposed **CO**mprehensive **V**ideo quality **E**valuato**R** (**COVER**) is illustrated in Fig.2. This network accepts videos that have been subjected to temporal-spatial sampling as its input. Its architecture is divided into three branches: a CLIP-based semantic branch, an aesthetic branch and a technical branch, each consisting of a feature extraction module and a quality regression module. Notably, aesthetic and technical branches additionally incorporate a feature fusion module to integrate features from the semantic branch. The input video is processed through these branches to generate three scores, reflecting the video's quality across the respective dimensions. The final score is the average of scores from three dimensions.

4.1.2 Temporal and Spatial Sampling

As shown in Fig. 2, before serving as input to each branch's feature extraction module, the input videos first undergo temporal-spatio sampling. To enhance the real-time performance of the network, temporal sampling is designed to be very sparse. In the temporal sampling process for the input video, the semantic branch samples one frame out of every thirty frames, while the aesthetic and technical branches sample two frames out of every thirty frames.

For spatial sampling, the semantic and aesthetic branches resize the video resolution to 512x512 and 224x224, respectively. The technical branch, however, employs a fragment operation, where a frame from the video is divided into 7x7 sub-blocks. These sub-blocks are then randomly sampled and reassembled into a frame with a resolution of 224x224.

4.1.3 Feature Extraction

Several studies have demonstrated the effectiveness of CLIP [21], a foundation model, in both IQA [32] and VQA [39] tasks. By extracting semantic information from images and videos, CLIP can accurately assess their subjective quality. However, the aforementioned studies did not address the more challenging task of UGC-VQA. This motivates us to employ the Image Encoder of CLIP as the backbone of the feature extraction module for the semantic branch. The pretrained weights (ViT-L/14) on OpenAI is imported and frozen.

For the technical branch, the Swin Transformer [15] is utilized as the backbone of the feature extraction module. A CNN network, specifically the ConvNet [16], is used as the backbone of the feature extraction module for the aesthetic branch. These two branches are initialized with weights pretrained on the LSVQ [44] from DOVER [39], and it will be fine-tuned during subsequent training.

4.1.4 Feature Fusion

CLIP's image encoder is endowed with robust capabilities in representing image semantics by its numerous training samples. Thus, the abundant information contained in CLIP's output features may inherently correlate with the features of the other branches. To fully harness the representative features generated by the semantic branch and let it modulate the other branches, we propose a feature fusion block. More specifically, we modify the cross-gating block [30], and name it Simple Cross-Gating Block (SCGB), for feature fusion between the semantic-aesthetic and semantic-technical feature pairs. As illustrated in Fig. 2, The fused features from the aesthetic and technical branches, along with the features from the semantic branch, are then fed into their respective quality regression modules.

The detailed architecture of SCGB is depicted in Fig. 2. The input of the block are two tensors X and Y. X is the feature from the technical or aethetic branch, while Y is

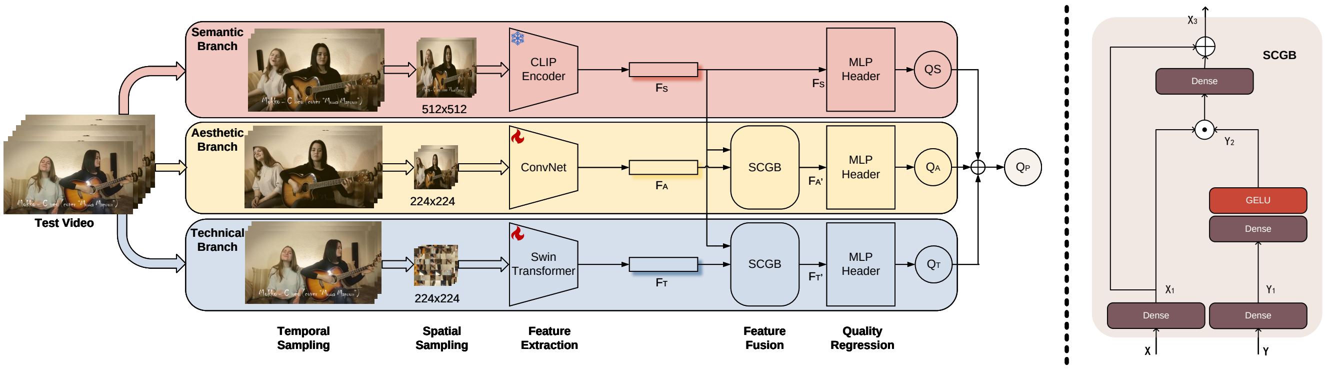


Figure 2. The architecture of our proposed **CO**mprehensive **V**ideo quality **E**valuato**R** (**COVER**). COVER processes a video clip in three parallel branches: 1) a semantic branch that extracts high-level object-semantics-related information using a pre-trained CLIP image Encoder; 2) an aesthetic branch that leverages a ConvNet run on subsampled image thumbnails to analyze their looking; 3) a technical branch utilizing Swin Transformer to execute on fragments. We also devise a simplified cross-gating block (SCGB) to fuse multi-branch features together, yielding the final quality score.

from the CLIP-based semantic branch. After the input channel projections are applied, the projected CLIP features are fed to a gating pathway to yield the gating weights, which are then multiplied by the features from the other branch. Finally, the output projection and residual connection are applied.

4.1.5 Quality Regression

The features from each branch are individually fed into a multi-layer perceptron (MLP) Header to predict quality scores, i.e., Q_S, Q_A, and Q_T, as shown in Fig. 2, and the final predicted quality, $Q_P = (Q_S + Q_A + Q_T)/3$. To enforce that each branch can independently capture the features of its focused dimension and accurately predict video quality, we adopted the limited view biased supervision scheme [39], which minimizes the relative loss between predictions in each branch with the overall opinion MOS, as formulated below:

$$\begin{aligned}\mathcal{L}_{all} =&\mathcal{L}_{rel}(Q_S, \text{MOS}) + \mathcal{L}_{rel}(Q_A, \text{MOS})\\ &+\mathcal{L}_{rel}(Q_T, \text{MOS})\end{aligned} \tag{1}$$

4.1.6 Inference Time

VQA models are highly practical tools potentially deployed on large-scale video streaming platforms to process millions of video streams every day. Therefore, the actual inference cost per video is highly significant to the system's total performance and revenue. We have imbued efficient modular design in every aspect of the COVER model, leading to highly efficient inference speed. We benchmarked the model inference time required by COVER on a video clip of 30 frames of 1080p resolution using a TITAN RTX graphic card. As shown in Table 4, COVER's semantic, aesthetic, and technical branch demands 191, 96, and 23 milliseconds to complete, together adding up to a total inference time of 311 milliseconds. In other words, this inference latency translates to a highly efficient VQA metric that attains state-of-the-art performance with explainable properties and inferences at **96 fps**, almost 3x faster than real-time processing speed.

Table 4. Inference time of COVER on a 30-frame chunk of a 1080p video on a TITAN RTX GPU card. The total 311 ms inference time translates to **96 fps**, 3x faster than real-time processing.

Branch	Semantic	Aesthetic	Technical	All
Time (ms)	191	96	23	311

Implementation details The hyper-parameter settings within COVER for its various components are outlined as follows: i) the backbone of the feature extraction module for semantic branch is the Image Encoder from CLIP [21] of type ViT-L/14; ii) the feature extraction backbone of aesthetic branch is a ConvNet [16], structured into 4 stages. The configuration of each stage, defined by the number of blocks and feature dimensions, is as follows: (3, 96), (192, 3), (384, 9), and (768, 3); iii) the feature extraction backbone of technical branch is a Swin Transformer [15], which also comprises 4 stages. Within each stage, the number of heads is set to 3, 6, 12, and 24, respectively, with the number of projection output channels being 96; iv) the SCGB module operates with input and output feature dimensions both set to 768, and its dropout layer has a drop ratio of 0.1; v) the input feature dimension for the MLP Header module is 768. It includes two dropout layers, both with a drop ratio of 0.5.

The training process for our model is structured into three distinct stages:

1. **Initial Training of Technical and Aesthetic Branches:** Initially, we train the entire network for both the technical and aesthetic branches. During this stage, the weights of both backbones and MLP Headers for all branches are fine-tuned.
2. **Integrating Semantic Branch and Further Fine-tuning:** Building on the best weights obtained from stage 1, we integrate the semantic branch into model. Then MLP Headers of all branches, along with backbones of both technical and aesthetic branches are fine-tuned.
3. **Incorporation of SCGB and Final Fine-tuning:** Based on the optimal weights from stage 2, we add two SCGBs to model. Subsequent fine-tuning of both SCGBs along with all MLP Headers is conducted.

Given the specific validation set of YouTube-UGC, our multi-stage training approach maintains the same data split across each step, allowing for incremental improvements in training effectiveness.

Throughout different training stages, only the specific training set of YouTube-UGC is used. For training strategies. we employ the ADAM optimizer with an initial learning rate of 1×10^{-3} and a cosine learning rate decay strategy with a decay weight of 0.05, over a total of 20 epochs. However, the batch size varies across different stages, being set to 10, 8, and 24 respectively. Our network, implemented in the Pytorch framework and running on an A6000 GPU card, approximately requires one day to complete the entire training process.

4.2. TVQE: Tencent Video Quality Evaluator

Team TVQE

Haiqiang Wang [1], *Xiangguang Chen* [1], *Wenhui Meng* [1],
Xiang Pan [1], *Huiying Shi* [2], *Han Zhu* [2], *Xiaozhong Xu* [1],
Lei Sun [1], *Zhenzhong Chen* [2], *Shan Liu* [1]

[1] *Tencent*
[2] *Wuhan University*

TVQE is a hybrid model trained for VQA tasks. The proposed method fully takes into account several aspects of video quality subjective assessment: 1. Humans make judgments with attention to both global semantic and local visual information; 2. Subjective evaluation experiments usually require observers to learn and judge in discrete text-defined levels. Therefore, it combines three networks, *i.e.*, IQA network, DOVER [39], and Q-Align [40] model, to extract visual information and semantic information and predicts the subjective quality more accurately via weighted fusion operation. The framework of the proposed method is shown in Fig.3.

First, considering that humans have a strong perception of visual information in the spatial dimension when making the judgment, we introduce a feature pyramid aggregation mechanism on the backbone, *i.e.*, the ConvNeXt, to extract visual representations of the key frame. The pyramid structure facilitates the full utilization of the extracted information as well as better exploitation of the shallow visual features. Then, considering the influence of video content on subjective assessment, we use the DOVER model [39] with 3D convolution to assess video quality through aesthetic and technical branches.

Variant	Fusion Ratio	SROCC	PLCC
DOVER (v0)	-	0.822	0.830
DOVER (v1)	-	0.881	0.887
Q-Align5 (v0)	-	0.842	0.838
Q-Align5 (v1)	-	0.895	0.885
Q-Align5 (v2)	-	0.908	0.871
DOVER+Q-Align5	7:8	0.913	0.915

Table 5. Performance of Different TVQE Variants. DOVER (v0) represents the pre-trained model, and (v1) the fine-tuned model. Q-Align5 (v0) represents the pre-trained model, (v1) represents the results by finetuning Visual Abstractor, and (v2) represents the results by finetuning the last 5 transformer layers in Visual Encoder and Visual Abstractor.

Finally, we adopt a large multi-modality model, *i.e.*, Q-align [40], to fit the fact that subjective judgment is usually in discrete text-defined levels. The purpose is to stimulate the behavior of the human annotation process by tuning LLMs (Large Language Models).

These three models were trained independently on the official YT-UGC dataset [33] following the challenge splits. During the inference stage, the final predicted score could be obtained by heuristically fusing the prediction results of these models.

Ablation Study Table 5 gives the ablation study of submitted solution. We finetuned the SOTA DOVER and Q-align model on the give YT-UGC dataset. We take a small split from the training set as the second validation set for model selection.

For the DOVER architecture, it could be seen that the SROCC value increases from 0.822 to 0.881 after carefully finetuning parts of the original network. For the Q-align architecture, we tried different finetune strategy. Empirically, we found that finetuning the last 5 layers of the visual encoder and the visual abstractor block gives the best performance gain, *i.e.*, 0.07 in terms of SROCC.

Then, thanks to the ensemble strategy, the performance is further boosted by 0.005 in terms of SROCC and 0.44 in terms of PLCC, respectively.

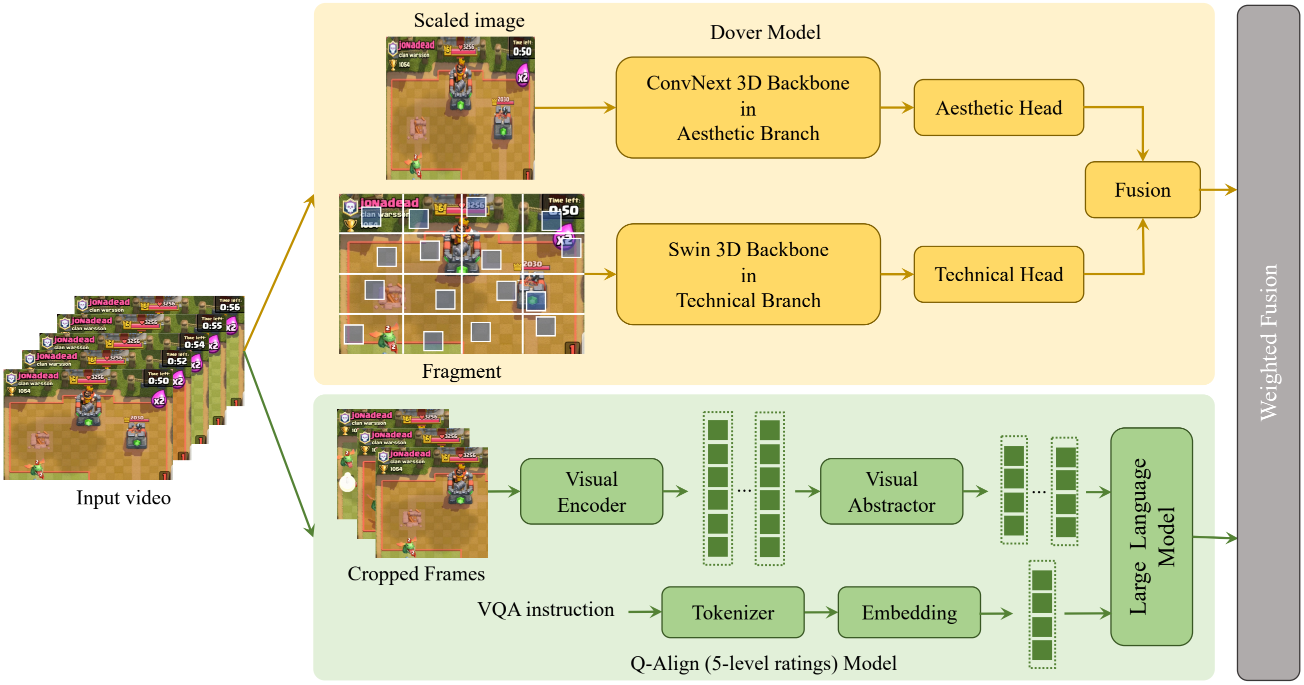


Figure 3. The architecture of the TVQE method.

Inference The processing time for 30 frames with 4K resolution on the NVIDIA RTX 3090 GPU is 0.8 seconds, which meets the required 30 FPS. Thus, the inference runtime with other lower resolutions (e.g., 2K and 1K resolutions) can also satisfy the 30-frames under 1s requirement.

Implementation details

- **Framework:** Pytorch (version 2.0.1)
- **Optimizer and Learning Rate:** AdamW with initial learning rate 2e-5
- **GPU:** NVIDIA Tesla A100 (40G)
- **Datasets:** YT-UGC dataset (challenge split)
- **Training Time:** approximately 10 hours.
- **Training Strategies:** Initialization with the public pretrained model, and training for several epochs.

4.3. Q-Align: Aligning video quality with text descriptions based on LMM

Team Q-Align

Zicheng Zhang [1], *Haoning Wu* [2], *Yingjie Zhou* [1], *Chunyi Li* [1], *Xiaohong Liu* [1], *Weisi Lin* [2], *Guangtao Zhai* [1]

[1] *Shanghai Jiao Tong University*
[2] *Nanyang Technological University*

Contact: zzc1998@sjtu.edu.cn

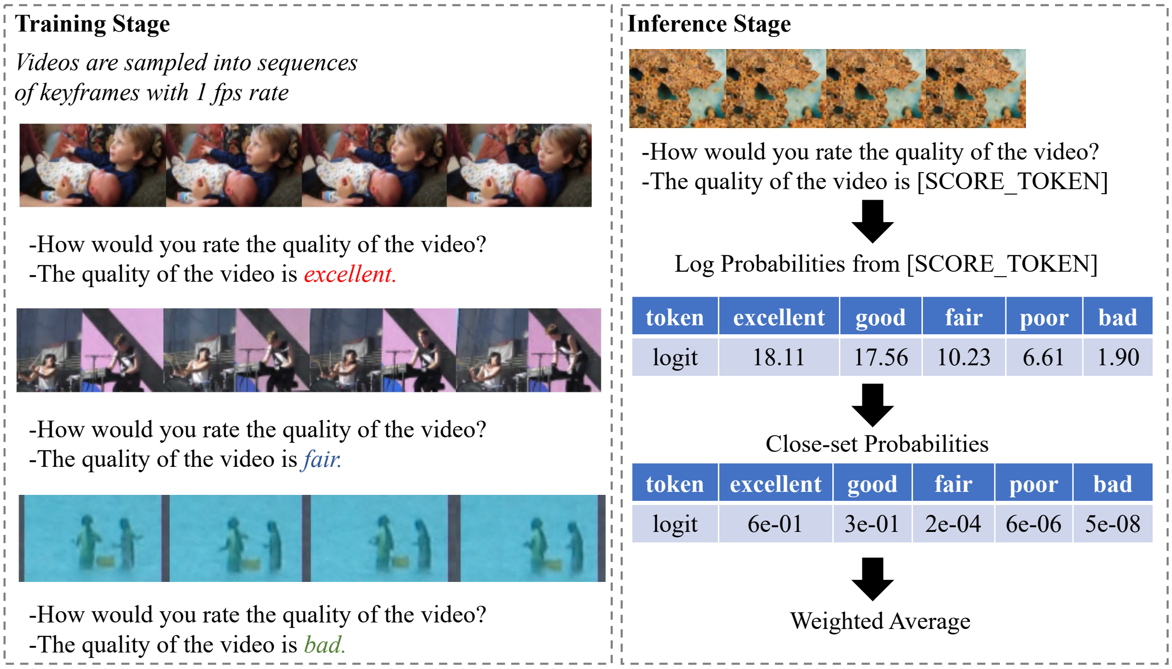
We convert the traditional mean opinion scores (MOS) and the corresponding video into question-answer pairs to teach LMM VQA knowledge. Then we acquire the probabilities of the video quality from LMM response and obtain the final quality values via weighted average.

Q-Align [40] is based on large multi-modality models (LMMs). During the training stage, we divide the quality labels into specific rating categories. Given that the human-assigned ratings are evenly spaced, we utilize equally spaced intervals for transforming scores into these categories. We achieve this by evenly dividing the range from the maximum score (M) to the minimum score (m) into five separate intervals, assigning scores within each interval to corresponding categories:

$$L(s) = l_i \text{ if } \mathrm{m} + \frac{i-1}{5} \times (\mathrm{M}-\mathrm{m}) < s \leq \mathrm{m} + \frac{i}{5} \times (\mathrm{M}-\mathrm{m}) \tag{2}$$

where the set $l_i|_{i=1}^{5} = \{$*bad, poor, fair, good, excellent*$\}$ denotes the established textual rating categories as defined by the ITU. We convert the videos into sequences of keyframes, which are sampled as the first frame of every second. Then we form the question-answer pairs like '*How would you rate the quality of the video?* $|keyframe1||keyframe2|$ ... *The quality of the video is bad/poor/fair/good/excellent*' to fine-tune the LMM.

After training, we can prompt LMM with the same question-answer structure and obtain the responded *[SCORE_TOKEN]* from the '*The quality of the video is [SCORE_TOKEN]*'. The *[SCORE_TOKEN]* can then be



Training Stage
Videos are sampled into sequences of keyframes with 1 fps rate
Inference Stage
Log Probabilities from [SCORE_TOKEN]
Close-set Probabilities
Weighted Average

-How would you rate the quality of the video?
-The quality of the video is *excellent.*

-How would you rate the quality of the video?
-The quality of the video is *fair.*

-How would you rate the quality of the video?
-The quality of the video is *bad.*

-How would you rate the quality of the video?
-The quality of the video is [SCORE_TOKEN]

token	excellent	good	fair	poor	bad
logit	18.11	17.56	10.23	6.61	1.90

token	excellent	good	fair	poor	bad
logit	6e-01	3e-01	2e-04	6e-06	5e-08

Figure 4. The framework of Q-Align [40], where we feed quality question-answer pairs to train LMMs and obtain the 5-level quality probabilities during the inference stage.

converted to the log probabilities of $\{$*bad, poor, fair, good, excellent*$\}$. Finally, we conduct a close-set softmax on log probabilities to get the probabilities p_{l_i} for each level (p_{l_i} for all l_i sum as 1):

$$p_{l_i} = \frac{e^{\mathcal{X}_{l_i}}}{\sum_{j=1}^{5} e^{\mathcal{X}_{l_j}}} \tag{3}$$

and the final predicted scores of LMMs can be derived as

$$\mathrm{S}_{\mathrm{LMM}} = \sum_{i=1}^{5} p_{l_i} G(l_i) = i \times \frac{e^{\mathcal{X}_{l_i}}}{\sum_{j=1}^{5} e^{\mathcal{X}_{l_j}}} \tag{4}$$

During the efficiency test, we find the Q-Align takes up about 8,179M parameters and 991G MACs. Q-Align deals with every 30fps video clip for about 533ms on GPU 3090.

Implementation details We use the PyTorch framework. In experiments, we set batch sizes as 64 and the learning rate is set as $2e-5$. We select mPLUG-Owl-2 as the LMM model. We only train the model on the training set of YT-UGC. We train for 2 epochs for all variants, which takes up about 50 minutes. We conduct training on 4*NVIDIA A100 80G GPUs, and report inference latency on one RTX3090 24G GPU. For videos, we sample at rate *1fps*. The sampled frames are padded to square and then resized to 448×448.

4.4. Blind Video Quality Assessment Models through Spatial and Temporal Quality-Aware Features

Team SJTU MMLab

Wei Sun, Yuqin Cao, Yanwei Jiang, Jun Jia, Zhichao Zhang, Zijian Chen, Weixia Zhang, Xiongkuo Min

Shanghai Jiao Tong University

Contact: suguwei@sjtu.edu.cn

The proposed BVQA model is based on SimpleVQA+ [26, 27], comprising the Swin Transformer-B [15] for spatial feature extraction from key frames, and a temporal pathway of SlowFast for temporal feature extraction from video chunks. Then, we concatenate these features and fuse them into the final quality score via a two-layer MLP. The model is shown in Fig. 5.

We trained SimpleVQA+ on the LSVQ dataset[43]. We utilize LSVQ [43] and YT-UGC dataset [34] for training. During the pre-processing process, we sample one key frame from one-second video chunks (i.e. 1 fps) for the spatial feature extraction module. The resolution of key frames is further resized to 384×384 for training. For the temporal feature extraction module, the resolution of the videos is resized to 224×224. We then split the whole video into

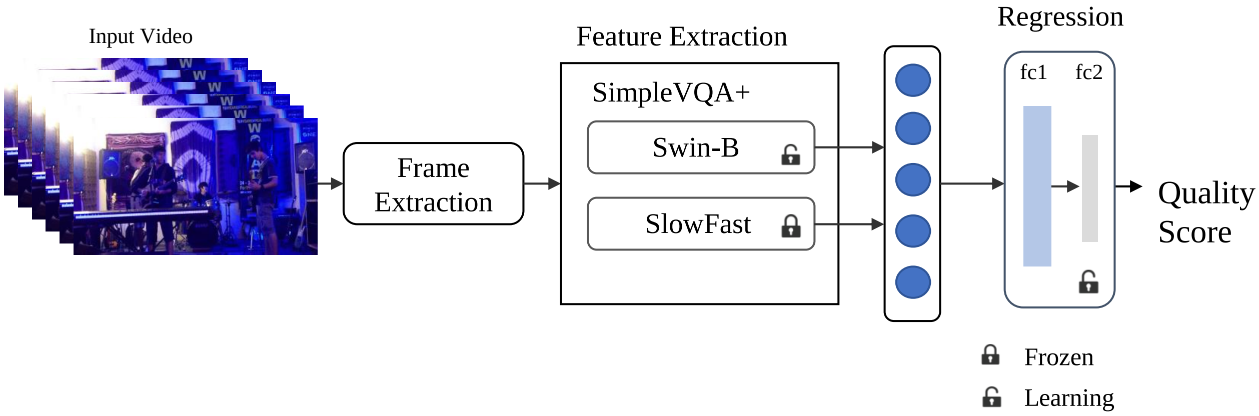


Figure 5. The framework of SimpleVQA+ [26, 27] proposed by Team SJTU MMLab.

several one-second length video chunks to extract the corresponding temporal features.

We train the proposed model on 2 Nvidia RTX 3090 GPUs with a batch size 6 for 30 epochs ($\approx$ 3hrs). The learning rate is set as 10^{-5}. During the inference phase, we feed the video into two models which are trained on the LSVQ and YT-UGC datasets respectively, to obtain prediction scores. Then, we average two scores to obtain the final prediction score. Our proposed model is trained efficiently and can take advantage of other quality-aware pre-trained features, which can help decrease the risk of overfitting.

4.5. Frankenstone – a video quality prediction model combining other models and features

Team AVT

Steve Göring

Audiovisual Technology Group; Technische Universität Ilmenau; Germany

Contact: steve.goering@tu-ilmenau.de

The Frankenstone model uses several other models/features as a baseline and combines them with a random forest regression, similar to [6]. Four main groups are used as features, for each feature value mean aggregation is performed. For example, NVENCC is used to extract metadata and encoding properties (such as bitrate for a specific encoding setting). Furthermore, the DOVER model [39] score and two of its atomic features are used in the Frankenstone model.

In addition, signal-based features, e.g. SI, TI, colorfulness, average luminance, for a subset of the frames are extracted, and on the same subset also VILA model [11] predictions (image appeal) are performed.

The subset of processed frames is done in two steps, the first samples for each second of the video the first frame. The second step takes the sampled frames and reduces them with more importance to the end of the video. That means for a 20 s 30 fps video, 20 frames are sampled, and then out of them, the following 5 frames are used: [0, 6, 11, 15, 18].

All features are extracted in separate threads to make the model faster. Afterwards, the Frankenstone model combines the mentioned features and scores using a Random Forest Regression model. AVT uses DOVER [39] for user-generated video quality prediction, and VILA for per-frame image appeal [11] prediction. Only the YouTube UGC [33] training data was used.

In Fig. 6 an overview of the model structure is provided. The video is fed into the model and then several features are calculated in threads (parallel computation), dover and nvencc features (height, aspect ratio, bitrate for a specific encoding) are calculated for the full video, while pixel (SI, TI, colorfulness, average luminance, sharpness, nima appeal/quality [14], TI calculations to the first frame, SSIM pairwise and to the first frame) and vila features are only calculated for a subset of the video frames (because otherwise, the model would not hit the runtime requirements). The extracted features are combined using a random forest regression (during model development with a varying number of trees, the submitted model uses 300 trees).

The runtime of the model has been evaluated exemplarily with various videos, in the following the `Sports_2160P-210c.mkv` (30 fps, UHD-1, 20s duration) video is used. The 24 time measurements result in an average runtime of $\approx 19.616\,s$, with a standard derivation of $\approx 0.138\,s$. However, this may vary, depending on a warm start of the model (and corresponding file-system caches). The model may not be fast enough for smaller videos, because the data must be transferred to the GPU first.

https://github.com/VQAssessment/DOVER

https://github.com/google-research/google-research/tree/master/vila

dover does also frame sub-sampling

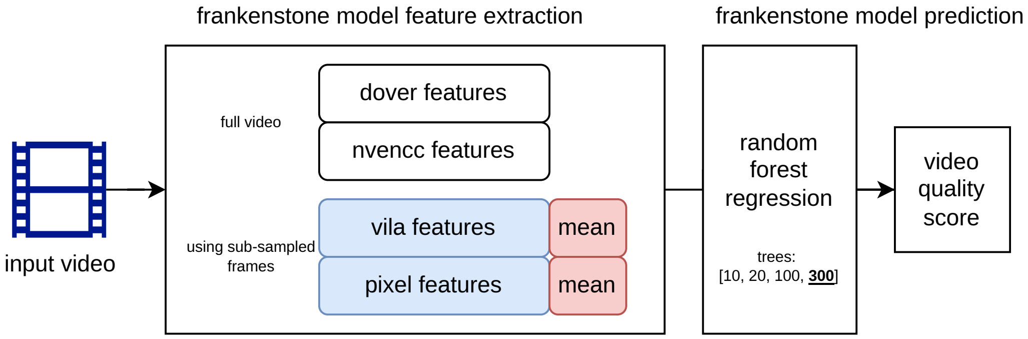


Figure 6. Overview of the `frankenstone` model proposed by Team AVT.

Implementation details

- **Framework:** For feature extraction mainly Tensorflow is used, however, some of the included models rely on PyTorch, and the final score is predicted with a random forest regression model (part of Tensorflow Decision Forests package).
- **Optimizer and Learning Rate:** A random forest model with a variable number of trees (10, 20, 100, and 300) have been used, there was no improvement using more trees, the final model has 300 trees.
- **GPU:** NVIDIA GeForce RTX 3090 Ti (24 GB)
- **Datasets:** Youtube UGC training data, no augmentation.
- **Training Time:** Extraction of features for each video ≈ 20 s max, thus 892 training videos, ≈ 12 h extraction time (was performed with 3-4 parallel processes to reduce the time, overall on one PC), training the random forest regression model takes $<$ 1 min (part of the Tensorflow Decision Forests package).
- **Efficiency Optimization Strategies:** Performing feature extraction in parallel threads.

4.6. Ranking-based training strategy in siamese manner

Team BVI-VQA

Zihao Qi, Chen Feng

Visual Information Laboratory, University of Bristol

Contact: `zihao.qi@bristol.ac.uk`

The team uses FasterVQA [38] as backbone, training in a siamese manner. During training, the siamese network takes a pair of videos as input and tries to predict which one is in better quality. This training strategy, following a similar methodology proposed in previous works [4, 20], makes it possible to train our model on multiple datasets with various scoring scale (YouTube-UGC [33], LIVE-VQC [23], KoNVid-1k). After trained in siamese manner, the FasterVQA model is then fine-tuned on YouTube-UGC.

Method	SROC
FasterVQA with Siamese Training	0.818
Pre-trained FasterVQA	0.813
Pre-trained SimpleVQA	0.792

Table 6. Ablation study on the testing set by Team BVI-VQA.

Based on the intuition to train our model over multiple datasets, we proposed a ranking-based training strategy to train an existing SOTA network, FasterVQA [38], in a siamese manner.

A common challenge when training on multiple datasets is: different datasets usually have inconsistent scoring scale and crowdsourcing protocol. To solve this problem, we trained our model using a siamese structure, consisting of two FasterVQA networks sharing the same weights. At each time, the siamese network takes a random pair of videos from the same dataset as input and learns to predict which one is of the better quality (with higher MOS ground-truth value). Because the network does not directly take MOS as training labels, it avoids the problem that MOS from different datasets may have different scoring scale. This ranking-based training strategy shares a similar insight as previous works [4, 20]. Pre-trained model from FasterVQA has been used to initialize the training. After trained 20 epoches over 3 datasets (YouTube-UGC [33], LIVE-VQC [23], KoNVid-1k) in siamese manner, the model is then finetuned on YouTube-UGC. The training framework is illustrated in Fig. 7.

Implementation details

- **Framework:** PyTorch.
- **Optimizer and Learning Rate:** AdamW using learning rate 1e-4 and weight decay 0.05.
- **GPU:** NVIDIA RTX 3090.
- **Datasets:** YouTube-UGC, LIVE-VQC, KoNVid-1k.
- **Training Time:** 12h.

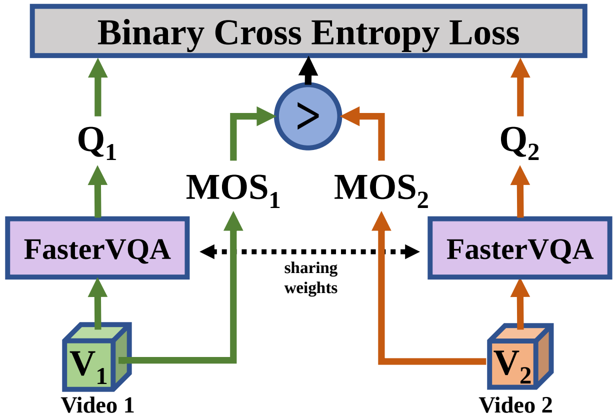


Figure 7. Overall picture of the siamese training process proposed by Team BVI-VQA.

Acknowledgements

This work was partially supported by the Humboldt Foundation. We thank the AIS 2024 sponsors: Meta Reality Labs, Meta, Netflix, Sony Interactive Entertainment (FTG), and the University of Würzburg (Computer Vision Lab).

The challenge organizers thank Ioannis Katsavounidis (Meta), Christos Bampis (Netflix), and Balu Adsumilli (Google) for their feedback.

References

[1] Nabajeet Barman and Maria G. Martini. QoE Modeling for HTTP Adaptive Video Streaming–A Survey and Open Challenges. *IEEE Access*, 7:30831–30859, 2019. 1

[2] Marcos V. Conde, Zhijun Lei, Wen Li, Ioannis Katsavounidis, Radu Timofte, et al. Real-time 4k super-resolution of compressed AVIF images. AIS 2024 challenge survey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3

[3] Marcos V. Conde, Saman Zadtootaghaj, Nabajeet Barman, Radu Timofte, et al. AIS 2024 challenge on video quality assessment of user-generated content: Methods and results. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3

[4] Chen Feng, Duolikun Danier, Fan Zhang, and David R Bull. RankDVQA: Deep vqa based on ranking-inspired hybrid training. *arXiv preprint arXiv:2202.08595*, 2022. 10

[5] Deepti Ghadiyaram. Perceptual quality prediction on authentically distorted images using a bag of features approach. *Journal of Vision*, 17(1)(32):1–25, 2017. 3

[6] Steve Göring, Rakesh Rao Ramachandra Rao, Bernhard Feiten, and Alexander Raake. Modular framework and instances of pixel-based video quality models for uhd-1/4k. *IEEE Access*, 9:31842–31864, 2021. 9

[7] Franz Götz-Hahn, Vlad Hosu, Hanhe Lin, and Dietmar Saupe. KonVid-150k: A Dataset for No-Reference Video Quality Assessment of Videos in-the-Wild. In *IEEE Access 9*, pages 72139–72160. IEEE, 2021. 2

[8] Jinjin Gu, Haoming Cai, Chao Dong, Jimmy S Ren, Radu Timofte, Yuan Gong, Shanshan Lao, Shuwei Shi, Jiahao Wang, Sidi Yang, et al. Ntire 2022 challenge on perceptual image quality assessment. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 951–967, 2022. 1

[9] Chenlong He, Chenlong He, Ruoxi Zhu, Xiaoyang Zeng, Yibo Fan, and Zhengzhong Tu. COVER: A comprehensive video quality evaluator. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 2, 3, 4

[10] Vlad Hosu, Franz Hahn, Mohsen Jenadeleh, Hanhe Lin, Hui Men, Tamás Szirányi, Shujun Li, and Dietmar Saupe. The konstanz natural video database (konvid-1k). In *2017 Ninth international conference on quality of multimedia experience (QoMEX)*, pages 1–6. IEEE, 2017. 1, 2

[11] Junjie Ke, Keren Ye, Jiahui Yu, Yonghui Wu, Peyman Milanfar, and Feng Yang. Vila: Learning image aesthetics from user comments with vision-language pretraining. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10041–10051, 2023. 9

[12] Jari Korhonen. Two-level approach for no-reference consumer video quality assessment. *IEEE Trans. Image Process.*, 28(12):5923–5938, 2019. 3

[13] Debarati Kundu, Deepti Ghadiyaram, Alan C Bovik, and Brian L Evans. No-reference quality assessment of tone-mapped HDR pictures. *IEEE Trans. Image Process.*, 26(6): 2957–2971, 2017. 3

[14] Christopher Lennan, Hao Nguyen, and Dat Tran. Image quality assessment. https://github.com/idealo/image-quality-assessment, 2018. 9

[15] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021. 4, 5, 8

[16] Zhuang Liu, Hanzi Mao, Chao-Yuan Wu, Christoph Feichtenhofer, Trevor Darrell, and Saining Xie. A convnet for the 2020s. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 11976–11986, 2022. 4, 5

[17] Anish Mittal, Anush Krishna Moorthy, and Alan Conrad Bovik. No-reference image quality assessment in the spatial domain. *IEEE Trans. Image Process.*, 21(12):4695–4708, 2012. 3

[18] Anish Mittal, Rajiv Soundararajan, and Alan C. Bovik. Making a "completely blind" image quality analyzer. *IEEE Signal Processing Letters*, 20(3):209–212, 2013. 3

[19] Netflix. VMAF - Video Multi-Method Assessment Fusion. https://github.com/Netflix/vmaf. 1

[20] Zihao Qi, Chen Feng, Duolikun Danier, Fan Zhang, Xiaozhong Xu, Shan Liu, and David Bull. Full-reference video quality assessment for user generated content transcoding. *arXiv preprint arXiv:2312.12317*, 2023. 10

[21] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning

transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021. 4, 5

[22] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. Mobilenetv2: Inverted residuals and linear bottlenecks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4510–4520, 2018. 2

[23] Zeina Sinno and Alan Conrad Bovik. Large-scale study of perceptual video quality. *IEEE Transactions on Image Processing*, 28(2):612–627, 2018. 1, 2, 10

[24] Statista. Number of users of OTT video worldwide from 2020 to 2029 (in millions) [Graph]. https://www.statista.com/forecasts/1207843/ott-video-users-worldwide/, . 1

[25] Statista. Daily time spent on social networking by internet users worldwide from 2012 to 2024 (in minutes) [Graph]. https://www.statista.com/statistics/433871/daily-social-media-usage-worldwide/, . 1

[26] Wei Sun, Xiongkuo Min, Wei Lu, and Guangtao Zhai. A deep learning based no-reference quality assessment model for ugc videos. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 856–865, 2022. 8, 9

[27] Wei Sun, Wen Wen, Xiongkuo Min, Long Lan, Guangtao Zhai, and Kede Ma. Analysis of video quality datasets via design of minimalistic video quality models. *arXiv preprint arXiv:2307.13981*, 2023. 3, 8, 9

[28] Zhengzhong Tu, Yilin Wang, Neil Birkbeck, Balu Adsumilli, and Alan C. Bovik. UGC-VQA: Benchmarking blind video quality assessment for user generated content. *IEEE Trans. Image Process.*, 30:4449–4464, 2021. 3

[29] Zhengzhong Tu, Xiangxu Yu, Yilin Wang, Neil Birkbeck, Balu Adsumilli, and Alan C Bovik. RAPIQUE: Rapid and accurate video quality prediction of user generated content. *IEEE Open Journal of Signal Processing*, 2:425–440, 2021. 3

[30] Zhengzhong Tu, Hossein Talebi, Han Zhang, Feng Yang, Peyman Milanfar, Alan Bovik, and Yinxiao Li. Maxim: Multi-axis mlp for image processing. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 5769–5780, 2022. 4

[31] Markus Utke, Saman Zadtootaghaj, Steven Schmidt, Sebastian Bosse, and Sebastian Möller. Ndnetgaming-development of a no-reference deep cnn for gaming video quality prediction. *Multimedia Tools and Applications*, pages 1–23, 2022. 2, 3

[32] Jianyi Wang, Kelvin CK Chan, and Chen Change Loy. Exploring clip for assessing the look and feel of images. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 2555–2563, 2023. 3, 4

[33] Yilin Wang, Sasi Inguva, and Balu Adsumilli. Youtube ugc dataset for video compression research. In *2019 IEEE 21st International Workshop on Multimedia Signal Processing (MMSP)*, pages 1–5. IEEE, 2019. 1, 2, 6, 9, 10

[34] Yilin Wang, Junjie Ke, Hossein Talebi, Joong Gon Yim, Neil Birkbeck, Balu Adsumilli, Peyman Milanfar, and Feng Yang. Rich features for perceptual quality assessment of ugc videos. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13435–13444, 2021. 2, 8

[35] Zuowen Wang, Chang Gao, Zongwei Wu, Marcos V. Conde, Radu Timofte, Shih-Chii Liu, Qinyu Chen, et al. Event-Based Eye Tracking. AIS 2024 Challenge Survey. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, 2024. 3

[36] Wen Wen, Mu Li, Yabin Zhang, Yiting Liao, Junlin Li, Li Zhang, and Kede Ma. Modular Blind Video Quality Assessment, 2024. 2

[37] Haoning Wu, Chaofeng Chen, Jingwen Hou, Liang Liao, Annan Wang, Wenxiu Sun, Qiong Yan, and Weisi Lin. Fast-vqa: Efficient end-to-end video quality assessment with fragment sampling. In *European conference on computer vision*, pages 538–554. Springer, 2022. 3

[38] Haoning Wu, Chaofeng Chen, Liang Liao, Jingwen Hou, Wenxiu Sun, Qiong Yan, Jinwei Gu, and Weisi Lin. Neighbourhood representative sampling for efficient end-to-end video quality assessment. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023. 3, 10

[39] Haoning Wu, Erli Zhang, Liang Liao, Chaofeng Chen, Jingwen Hou Hou, Annan Wang, Wenxiu Sun Sun, Qiong Yan, and Weisi Lin. Exploring video quality assessment on user generated contents from aesthetic and technical perspectives. In *International Conference on Computer Vision (ICCV)*, 2023. 2, 3, 4, 5, 6, 9

[40] Haoning Wu, Zicheng Zhang, Weixia Zhang, Chaofeng Chen, Liang Liao, Chunyi Li, Yixuan Gao, Annan Wang, Erli Zhang, Wenxiu Sun, et al. Q-align: Teaching lmms for visual scoring via discrete text-defined levels. *arXiv preprint arXiv:2312.17090*, 2023. 3, 6, 7, 8

[41] Wufeng Xue, Xuanqin Mou, Lei Zhang, Alan C Bovik, and Xiangchu Feng. Blind image quality assessment using joint statistics of gradient magnitude and laplacian features. *IEEE Trans. Image Process.*, 23(11):4850–4862, 2014. 3

[42] Peng Ye, Jayant Kumar, Le Kang, and David Doermann. Unsupervised feature learning framework for no-reference image quality assessment. In *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, pages 1098–1105, 2012. 3

[43] Z Ying, M Mandal, D Ghadiyaram, and AC Bovik. Live large-scale social video quality (lsvq) database. *Online: https://github. com/baidut/PatchVQ*, 2020. 8

[44] Zhenqiang Ying, Maniratnam Mandal, Deepti Ghadiyaram, and Alan Bovik. Patch-vq:'patching up'the video quality problem. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 14019–14029, 2021. 4

[45] Qi Zheng, Zhengzhong Tu, Pavan C Madhusudana, Xiaoyang Zeng, Alan C Bovik, and Yibo Fan. Faver: Blind quality prediction of variable frame rate videos. *Signal Processing: Image Communication*, 122:117101, 2024. 3