POLITECNICO DI TORINO

Corso di Laurea in Ingegneria Informatica

Tesi di Laurea Magistrale

Sviluppo di un framework per la misurazione della qualità video: confronto tra VMAF e valutazioni soggettive

Relatore
Prof. Enrico Masala

Laureando
Roberto Greco

Anno accademico 2024/2025

Sommario

Per garantire un'esperienza visiva di alta qualità, le piattaforme di streaming devono disporre di strumenti precisi per misurare la qualità percepita di un flusso video su larga scala. Nel corso degli anni sono stati sviluppati diversi approcci, basati su metodi soggettivi o oggettivi. La prima parte si concentra sull'analisi dei differenti metodi di valutazione. L'obiettivo centrale del lavoro è la creazione di un framework automatizzato per l'esecuzione di valutazioni di qualità su un insieme di dataset selezionati, garantendo la riproducibilità dei risultati. Il framework utilizza VMAF, un tool sviluppato da Netflix, considerato lo stato dell'arte nella valutazione della qualità video, che consente l'analisi e il confronto tra diverse metriche di valutazione. Inoltre, è stato adoperato anche il tool eSSIM, sviluppato da Meta, per effettuare valutazioni utilizzando la metrica oggettiva eSSIM. L'ultima parte dell'elaborato si concentra sull'analisi dei risultati, con particolare attenzione al confronto tra valutazioni oggettive e soggettive, quest'ultime fornite dal MOS (Mean Opinion Score), un indice che misura la qualità percepita dagli utenti in una scala che va da 1 a 5. Particolare attenzione è stata posta nel valutare l'efficacia di diversi metodi di temporal pooling, metodi che aggregano le valutazioni di qualità provenienti da singoli frame per ottenere un unico punteggio complessivo che rappresenta la qualità dell'intera sequenza video. VMAF si è confermato come la metrica migliore per le valutazioni oggettive, mentre eSSIM ha comunque dato risultati rilevanti. Si è osservato che, con il calo del MOS, aumenta l'incertezza nei valori di VMAF per i modelli analizzati. È emerso inoltre come la scelta del modello influenzi l'esito della valutazione. Infine, la media si è dimostrata un metodo efficace per l'aggregazione temporale dei dati.

Indice

Elenco delle tabelle

Elenco delle figure

Abbreviazioni

- **AV1**: AOMedia Video 1
- **AVC**: Advanced Video Coding
- **DLM** : Detail Loss Metric
- **FPS**: Frames Per Second
- **FR**: Full Reference
- **HEVC**: High Efficiency Video Coding
- **HRC**: Hypothetical Reference Circuit
- **HVS**: Human Visual System
- **KUGVD** : Kingston University Gaming Video Dataset
- **MSE**: Mean Squared Error
- **MOS**: Mean Opinion Score
- **MS-SSIM**: Multi-Scale Structural Similarity Index
- **NR**: No Reference
- **OS**: Opinion Scores
- **PSNR**: Peak Signal-to-Noise Ratio
- **PVS**: Processed Video Sequences
- **RR**: Reduced Reference

- **SI** : Spatial Information
- **SVM**: Support Vector Machine
- **TI** :Temporal Information
- **VMAF**: Video Multi-Method Assessment Fusion
- **VQEG**: Video Quality Experts Group
- **VIF** : Visual information Fidelity
- **VVC**: Versatile Video Coding

Capitolo 1

Introduzione

1.1 L'importanza dei contenuti video

La fruizione di contenuti multimediali, come immagini e video, è diventata un aspetto centrale della nostra vita quotidiana. Nel corso degli anni, la quantità di video condivisi online è aumentata in modo esponenziale. Come affermato nel Global Internet Phenomena di Sandvine del 2024[2]:

- il video è di gran lunga il principale responsabile del volume di traffico in downstream su reti fisse e mobili, rappresentando il 38% del traffico con un volume medio totale di 15,7 GB per subscriber al giorno. Per "volume per subscriber" si intende il volume giornaliero di un'app diviso per il numero totale di abbonati. Il subscriber, in una rete fissa, include tutti i membri e i dispositivi che utilizzano le app. Su reti fisse, il Video rappresenta il 39% con 5,7 GB per famiglia al giorno, mentre su reti mobili rappresenta il 31% del volume totale con 493 MB per utenza familiare al giorno

- lo Streaming On-Demand è il tipo di video che genera più traffico, con un volume per unità familiare di 7,9 GB al giorno (54% del volume totale in downstream). Tra queste, l'app che genera più traffico è YouTube, con 1,9 GB al giorno, seguita da Netflix con 1,4 GB per nucleo domestico.

- i social media sono la principale categoria di applicazioni nel traffico in downstream su reti mobili, con il 35%, e sette delle dieci principali app attribuibili a Facebook, TikTok, Instagram, Snapchat, WhatsApp, Telegram e X. I Social Media sono anche la principale fonte di traffico in upstream su dispositivi mobili, principalmente a causa della condivisione e pubblicazione di video su app come TikTok e Facebook Messenger.

1.1.1 L'importanza della compressione

Per *video* si intende una sequenza di immagini, chiamate *frames*, che si susseguono rapidamente in sequenza ad una velocità costante del tempo chiamate *frame-rate*. Il cervello le interpreta come un flusso in movimento grazie al fenomeno della persistenza della visione.
Un parametro fondamentale nell'ambito del video è il **bitrate**. Esso indica la quantità di dati utilizzata per rappresentare il video in un dato intervallo di tempo, solitamente espresso in kilobit al secondo (kbps) o megabit al secondo (Mbps). Un bitrate più alto migliora la qualità del video, ma richiede maggiore spazio di archiviazione. Per ridurre le dimensioni dei file e facilitare anche la loro trasmissione è necessario comprimerli. L'operazione di codifica dei contenuti è fondamentale per diversi motivi:

  - i video non codificati raggiungono dimensioni considerevoli: ad esempio, una sequenza video non compressa a risoluzione Full HD (1080p) può arrivare ad occupare oltre 1.2 Gbps.
  - risulta necessaria per lo streaming: senza alcuna forma di riduzione di dimensioni dei contenuti alcuni servizi non potrebbero essere disponibili in alcune parti del mondo
  - non si potrebbero gestire alti bitrate, poichè ci permette di ridurre le dimensioni dei file mantenendo una qualità accettabile.

Nel corso degli anni, sono stati sviluppati diversi standard di codifica con l'obiettivo di migliorare l'efficienza della compressione senza compromettere eccessivamente la qualità dell'immagine. Tra

i più diffusi vi sono AVC (Advanced Video Coding, noto anche come H.264) e HEVC (High Efficiency Video Coding, o H.265). AVC è stato a lungo lo standard più utilizzato grazie alla sua buona qualità e alla compatibilità con la maggior parte dei dispositivi, mentre HEVC ha introdotto tecniche di compressione più avanzate, consentendo di ottenere una qualità simile o superiore rispetto ad AVC, ma con un bitrate inferiore. Negli ultimi anni, sono emersi codec ancora più avanzati, come AV1 e VVC (Versatile Video Coding, o H.266), che promettono una compressione ancora più efficiente, riducendo ulteriormente il bitrate necessario per mantenere uno standard di qualità elevato. Tuttavia, l'utilizzo di codec più recenti comporta anche costi computazionali maggiori, sia in fase di codifica che di decodifica, rendendone l'adozione più complessa per alcune piattaforme. La qualità percepita del video, quindi, non dipende solo dal bitrate, ma anche dal tipo di codec utilizzato. All'interno del lavoro di tesi verranno utilizzati cinque dataset, ciascuno caratterizzato da contenuti diversi, differenti codificatori e vari bitrate. Questi dataset includono valutazioni soggettive, che verranno confrontate con metriche oggettive per analizzare in modo approfondito la qualità video.

Capitolo 2

Valutazione della qualità nei video

Si possono utilizzare due approcci differenti per effettuare una valutazione di qualità video:

- **test soggettivi**: vengono mostrate alcune sequenze video a gruppo di persone (non è necessario che siano esperte di qualità video) e viene chiesto loro di esprimere un giudizio sulla qualità percepita. Tali esperimenti vengono solitamente realizzati in ambienti controllati e progettati per evitare che fattori esterni influenzino la percezione del video, garantendo così l'affidabilità dei giudizi espressi. I risultati ottenuti da questi test vengono poi utilizzati per calcolare il MOS (Mean Opinion Score)
- **metodi oggettivi**: si utilizzano algoritmi che cercano di approssimare in maniera ragionevole la valutazione di qualità

2.1 MOS

Diversi partecipanti vengono sottoposti a un test in cui sono invitati ad assegnare un giudizio alla qualità del contenuto visualizzato, su una scala che va da 1 a 5[3] . Al termine del test, vengono raccolti i punteggi dei partecipanti e calcolata la media, che fornirà il valore del MOS, indicando la qualità percepita dagli utenti[4].

2.2 Test soggettivi

Per garantire risultati accurati e riproducibili, le condizioni di test e le operazioni devono essere definite in modo chiaro e inequivocabile per poter essere riprodotte nel modo più fedele possibile. Gli standard rilevanti per questo scopo sono:

- ITU-R BT.500-14[5] (riguarda il sistema televisivo)
- ITU-T P.910 [6] (si occupa delle nuove applicazioni multimediali)

Gli standard specificano:

- condizioni ambientali (tipo di display, intensità della luminanza, ambiente di illuminazione, colore dello sfondo, ecc.).
- test per selezionare i soggetti partecipanti (per valutare le loro capacità visive, ad esempio, acuità visiva normale da entrambi gli occhi, corretta percezione dei colori, ecc.)
- procedure e scale di valutazione da utilizzare.

2.3 Confronto tra test soggettivi e oggettivi

I test soggettivi forniscono le migliori valutazioni ma sono costosi, richiedono molte risorse e sono difficili da ripetere in modo perfetto.
Nel caso del video, ad esempio, è necessario predisporre una stanza con caratteristiche specifiche, come una certa luminosità e un dispositivo di visualizzazione calibrato in un certo modo.
Per ovviare a queste difficoltà, sono stati sviluppati algoritmi in grado di approssimare la valutazione della qualità in modo oggettivo e automatizzato.

2.4 Metodi Oggettivi

I metodi si possono classificare in:

- **Full reference (FR)** : si confrontano il video originale e quello processato per ottenere una valutazione di qualità. Hanno bisogno dei dati multimediali originali e non compressi.
- **Reduced reference (RR)**: sono necessarie solo una versione a bassa qualità del video originale, o alcune caratteristiche estratte dal video originale. Sono utili per scenari di trasmissione in cui i dati multimediali originali e non corrotti non sono disponibili al decoder.
- **No reference (NR)**: non c'è bisogno dei dati multimediali originali e non compressi poichè si lavora sul contenuto distorto. Sono molto meno affidabili rispetto ai metodi FR e RR. Sono disponibili pochi algoritmi.

2.5 Algoritmi full reference

Tra i più comuni algoritmi Full reference troviamo:

- **MSE e PSNR**
- **SSIM e varianti**
- **VMAF**

2.5.1 MSE E PSNR

Sono utilizzati per misurare la distorsione, cioè la differenza tra l'immagine ricostruita e l'originale. L'MSE (2.2) viene definito come la media delle differenze al quadrato tra i pixel corrispondenti delle due immagini da confrontare. Il PSNR (*Peak Signal-to-Noise Ratio*) misura il rapporto tra il valore di picco del segnale (il massimo valore possibile) e l'MSE.

$$\text{PSNR (dB)} = 10 \cdot \log_{10} \left(\frac{(2^N - 1)^2}{\text{MSE}} \right) \tag{2.1}$$

$$\text{MSE} = \frac{1}{M \cdot N} \sum_{x=1}^{M} \sum_{y=1}^{N} e^2(x,y) = \sum_{x=1}^{M} \sum_{y=1}^{N} \left(d(x,y) - i(x,y)\right)^2 \qquad (2.2)$$

Con

- i(x,y): immagine originale
- d(x,y): immagine ricostruita
- l'errore è dato da: $e(x,y) = |d(x,y) - i(x,y)|$
- M e N corrispondono alla larghezza e all'altezza in pixel di un singolo frame
- $2^n - 1$ valore di picco della luminanza per n bit

Il PSNR di un video è dato dalla media del PSNR per ogni singolo frame. Il PSNR viene misurato in dB. Valori maggiori di PSNR corrispondono ad una maggiore somiglianza con il frame originale. I valori tipici variano tra 20 e 50 dB. Scendendo sotto i 30dB la qualità è tipicamente molto bassa.

2.5.2 SSIM: Structural Similarity Index

È stato sviluppato da un gruppo di ricerca accademico ed è basato su semplici operazioni come i filtri passa-basso, le medie e le varianze. Cerca di catturare, per ogni parte dell'immagine, la sua somiglianza al riferimento in termini di presenza di contorni, valori medi, rumore, ecc.): caratteristiche che potremmo definire più **"percettive"**. Il suo punto di forza è quello di essere adatto a un'eventuale ottimizzazione "analitica" del processo di codifica (ad esempio, quantizzazione e modalità di codifica) in un modo percettivamente significativo[7]. Sono state sviluppate diverse varianti:

- V-SSIM per il video
- MS-SSIM (multiscala) per migliorare le prestazioni dell'originale SSIM, analizzando l'immagine a diverse risoluzioni
- eSSIM

2.5.3 eSSIM

eSSIM (***Edge-based Structural Similarity***) [8] è un'evoluzione di SSIM nata per risolvere il problema della valutazione di qualità per immagini gravemente sfocate. I risultati sperimentali hanno dimostrato una maggiore affidabilità di eSSIM rispetto a PSNR e SSIM in termini di correlazione con la percezione umana (HVS).

2.5.4 Netflix VMAF

Netflix ha progettato una propria metrica percettiva chiamata VMAF ***Visual Multimodal Assessment Fusion(VMAF)***[9]. La metrica è stata rilasciata come open source nel giugno 2016 per incoraggiare i ricercatori a migliorarne il sistema. VMAF è stata estesa a schermi 4K e dispositivi mobili e sta venendo ampliato a spazi colore più ampi (High Dynamic Range, Wide Color Gamut). In VMAF sono integrate diverse metriche tra cui:

- **VIF (Visual Information Fidelity)**: metrica usata per valutare la qualità delle immagini, basata sull'idea che la qualità dell'immagine sia legata alla quantità di informazione che viene mantenuta/persa rispetto all'immagine originale a seguito dei processi di compressione o deterioramento

- **DLM (Detail Loss Metric)**: una metrica che valuta separatamente la perdita di dettagli, che incide sulla chiarezza e visibilità del contenuto, e le distorsioni indesiderate, che possono distrarre l'attenzione dello spettatore.

- **Motion**: oltre alle due metriche sulle immagini viene introdotta la feature **motion** per tenere conto delle caratteristiche temporali dei video: misura la differenza assoluta media dei pixel per la componente di luminanza

Le metriche menzionate in precedenza vengono combinate utilizzando una regressione basata su **SVM (Support Vector Machine)**, che assegna un punteggio a ciascun frame su una scala da 0 a 100. Un valore di 100 indica una qualità equivalente a quella del fotogramma

di riferimento, mentre un punteggio di 70 è considerato soddisfacente. Valori pari o inferiori a 20 segnalano una qualità estremamente bassa. Infine, i punteggi di tutti i fotogrammi vengono aggregati e mediati per ottenere una valutazione complessiva del contenuto.

2.5.5 Dal PSNR a VMAF: verso una valutazione più accurata della qualità video

Il PSNR può essere una metrica utile in determinate situazioni, ma ha il limite di non tenere conto della percezione visiva umana, mancando quindi di una correlazione diretta con il modo in cui le persone percepiscono la qualità dell'immagine. In figura 2.1 le immagini hanno lo

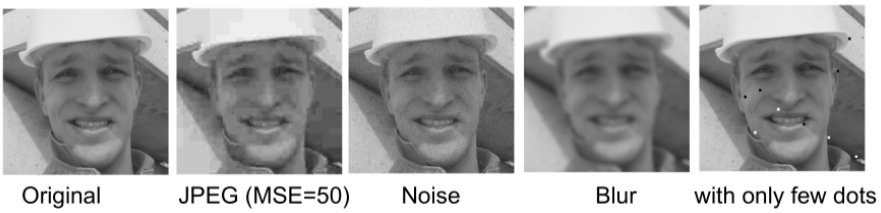

Figura 2.1: Immagini con uguale valore di PSNR e differente aspetto

stesso valore di PSNR ma un aspetto tendenzialmente diverso. L'immagine con i punti risulta più fastidiosa ad un occhio umano nonostante lo stesso punteggio. Il valore può avere significato differente anche in base al tipo di contenuto: in figura 2.2 [9] una differenza che risulta poco significativa nell'immagine a sinistra (per via della scena affollata e poco chiara) sarà molto più evidente nell'immagine a destra a maggiore focus su un singolo soggetto.
Con VMAF, Netflix ha voluto superare i limiti dei vecchi approcci tradizionali. La piattaforma stava cercando un modo per scegliere il miglior bitrate, in modo da offrire un servizio di buona qualità riducendo al minimo i costi. Fino al 2015 il bitrate ladder (raccolta di file video codificati a bit rate, risoluzioni e qualità differenti.) di Netflix era ancora fermo a dei valori fissi, indipendenti dal contenuto. Per fare ciò, è stato sviluppato un algoritmo per ottimizzare il bitrate sulla base dei contenuti[10]. Viene fatto un encoding offline che analizza ogni scena

Figura 2.2: Dipendenza dei valori di PSNR dal contenuto

e codifica ogni scena indipendentemente dalle altre con differenti parametri. Viene così creata la combinazione di scene a più alta qualità con la massima riduzione di banda. La piattaforma non disponeva, però, di una metrica affidabile per la valutazione della qualità basata sulla percezione finale dell'utente.
Sviluppando in home, la piattaforma:

- ha avuto a disposizione un data set molto grande per fare il training
- ha ottenuto un vantaggio commerciale nato dalla possibilità di scalare il processo di valutazione della qualità senza preoccuparsi dei costi di licenza crescenti (ad esempio il software)

La metrica è risultata affidabile. In figura 2.3 [9]: viene mostrato un confronto in termini di valutazioni rispetto a quelle del Mean Opinion Score.

2.5.6 Il problema della qualità

Non tutte le problematiche sono state risolte da VMAF. Passando alla codifica per scene:

- è noto che la percezione è maggiormente influenzata dalla qualità peggiore
- si è ancora alla ricerca di buone strategie di "temporal pooling"

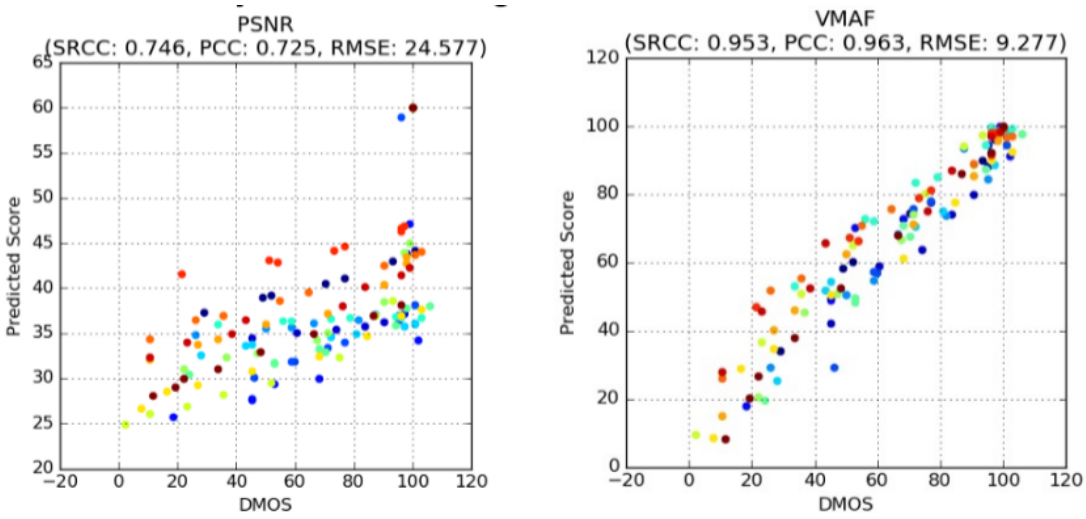

Figura 2.3: Confronto tra PSNR VS MOS e VMAF e MOS

2.6 Affidabilità delle misure oggettive e necessità di studiarne la correlazione con il MOS

Le misure oggettive, pur essendo utili per la valutazione della qualità video, non sono sempre completamente affidabili, in quanto la loro efficacia dipende molto dai contenuti specifici analizzati (come visto per il PSNR). Ad esempio, una misura potrebbe dare risultati diversi a seconda della risoluzione dell'immagine o del codificatore utilizzato. I contenuti video possono variare significativamente in termini di complessità spaziale (dettagli) e temporale (movimento). Ad esempio, una sequenza con molti dettagli e texture complesse può risultare più difficile da codificare rispetto a una sequenza con aree uniformi e pochi dettagli. Allo stesso modo, un video con un movimento rapido e complesso può presentare sfide maggiori per i codificatori rispetto a una scena statica o con movimenti lenti. Per ottenere una valutazione più accurata e coerente con la percezione umana, è fondamentale analizzare una varietà di contenuti, testati in diverse risoluzioni e con diversi algoritmi di codifica. Solo così si può studiare la correlazione tra queste misure oggettive

e il punteggio medio di opinione (MOS), che rappresenta la valutazione umana della qualità. Per garantire che i test siano rappresentativi, è importante selezionare sequenze video che coprano un'ampia gamma di scenari. Un utile strumento per valutare la complessità dei contenuti video è il diagramma SI/TI (Spatial Information/Temporal Information), che misura rispettivamente la complessità spaziale e temporale di una sequenza video. Idealmente, le sequenze selezionate per i test dovrebbero coprire un'ampia gamma di valori SI e TI, in modo da rappresentare diverse combinazioni di complessità spaziale e temporale. Non è sufficiente limitarsi a poche sequenze con caratteristiche simili, poiché ciò potrebbe portare a risultati parziali e non rappresentativi. La scelta di sequenze diversificate permette di valutare meglio le prestazioni dei codificatori e la correlazione tra misure oggettive e valutazioni soggettive.

2.7 Il progetto IGVQM

L'elaborato prende spunto dal documento di lavoro [11] **VQEG's Implementer's Guide to Video Quality Metrics (IGVQM) project**. Il progetto IGVQM [12] è ora parte integrante del progetto JEG-Hybrid [13], all'interno del gruppo di studio internazionale VQEG (Video Quality Experts Group) [14].

Nel progetto sono stati affrontati alcuni dei punti richiesti, quali:

- affrontare solo i difetti legati alla compressione video e al ridimensionamento.
- esaminare metriche oggettive full reference e tutte le metriche disponibili calcolate dalla libreria VMAF
- utilizzare metodi di aggregazione temporale delle metriche di qualità dell'immagine (come PSNR e SSIM) in metriche di qualità video.

Capitolo 3

Framework automatizzato per valutazioni di qualità su dataset di grandi dimensioni

In questo capitolo si presenterà il framework automatizzato impiegato per la valutazione della qualità, illustrando i dataset analizzati, i tools utilizzati e l'architettura del sistema di automazione adottato.

3.1 Dataset analizzati

Sono stati utilizzati cinque differenti database, ognuno con caratteristiche differenti. Si è scelto di analizzare per ogni database solo le sequenze per le quali il corrispondente valore di MOS è disponibile.

3.1.1 AVT-VQDB-UHD-1

Il dataset [15]:

- presenta 17 differenti sequenze (ne sono state analizzate 12) di video sorgente in 4K e a 60 fps
- contiene video codificati utilizzando tre diversi codec video (H.264, HEVC, VP9).
- contiene video compressi la cui risoluzione varia tra i 360p e i 2160p con framerates varianti tra 15 e 60fps
- è a sua volta stato suddiviso in quattro differenti casi di test, per i quali sono state eseguite valutazioni soggettive differenti. Tutte le quattro categorie di test seguono un approccio di codifica a fixed bitrate 2-pass encoding tramite ffmpeg 3.2.2.
- contiene le valutazioni soggettive con annessi intervalli di confidenza

Figura 3.1: Esempio di sequenza compressa per il dataset AVT-VQDB-UHD-1: bigbuck_bunny_8bit

Sono ora rappresentate una serie di tabelle [15] 3.2, 3.3 e 3.4: descrivono le differenti coppie risoluzioni-bitrate per i differenti dataset per i quali sono state eseguite delle valutazioni.

Parameter	Value
Duration	8/10 sec
Resolution	360p, 480p, 720p, 1080p, 1440p, 2160p
Frame Rate	24 fps
Encoders	H.264

Tabella 3.1: Breve descrizione delle sequenze compresse all'interno del dataset ITS4S

Resolution	Bitrate [kbit/s]					
360p	200	750				
720p		750	2000			
1080p			2000	7500	15000	
2160p				7500	15000	40000

Figura 3.2: Test Design - Subjective Test 1

Resolution	Bits-per-pixel (bitrate in kbps)			
360p	0.007 (97)	0.0447 (617)	0.0823 (1138)	0.12 (1659)
720p	0.007 (387)	0.0447 (2470)	0.0823 (4553)	0.12 (6636)
1080p	0.007 (871)	0.0447 (5557)	0.0823 (10244)	0.12 (14930)
2160p	0.007 (3484)	0.0447 (22229)	0.0823 (40974)	0.12 (59720)

Figura 3.3: Test Design - Subjective Test 2 e 3

Resolution	Bitrate [kbit/s] (framerate)			
360p	200 (15)	500 (15)	500 (24)	1000 (24)
480p	500 (15)	1000 (15)	1000 (24)	2000 (24)
720p	1000 (24)	2000 (24)	2000 (30)	4000 (30)
1080p	2000 (24)	4000 (24)	4000 (30)	6000 (30)
1440p	4000 (30)	6000 (30)	6000 (60)	8000 (60)
2160p	6000 (30)	8000 (30)	8000 (60)	15000 (60)

Figura 3.4: Test Design - Subjective Test 4

3.1.2 GamingVideoSet

GamingVideoSet[1] contiene:

- ventiquattro video non compressi, di durata pari a 30 secondi, risoluzione 1080p e 30fps, ottenuti a partire da 12 famosi videogiochi, scelti in base a genere e popolarità. Sono state introdotte due scene per ogni gioco: **Counter Strike: Global Offensive (CSGO)**,**Diablo III**, **Defense of the Ancients 2 (Dota2)**,**FIFA 2017 (FIFA)**, **H1Z1: Just Survive (H1Z1)**,**Hearthstone (HS)**,**Heros of the Storm**,**League of Legends (LoL)**,**Project Cars (PC)**,**PlayerUnknown's Battleground (PUBG): PlayerUnknown's Battlegrounds (PUBG)**,**Starcraft 2 (SC)**, **World of Warcraft (WoW)**
- un totale di 576 video distorti in formato mp4, ottenuti codificando le ventiquattro sequenze con diverse combinazioni di risoluzioni e bitrate
- i risultati della valutazione soggettiva della qualità per 90 sequenze video, ottenute codificando diversi video utilizzando lo standard del codec H.264/MPEG-AVC in 15 diverse combinazioni di risoluzione-bitrate (tre risoluzioni, cinque bitrate ciascuna)
- in figura 3.5 sono riassunti i principali parametri di codifica e in figura 3.6[1] le coppie risoluzione-bitrate contenute all'interno del dataset

Parameter	**Value**
Duration	30 sec
Resolution	1080p, 720p, 480p
Frame Rate	30
Encoder	FFmpeg
Encoding Mode	CBR
Video Compression Standard	H.264, Main 4.0
Preset	veryfast

Figura 3.5: Parametri scelti per l'encoding di GamingVideoSet[1]

Resolution	Bitrate (kbps)
1080p	**600**, **750**, 1000, **1200**, 1500, **2000**, 3000, **4000**
720p	**500**, **600**, 750, 900, **1200**, 1600, **2000**, 2500, **4000**
480p	**300**, 400, **600**, 900, **1200**, **2000**, **4000**

Figura 3.6: Coppie risoluzioni-bitrate disponibili, in grassetto quelle per cui è disponibile la valutazione soggettiva

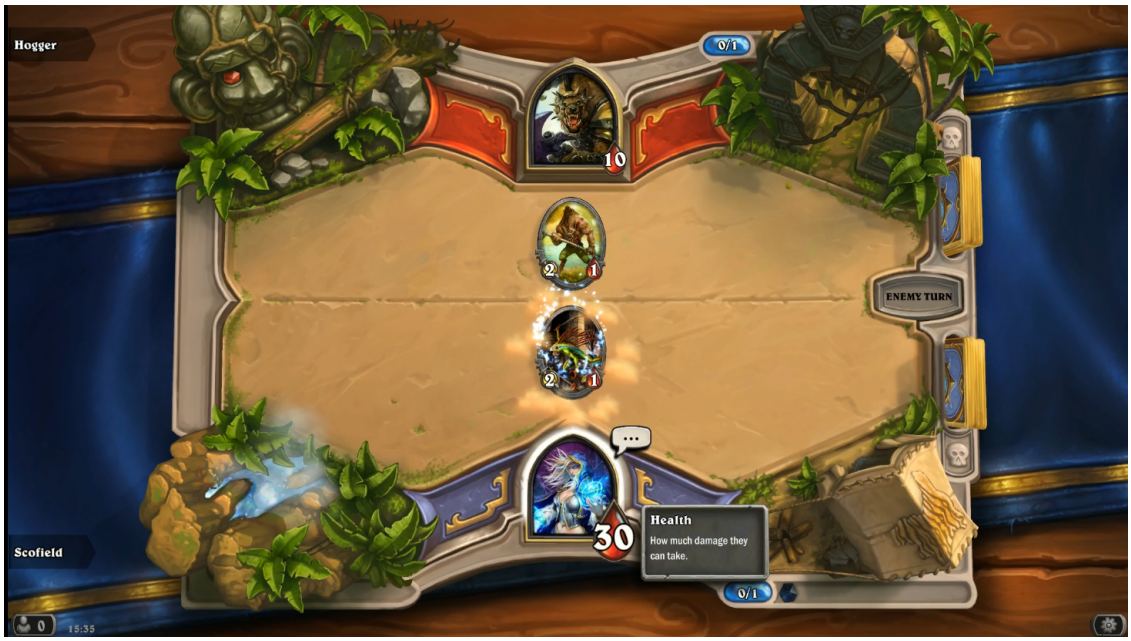

Figura 3.7: Esempio di sequenza compressa per il dataset GamingVideoSet: Heartstone

3.1.3 KUGVD

Per KUGVD (Kingston University Gaming Video Dataset) [16]

- sono stati scelti sei video di gioco e utilizzate le stesse impostazioni di codifica del dataset GamingVideoSet per ventiquattro coppie di risoluzione e bitrate (le stesse di GamingVideoSet)
- per 90 sequenze video sono disponibili i risultati della valutazione soggettiva
- il gioco CSGO è stato mantenuto in entrambi i dataset, quattro giochi selezionati sono gli stessi (FIFA17, H1Z1, HS e LoL), ma è stato

considerato una parte diversa (scenario) del gioco. A seconda della fase del gioco è infatti possibile avere una complessità del contenuto molto diversa. E' stato introdotto un nuovo gioco Overwatch, sparatutto in prima persona molto popolare sulle piattaforme di streaming e di alta complessità.

Figura 3.8: Esempio di sequenza compressa per il dataset KUGVD: League of legends

3.1.4 ITS4S

Il dataset [17] contiene:

- 813 sequenze, ognuna di durata pari a 4 secondi
- per ogni sequenza originale ne esiste una sola corrispettiva compressa
- i video originali sono stati convertiti nel formato 720p a 24fps. La scelta dei 24fps è stata effettuata poiché rappresenta una frequenza di fotogrammi molto comune nei servizi di streaming adattivo attuali.

Figura 3.9: Esempio di sequenza compressa per il dataset ITS4S: Ocean_075-Osrc18J_2340K

Parameter	**Value**
Duration	4 sec
Resolution	1280x720,512×288,696×392,824×464,1024×576
Frame Rate	24 fps
Encoders	H.264
Bitrate (bps)	2340K, 1732K, 1256K, 0951K, 0512K

Tabella 3.2: Breve descrizione delle sequenze compresse all'interno del dataset ITS4S

3.1.5 AGH_NTIA_DOLBY

- il dataset [18] contiene 84 sequenze sorgente
- 207 sequenze compresse
- contiene sei sessioni. Ogni sessione effettua il confronto della qualità di 10 codifiche video (HRC). Le codifiche esaminate includono il video originale e vari bitrate per MPEG-2, H.264 e H.265.

Parameter	**Value**
Duration	8 sec
Resolution	720p
Frame Rate	24 fps
Encoders	H.264,H.265,MPEG-2
Bitrate (bps)	1M,2M,512K,7M

Tabella 3.3: Breve descrizione delle sequenze compresse all'interno del dataset AGH_NTIA_Dolby

Figura 3.10: Esempio di sequenza compressa per il dataset AGH_NTIA_Dolby: AGH-NTIA-Dolby_poolhall_MPEG2at7M

3.2 Flusso del progetto

I dataset sono molto grandi, ognuno con differenti sequenze video e caratteristiche differenti. È stato quindi sviluppato un framework per automatizzare l'esecuzione degli esperimenti e la raccolta dei risultati. Il lavoro si è focalizzato su:

- creazione di file JSON contenenti le descrizioni dei dataset e delle valutazioni soggettive per ogni database
- creazione di script per l'esecuzione di valutazioni oggettive tramite

il tool VMAF di Netflix [19] e raccolta dei risultati ottenuti per le diverse metriche disponibili. Sono state fatte delle valutazioni per la metrica eSSIM tramite il tool sviluppato da META eSSIM [20].

- raccolta dei risultati (file JSON prodotti dal tool VMAF e file csv prodotti dal tool eSSIM) in file .csv
  - è possibile fare raccolte parziali di valutazioni di qualità in base alle esigenze (solo alcune metriche ad esempio o si preferisce fare girare solo eSSIM e non VMAF o viceversa)
  - il sistema in automatico andrà ad aggiornare i risultati in base agli esperimenti fatti
- creazione di grafici per l'analisi dei risultati basati sui file .csv prodotti

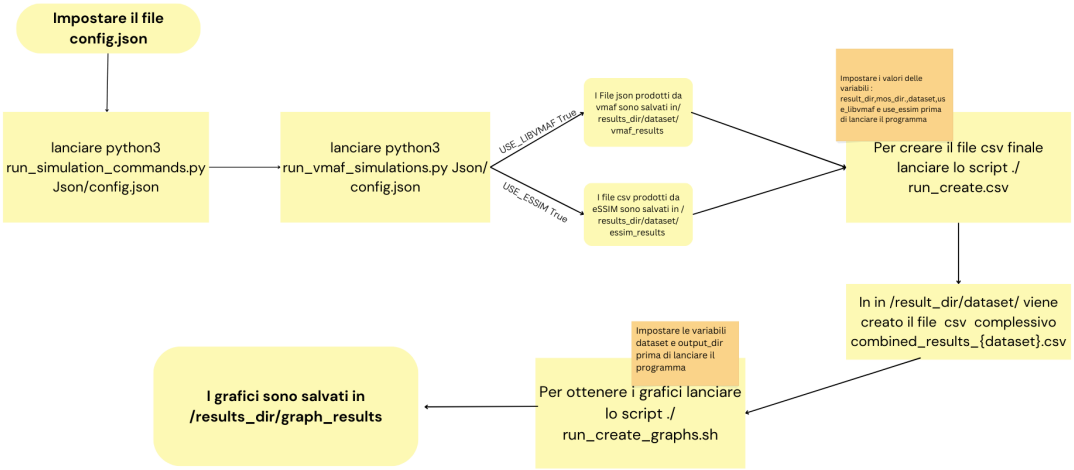

Figura 3.11: Rappresentazione del flusso

3.3 Tool utilizzati

3.3.1 VMAF

Per eseguire le valutazioni con VMAF [19], si è utilizzato un tool da linea di comando chiamato *vmaf*:

- effettua un confronto tra due video:
  - video sorgente (in formato .yuv o .y4m)
  - video distorto (in formato .yuv o .y4m)
- richiede in ingresso uno tra i modelli VMAF disponibili
- produce un output-log contenente punteggi VMAF per ogni frame e punteggi aggregati.

Il tool:

- si basa su libvmaf, una libreria C con cui è possibile integrare VMAF all'interno del codice
- presenta una serie di possibili opzioni aggiuntive tra le quali width, height, pixel_format e bitdepth per offrire informazioni sui video da analizzare se in formato .yuv.
- presenta l'opzione threads, con cui decidere il numero di threads da usare
- consente la scelta di uno tra i 9 modelli disponibili
- offre la possibilità di ottenere in output risultati per diverse metriche oltre a VMAF:
  - PSNR (Peak Signal-to-Noise Ratio)
  - PSNR-HVS (PSNR con la Sensibilità Visiva Umana)
  - FLOAT-SSIM (Structural Similarity Index) in versione floating point
  - FLOAT-MS-SSIM (Multiscale SSIM) in versione floating point
  - CIEDE2000 (Color Difference in CIE 2000)
  - CAMBI, un banding artifact detector

3.3.2 eSSIM

E' possibile ottenere anche i risultati per le metriche eSSIM e SSIM tramite l'implementazione del tool eSSIM sviluppato da Meta[20]. Il tool richiede in ingresso un video di riferimento, in formato yuv, ed un corrispettivo distorto (sempre in formato .yuv). Parametri aggiuntivi sono richiesti riguardanti le caratteristiche del video (width, height e bitdepth). I risultati prodotti possono essere salvati in file .csv. In merito al calcolo dei valori è possibile fissare valori differenti per quattro parametri:

- wsize, Window size for eSSIM: dimensione della finestra di calcolo
- wstride, Window stride for eSSIM: passo della finestra
- mink, SSIM Minkowski pooling: il pooling Minkowski per il calcolo del punteggio SSIM.
- mode: indica il tipo di modalità da utilizzare:
  - 0 per la modalità di riferimento (SSIM_MODE_REF),
  - 1 per la modalità di tipo intermedio (SSIM_MODE_INT),
  - 2 per la modalità di prestazioni (SSIM_MODE_PERF)

3.3.3 FFMPEG

Per eseguire gli esperimenti è stato necessario effettuare numerose conversioni per sottostare alle necessità di VMAF ed eSSIM. Per effettuare ciò si è scelto l'utilizzo di ffmpeg [21], un software open-source contenente una serie di librerie e programmi utili per la gestione di video, audio e tutto ciò che riguarda il campo dei multimedia. I due tool permettono il confronto solo per video della stessa dimensione e ciò ha richiesto i seguenti passaggi:

- Decodifica dei video compressi, ogni volta con parametri adattati rispetto alle esigenze. Un esempio è il comando:

```
ffmpeg -i "$INPUT_DISTORTED_DIR/$distorted" -pix_fmt yuv422p
    -f rawvideo "$distorted_decoded" -loglevel quiet
```

- ridimensionamento dei video decodificati alle dimensioni del video originale. Un esempio è il comando:

```
ffmpeg -i "$distorted_decoded"
-vf "'scale=1280x720:flags=lanczos'"
-sws_flags lanczos+accurate_rnd+full_chroma_int
-pix_fmt yuv420p "$distorted_decoded_resized"
```

  Per il rescaling è stato scelto **l'algoritmo di scaling di Lanczos**[22] con parametro 3. Il filtro risulta essere una ottima approssimazione del filtro di ricostruzione teoricamente ideale delle immagini: la funzione Sinc [23] per segnali a banda limitata. Viene considerato il "miglior compromesso in termini di riduzione dell'aliasing, nitidezza e riduzione del ringing" [24] rispetto ad altri metodi di interpolazione per segnali discreti e in particolare altre approssimazioni della funzione sync, per l'interpolazione di immagini bidimensionali. Sono state abilitate le opzioni accurate_rnd e full_chroma_int:

  – -vf scale=1280x720:flags=lanczos:param0=3: ridimensiona il video alle dimensioni passate come parametro utilizzando l'algoritmo Lanczos con parametro 3.

  – sws_flags lanczos+accurate_rnd+full_chroma_int: specifica l'uso dell'algoritmo Lanczos, abilitando il rounding accurato (accurate_rnd) e l'interpolazione completa della crominanza (full_chroma_int).

- A seguito dell'utilizzo del tool eSSIM è stato necessario convertire video in formato .y4m in .yuv in quanto unico formato supportato.

Sono state effettuate altre conversioni aggiuntive:

- conversione dei video originali dal formato yuv 4:2:2 a yuv 4:2:0 per il dataset ITS4S

- conversione dei video originali per AVT-VQDB-UHD-1_1 da 10-bit depth a 8 bit-depth per il confronto con i corrispettivi video distorti a 8-bit depth

- conversione dei video originali di AVT-VQDB-UHD-1_4 a frame rate differente (da 60.0 fps a 30.0 o 15.0 fps) per il confronto con i corrispettivi video distorti

3.4 Design del sistema di automazione degli esperimenti

3.4.1 Podman

Per lo sviluppo del progetto, è stato scelto l'utilizzo di Podman. Tra gli obiettivi del lavoro, vi è la necessità di garantire la riproducibilità degli esperimenti in modo affidabile e coerente, un aspetto che viene assicurato da Podman.
Sebbene sia Podman che Docker siano strumenti equivalenti per la gestione dei containers si è scelto Podman [25] poichè:

- non richiede privilegi di root, il che lo rende particolarmente adatto per ambienti condivisi
- non avendo la necessità di utilizzare funzionalità avanzate legate alla rete o configurazioni complesse, si è rivelato uno strumento semplice e sicuro, garantendo la gestione dei container senza la necessità di operare come superutente.
- è inoltre compatibile con Docker in termini di sintassi dei comandi e script
- non presenta un "demone centrale", ovvero non dipende da un processo con privilegi di root per eseguire i container. Questo garantisce migliori risultati in termini di sicurezza eliminando un obiettivo particolarmente gradito agli attaccanti (prendendo il controllo del demone, dotato di privilegi da superutente, è possibile leggere file, installare programmi, modificare applicazioni e altro ancora).

E' stato prodotto un Dockerfile che crea un immagine basata su Python 3.12(Bookworm)[26] e configura l'ambiente per l'analisi della qualità video usando Python, VMAF, FFMPEG ed eSSIM.
Il dockerfile:

- installa i pacchetti necessari
- installa il tool VMAF [19]

- installa ffmpeg [21]
- installa il tool eSSIM [20]
- crea le cartelle necessarie per l'analisi:
  - /reference: directory dove trovare i video originali
  - /distorted: directory dove trovare i video distorti
  - /results: directory dove verranno salvati i risultati dell'analisi
  - /hash: directory dove vengono salvati gli hash dei file
  - /mos: directory dove trovare i file JSON contenenti i punteggi MOS (Mean Opinion Score) dei vari dataset presi in considerazione
- copia lo script *run_experiments.sh* nella directory app e lo rende eseguibile
- avvia una shell (/bin/sh) quando il container viene eseguito

3.4.2 Dataset e MOS

L'analisi è partita dalla necessità di avere le informazioni dei dataset in un formato comodo e fruibile. Per i dataset e i valori di MOS è stato scelto il formato JSON.

Dataset

Per ogni dataset tramite script python si sono estratte le informazioni dai nomi dei file all'interno dei server quando significativi o tramite il software linux mediainfo quando assenti. Sono state rimosse dall'analisi tutte le sequenze mancanti di informazioni sul relativo MOS. Ogni file descrizione del dataset contiene tre sezioni:

- **database**: un campo che descrive il database in esame
- **reference_videos**: contiene la lista dei video originali utilizzati per le analisi, ognuno di essi con due campi:

  - **id**: identificatore univoco del video sorgente
  - **file_name**: nome del file video distorto, comprensivo di estensione.

- **distorted_videos**: contiene la lista dei video distorti utilizzati per le analisi, ognuno dei quali con una serie di campi utili:
  - **id**: identificatore univoco del video distorto
  - **file_name**: nome del file video, comprensivo di estensione.
  - **width**: larghezza del video in pixel.
  - **height**: altezza del video in pixel.
  - **bitrate**: bitrate del video, espresso in bit per secondo (bps).
  - **video_codec**: codec video utilizzato per la compressione.
  - **bitdepth**: profondità di bit per pixel (es. 8-bit, 10-bit).
  - **pixel_format**: formato di rappresentazione dei pixel (es. YUV420, YUV422).
  - **fps**: numero di frame al secondo
  - **duration**: durata totale del video, espressa in secondi.

MOS

Le informazioni sono state estratte dai documenti contenuti all'interno dei database, ognuno con un diverso formato. Per ogni dataset sono state raccolte informazioni sulle sequenze per ogni PVS (Processed Video Sequences) in modo differente in base alle informazioni contenute nel documento. Per ogni database sono state estratte una serie di informazioni significative:

- **KUGVD, GamingVideoSet, AVT-VQDB-UHD**:
  - **dataset_name**: nome del dataset utilizzato.
  - **SRC_names**: elenco dei nomi dei video sorgente (SRC - Source).
  - **PVS_bitrates**: elenco dei bitrate utilizzati per i video distorti (PVS - Processed Video Sequences).

  - **PVS_encoders**: elenco degli encoder utilizzati per la compressione dei video PVS (es. x264).
  - **PVS_resolutions**: elenco delle risoluzioni utilizzate per i video PVS.
  - **scores**: campo contenente il punteggio MOS e il campo computed_mos calcolato come media tra tutte le valutazioni soggettive OS (Opinion Scores), se disponibili come ad esempio per KUGVD.

- **ITS4S**: sono state estratte informazioni aggiuntive come:
  - **Sessions**: gli esperimenti sono stati effettuati in sessioni differenti
  - **SRC_Names**: sequenze video sorgenti
  - **HRC_Names**: test conditions (Hypothetical reference circuits)
  - **SRC_Description**: descrizioni delle sequenze video sorgenti
  - **HRC_Description**: descrizioni delle varie HRC
  - **scores**: campo contenente questa volta le informazioni sulle varie sequenze PVS e per ognuna MOS, SOS e OS(Valutazioni soggettive dei singoli osservatori) oltre al computed_MOS(Mos calcolato sulla base delle valutazioni soggettive disponibili)

- **AGH_NTIA_Dolby**: sono state estratte come informazioni:
  - **dataset_name**
  - **SRC** : sequenze sorgenti
  - **scores**: contenente questa volta per ogni PVS:
    * **PVS_ID e SRC**: identificativo della PVS e della sequenza sorgente
    * **Subject_right**: valutazione soggettiva corretta
    * **MOS, OS e computed_MOS**

È stato necessario standardizzare il campo PVS_ID per tutte le sequenze di tutti i dataset, poichè fulcro delle analisi successive dei risultati(verrà utilizzato in uno script successivo per ricavare i valori dei MOS delle sequenze).

3.4.3 JSON

config.json

Per l'analisi è stato predisposto un file JSON contenente una serie di campi da compilare sulla base degli esperimenti che si è interessati ad eseguire:

- **IMAGE_NAME**: nome dell'immagine Podman compilata del contenitore nel quale verranno eseguiti gli esperimenti
- **INPUT_REF_DIR**: Nome della directory contenente i video di riferimento (Reference Videos), ovvero i video originali
- **INPUT_DIST_DIR**: Directory contenente i video distorti (Distorted Videos), ovvero le versioni compresse dei video di riferimento utilizzate per l'analisi.
- **OUTPUT_DIR**: Directory in cui vengono salvati i risultati dell'analisi.
- **HASH_DIR**: Directory contenente gli hash calcolati per i video
- **MOS_DIR**: Directory dove trovare i file JSON con i valori dei MOS per ogni dataset
- **DATASET_DIR**: Directory contenente i file descrizione dei dataset utilizzati per le analisi
- **REFERENCE_VIDEO_LIST_DIR**: Directory contenente i file descrizione dei vari file sorgenti per i differenti dataset
- **JSON_DIR**: Directory contenente il file di configurazione e il file JSON contenente lo schema per verificare il file di configurazione
- **ORIGINAL_VIDEO** Nome del video originale di riferimento su cui viene eseguita l'analisi
- **MODEL_VERSION**: Lista dei modelli VMAF (Video Multi-Method Assessment Fusion) utilizzati per l'analisi della qualità video. Il campo può contenere uno o più valori, in base alla necessità

di eseguire l'analisi su un solo modello, su più di uno (listando i vari modelli) o su tutti e 9 (tramite l'utilizzo dell'opzione VMAF_ALL)

- **DATASET**: Nome del dataset scelto per l'analisi
- **FEATURES**: Lista delle metriche di qualità video utilizzate. E' possibile l'utilizzo di:
  - cambi
  - SSIM
  - PSNR
  - float_ms_ssim
  - ciede
  - psnr_hvs
- **USE_LIBVMAF**: Indica se viene utilizzata la libreria libvmaf per il calcolo della qualità video (*true se attiva, false se non attiva*)
- **USE_ESSIM**: Indica se viene utilizzato il tool per calcolare la metrica eSSIM(*true se usato, false se non usato*)
- **ESSIM_PARAMETERS**: Lista di configurazioni per il calcolo della metrica ESSIM:
  - Window_size
  - Window_stride
  - SSIM_Minkowski_pooling
  - Mode

configschema.json

File contenente lo schema JSON necessario ad evitare errori durante l'analisi, come l'utilizzo di dataset non consentiti, features non utilizzabili o parametri eSSIM non validi.

3.4.4 Generare i comandi

Partendo dalla configurazione presente in config.json è possibile utilizzare lo script *create_commands.py* per generare i comandi da lanciare per eseguire l'analisi. Viene eseguito tramite il comando "python3 create_commands.py Json/config.json" Lo script:

- legge le informazioni dal file di configurazione
- crea, se non esistono, le cartelle necessarie
- valida lo schema JSON
- partendo dal video originale di partenza, per ogni video distorto contenuti nella cartella INPUT_DIST_DIR:
  - cerca se il nome del video originale è contenuto all'interno del nome del file distorto tramite un'espressione regolare
  - se presente, estrae le informazioni del video dal file JSON che rappresenta il dataset
  - chiama una funzione che genera il comando corrispondente. Il comando viene generato per tutti i modelli contenuti nella lista presente all'interno del file di configurazione.

  E' stato scelto di eseguire l'analisi per tutte le features per il modello vmaf_v0.6.1 (i risultati sarebbero uguali scegliendo un altro dei 9 modelli possibili). Tramite il comando vengono montate le directory locali all'interno del container sulle directory create all'interno del Dockerfile.

Generare tutti i comandi per un dataset

Per eseguire l'analisi è stato creato lo script *run_simulation_create_commands.py.* Viene eseguito con il comando "python3 run_simulation_create_commands.py Json/config.json" Lo script:

- legge da un file dataset_reference_video_list.txt. specifico per ogni dataset contenuto all'interno di REFERENCE_VIDEO_LIST_DIR, la lista dei video sorgente per cui si vuole effettuare l'analisi
- genera tutti i comandi andando a chiamare per ogni video sorgente *create_commands.py*
- cancella, se già presente per precedenti operazioni, il file con i comandi corrispondente

Il risultato è un file commands_database.txt contenente la serie di comandi da lanciare.

3.4.5 Lanciare la simulazione

Ogni comando, lanciato con i rispettivi parametri, esegue lo script **run_experiment.sh**. Lo script:

- esegue la decodifica del file distorto
- prepara i file originali da trattare in maniera differente per i casi particolari
- viene eseguito il resizing del video decodificato quando le dimensioni non coincidono con quelle del video originale
- calcola l'hash del file decodificato, lancia un warning se differente rispetto a quello già calcolato in precedenza ed effettua il corretto resizing
- in base ai valori di USE_LIBVMAF e USE_ESSIM
  - esegue la valutazione VMAF andando a effettuare il confronto tra video originale e decodificato e con parametri differenti in base ai dataset: aggiungendo informazioni su width, height e bitdepth per video .yuv e non per i video .y4m producendo in output un file JSON con i risultati dell'analisi
  - esegue eSSIM con i parametri specificati nel file di configurazione e producendo in output un file csv con nome significativo ricavato dai parametri dell'analisi

– esegue la cancellazione dei file intermedi prodotti quando arrivati alla fine dell'analisi (ultimo modello all'interno della lista)

Al fine di performare più velocemente l'analisi è possibile lanciare lo script python *run_vmaf_simulation.py*. Esegue ogni comando di shell contenuto all'interno del file commands_database.txt

3.4.6 Analisi dei risultati

Per analizzare i risultati è possibile utilizzare lo script *run_results_script.py*. Lo script percorre le cartelle contenenti i file risultato dell'analisi e per ogni file genera un comando che andrà a chiamare lo script *analyze.py*. Lo script analyze.py:

– crea un file .csv con n righe in base al numero di temporal pooling scelti per ogni singolo file distorto. Le colonne contengono la serie di informazioni utili sul file come width, height, ecc, il MOS, n colonne per le features possibili e una serie di colonne per l'analisi eSSIM, in numero variabile in base alle configurazioni eSSIM scelte durante l'analisi

– riempie il file .csv con i risultati. I file JSON vengono letti cosi come i file eSSIM e trasformati in dataframe tramite l'utilizzo delle librerie python numpy e pandas. Dai dataframe estratti vengono calcolati, per tutte le features i valori dei pooling

Per eseguire l'analisi, è stato creato uno script di shell denominato *run_create_csv.sh*. E' necessario impostare alcune variabili quali la directory contenente i risultati e i file JSON contenenti i valori MOS. E' possibile impostare di valutare solo i risultati prodotti da VMAF o solo i risultati di eSSIM tramite due flag. Lo script si occupa di preparare un ambiente virtuale Python specifico per il progetto, utilizzando i requisiti definiti nel file *requirements.txt*. Quest'ultimo contiene l'elenco delle librerie Python necessarie per eseguire correttamente l'analisi dei risultati. La scelta di utilizzare un ambiente virtuale è stata motivata dalla necessità di isolare

le dipendenze del progetto. In questo modo, è possibile mantenere stabili e compatibili le versioni delle librerie richieste, evitando potenziali conflitti con altre librerie o versioni diverse che potrebbero essere presenti nel sistema o in altri progetti. Questo approccio garantisce che l'ambiente di lavoro rimanga coerente nel tempo, riducendo il rischio di errori dovuti a cambiamenti esterni o incompatibilità future.

3.4.7 Creare i grafici

Per la creazione dei grafici è stato sviluppato lo script *run_create_graphs.sh*. Anche in questo caso la scelta fatta è stata quella di utilizzare un ambiente virtuale, con requisiti definiti all'interno del file requirements.txt. Lo script, crea se non esiste, un ambiente virtuale e lancia lo script *graph_script.py*. E' necessario definire due variabili per definire la cartella dove trovare i risultati delle valutazioni VMAF ed eSSIM e il dataset per cui si decide di creare i grafici.

3.4.8 graph_script.py

Lo script:

- definisce i limiti degli assi per le varie metriche di qualità video
- mappa i pooling temporali ad i vari simboli grafici
- legge i risultati delle valutazioni precedenti, salvate in un .csv apposito
- identifica i modelli VMAF, le metriche di valutazione oggettiva e i valori di pooling temporale da analizzare
- crea le cartelle di output per salvare i risultati dei grafici

I grafici prodotti sono i seguenti:

- un grafico per ogni feature presente in features (elenco delle metriche di qualità video prese in considerazione)

    - per ogni valore PVS, estrae il valore della media della feature, il valore di MOS corrispondente e genera un scatterplot tra MOS e la feature selezionata.
    - ne è prodotto anche un altro che plotta tutti i pooling e non solo la media
- un grafico per ogni modello VMAF in vmaf_models (elenco dei modelli VMAF), ripetendo il processo precedente ma in questo caso visualizzando i valori previsti dai modelli VMAF in funzione del MOS per ogni PVS
- un grafico che plotta per differenti pooling (media, media armonica e media geometrica) i valori dei vari modelli VMAF ognuno rispetto al MOS
- un grafico che plotta per differenti pooling (media, mediana, norma l1, norma l2, norma l3) i valori dei vari modelli VMAF ognuno rispetto al MOS
- un grafico error_plot con relativi limiti di confidenza (lo/hi) per il modello "float b" di VMAF (vmaf_float_b_v0.6.3) plottando il valore della media
- un grafico error_plot con relativi limiti di confidenza (lo/hi) per il modello "float b" di VMAF (vmaf_float_b_v0.6.3) ma con valore medio centrato rispetto alla deviazione standard (viene presa l'informazione da vmaf_float_b_v0.6.3_stddev). Viene plottato il valore della media
- altri due grafici simili ai due precedenti ma questa volta per il modello "b" di VMAF (vmaf_b_v0.6.3)
- un grafico che grafici per i vari modelli VMAF utilizzano una error_bar con valori estremi 5 e 95 esimo percentile e valore centrale media o mediana (50-esimo percentile) (sullo stesso grafico)
- un grafico che confronta il valore della media di due modelli rispetto al MOS

Capitolo 4

Analisi dei risultati

4.1 Confronto tra metriche oggettive e soggettive

Il capitolo analizza gli esiti dei confronti tra valutazioni oggettive e soggettive. L'analisi delle metriche oggettive rispetto al Mean Opinion Score (MOS) è fondamentale per valutare i risultati delle valutazioni prodotte da metodi oggettivi.

4.1.1 Correlazione con il MOS

Per esaminare la correlazione tra le metriche oggettive e il MOS, sono stati realizzati una serie di scatter plot.

Metrica di qualità vs MOS

- in ascissa viene rappresentato il valore del MOS (con valore compreso tra 1 e 5)
- in ordinata sono rappresentati i valori di una particolare metrica di qualità rispetto alla quale sono stati estratti i valori (limitata rispetto ai valori possibili)
- sono rappresentate tutte le diverse PVS di un particolare dataset e per ogni PVS viene rappresentato il valore della media

Sono ora rappresentati nelle figure successive una serie di grafici per i differenti database relativi al confronto tra 3 metriche oggettive (PSNR, eSSIM, VMAF) ed il Mean Opinion Score.

Nel grafico 4.1 viene rappresentato il confronto tra PSNR e MOS. Si può osservare che all'aumentare del PSNR, il MOS tende a migliorare, ma con risultati che non sono sempre allineati con la percezione umana. In figura 4.2 viene invece raffigurato il confronto tra eSSIM , versione migliorata dell'SSIM. La configurazione utilizzata per eSSIM nella rappresentazione è quella predefinita, con i seguenti parametri: la dimensione della finestra (`wsize`) impostata a 8, il passo della finestra (`wstride`) pari a 4, il pooling di Minkowski (`mink`) con valore 3 e modalità (`mode`) 2. Nel grafico, viene mostrata una correlazione più forte tra eSSIM e MOS rispetto al PSNR, con un miglioramento più evidente del MOS all'aumentare dell'eSSIM. Infine in figura 4.3 il confronto è tra VMAF (per il modello vmaf_v0.6.1) e MOS. Nel grafico, viene mostrata una correlazione molto forte tra VMAF e MOS, l'andamento è quasi lineare. Ciò indica che all'aumentare del VMAF, il MOS migliora significativamente. La correlazione più forte corrisponde anche a punti meno dispersi rispetto al PSNR. In un confronto complessivo il VMAF mostra la correlazione più forte con il MOS, seguito da eSSIM e poi da PSNR. Tutte e tre le metriche mostrano una relazione positiva con il MOS ma VMAF risulta la più affidabile nel riflettere la qualità percepita dagli utenti, seguita da eSSIM e poi da PSNR.

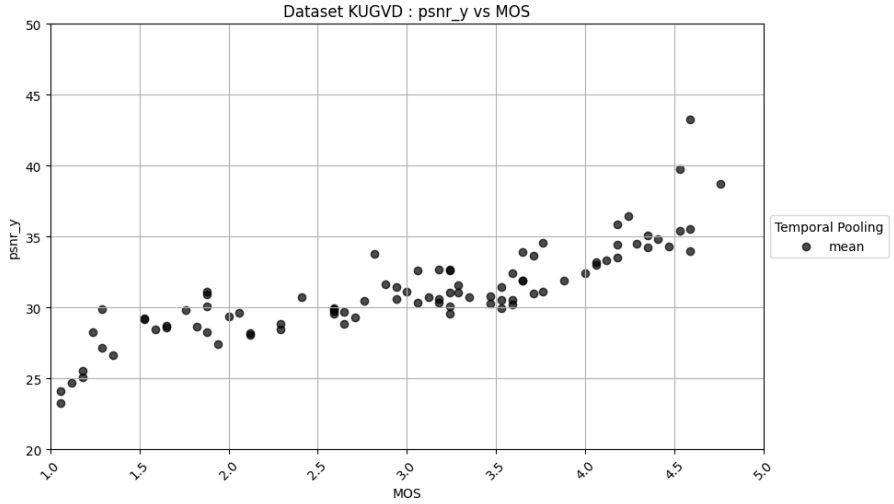

Figura 4.1: PSNR_y VS MOS per il dataset KUGVD

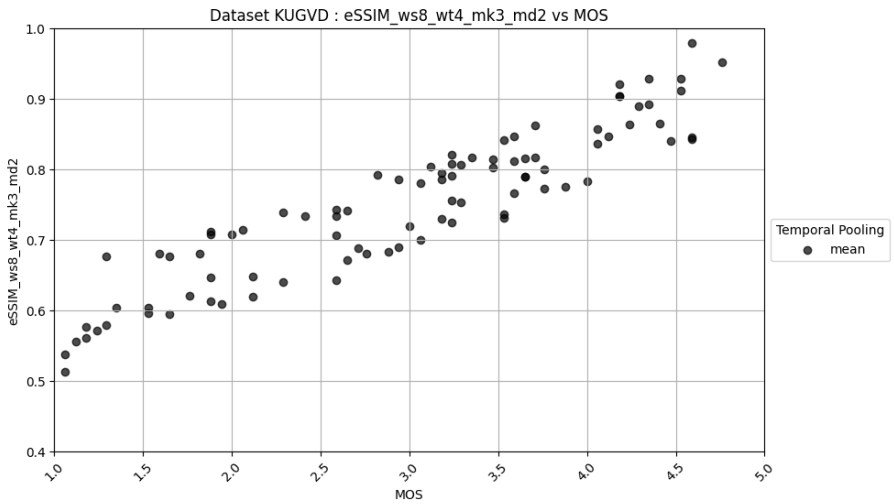

Figura 4.2: eSSIM(wsize:8,wstride:4, mink:3, mode:2) VS MOS per il dataset KUGVD

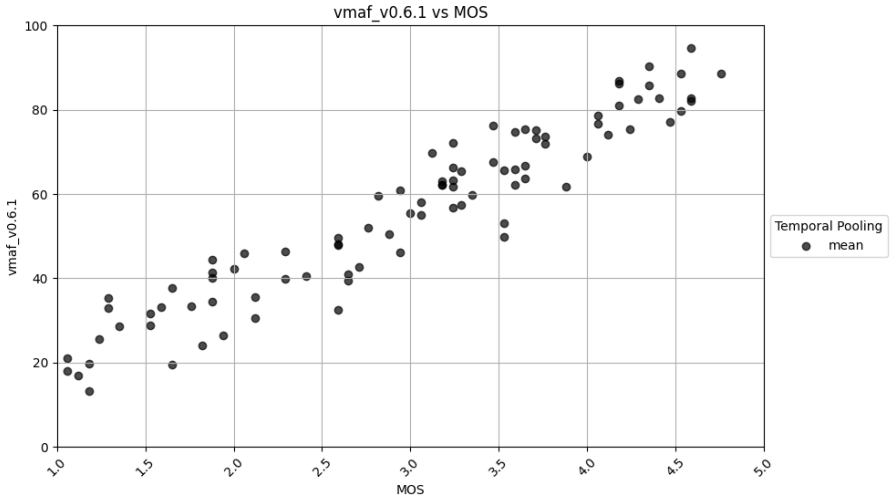

Figura 4.3: VMAF(modello vmaf_v0.6.1) vs MOS per il dataset KUG-VD

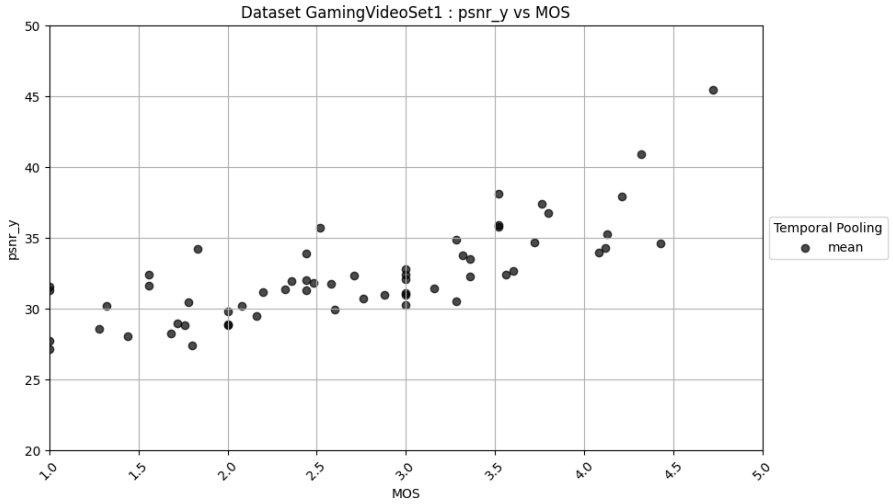

Figura 4.4: PSNR_y VS MOS per il dataset GamingVideoSet

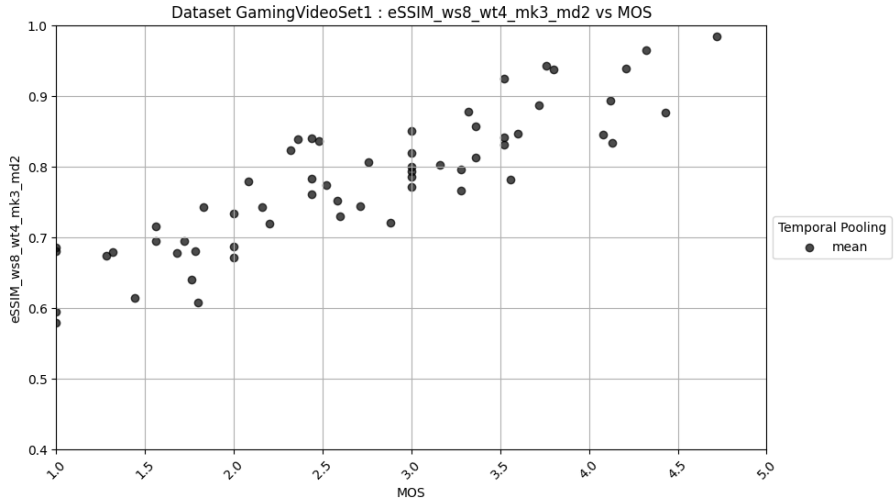

Figura 4.5: eSSIM(wsize:8,wstride:4, mink:3, mode:2) VS MOS per il dataset GamingVideoSet

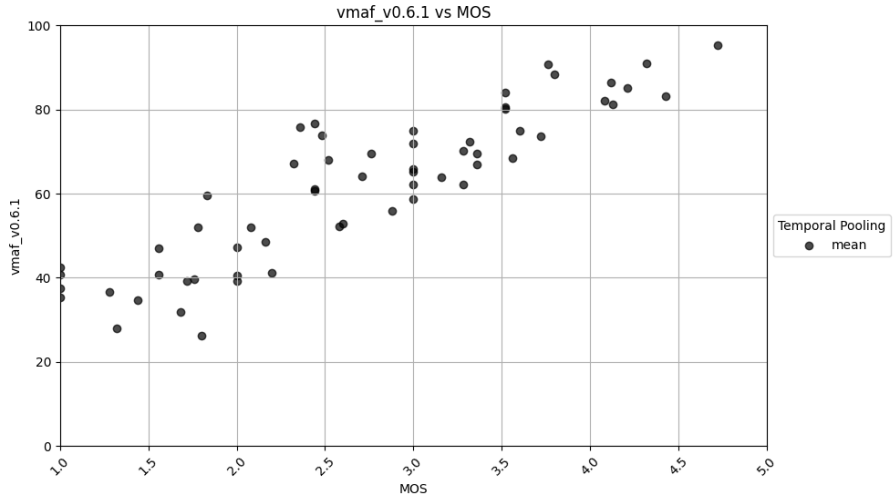

Figura 4.6: VMAF(modello vmaf_v0.6.1) vs MOS per il dataset GamingVideoSet

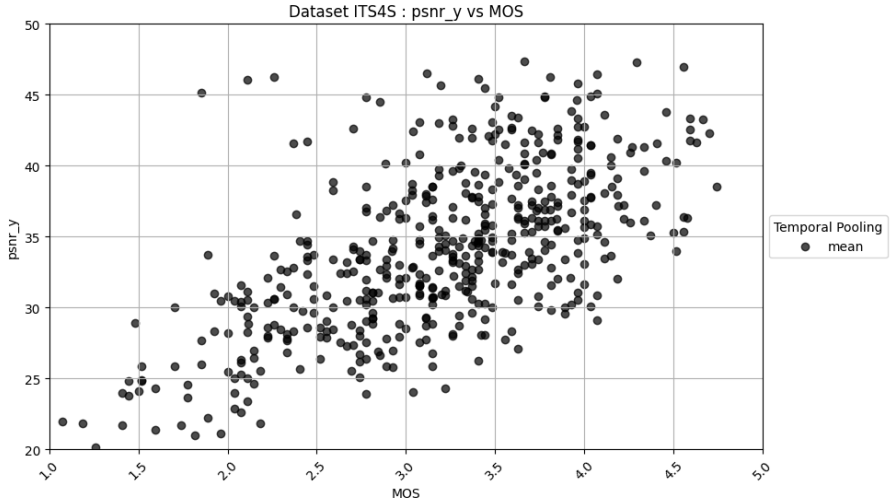

Figura 4.7: PSNR_y VS MOS per il dataset ITS4S

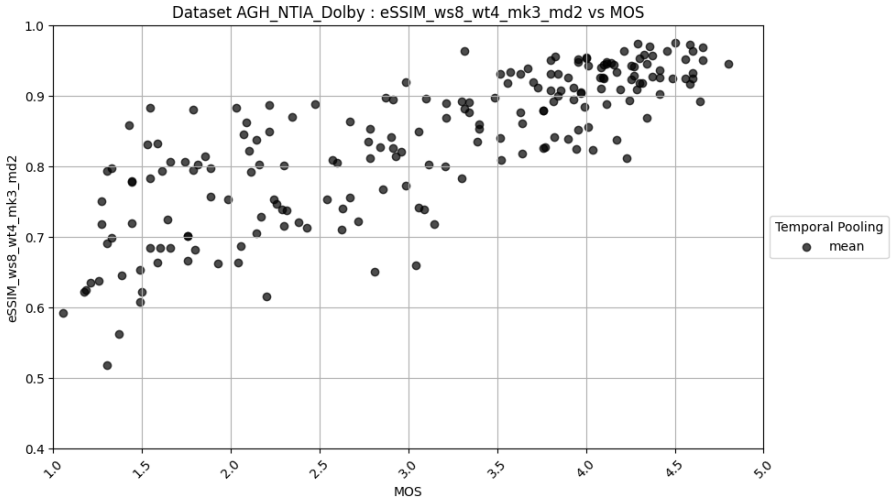

Figura 4.8: eSSIM(wsize:8,wstride:4, mink:3, mode:2) VS MOS per il dataset ITS4S

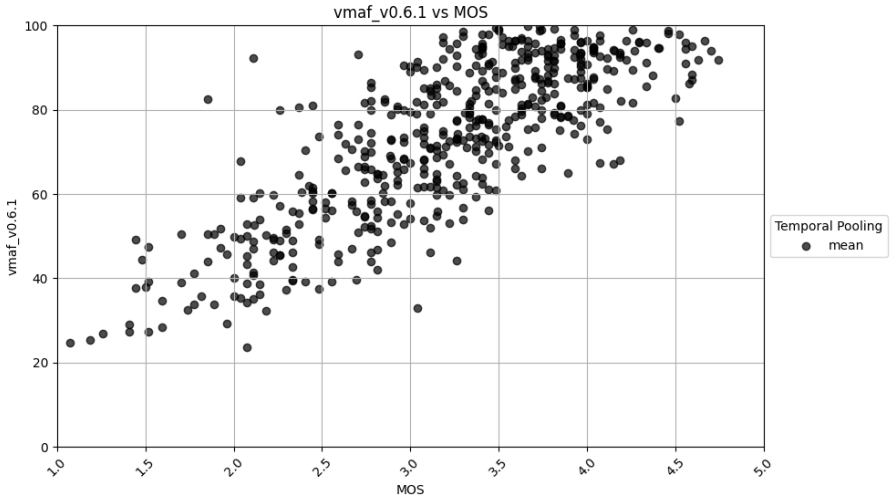

Figura 4.9: VMAF(modello vmaf_v0.6.1) vs MOS per il dataset ITS4S

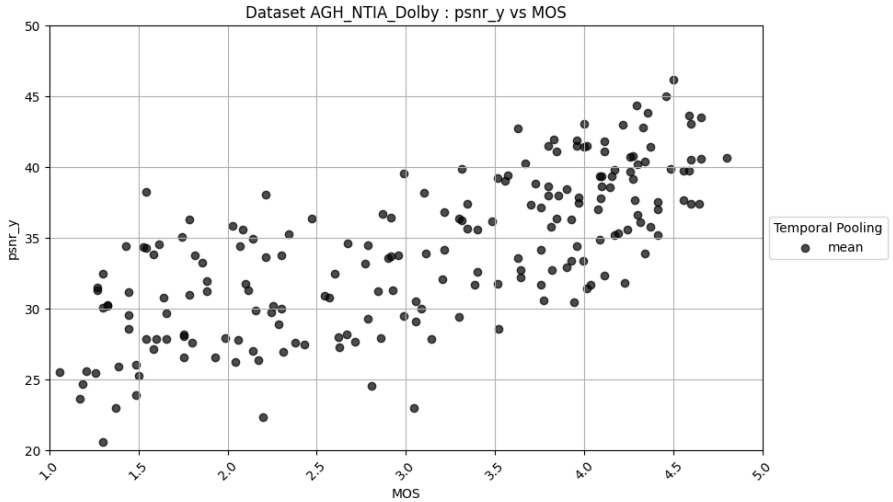

Figura 4.10: PSNR_y VS MOS per il dataset AGH_NTIA_Dolby

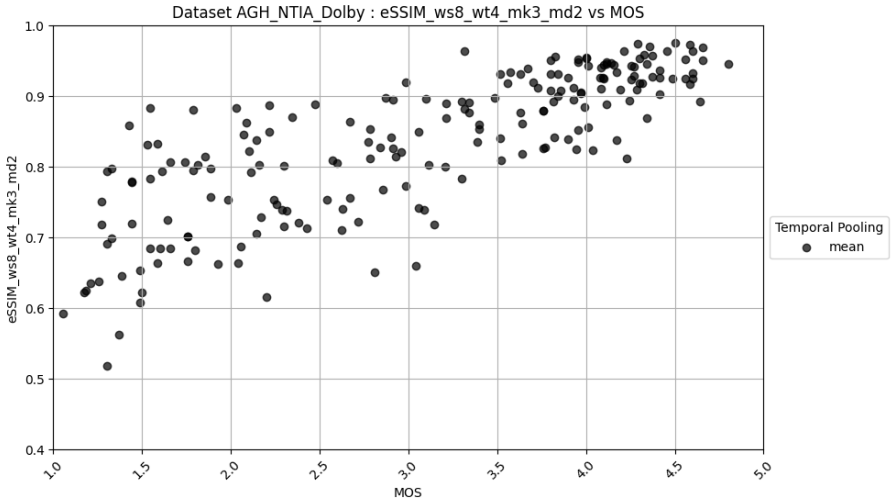

Figura 4.11: eSSIM(wsize:8,wstride:4, mink:3, mode:2) VS MOS per il dataset AGH_NTIA_Dolby

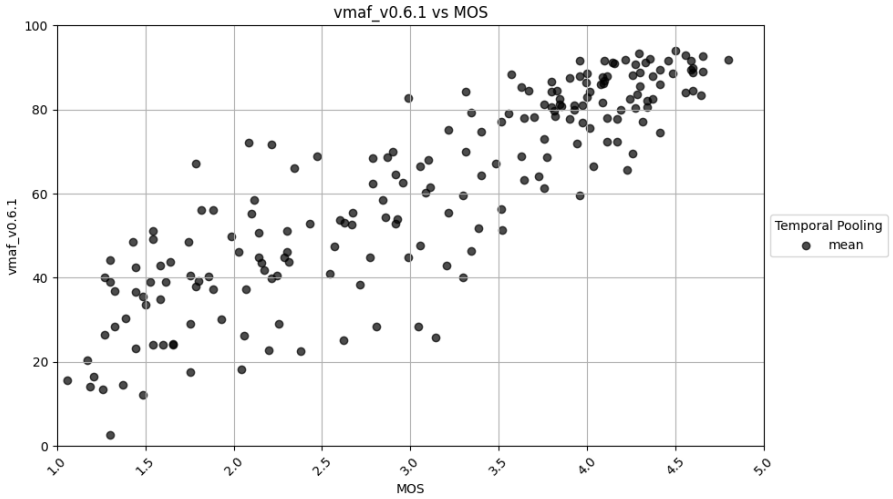

Figura 4.12: VMAF(modello vmaf_v0.6.1) vs MOS per il dataset AGH_NTIA_Dolby

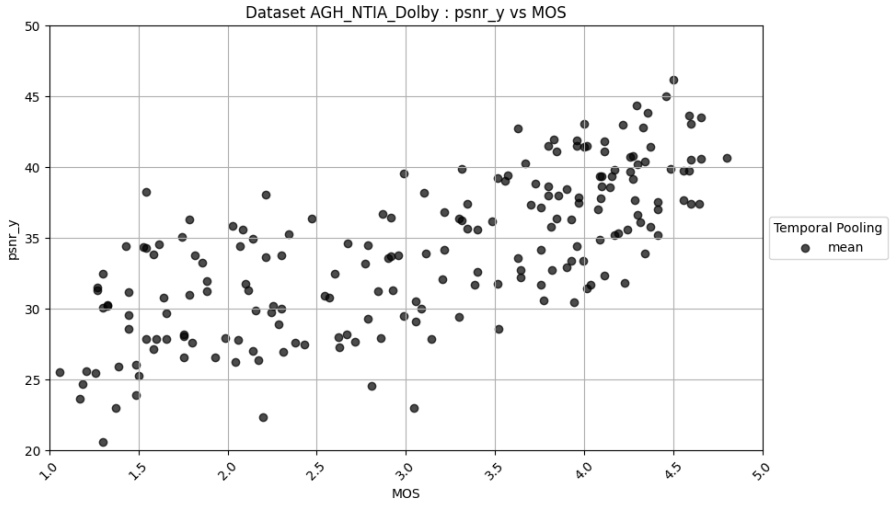

Figura 4.13: PSNR_y VS MOS per il dataset AVT-VQDB-UHD-1, test2

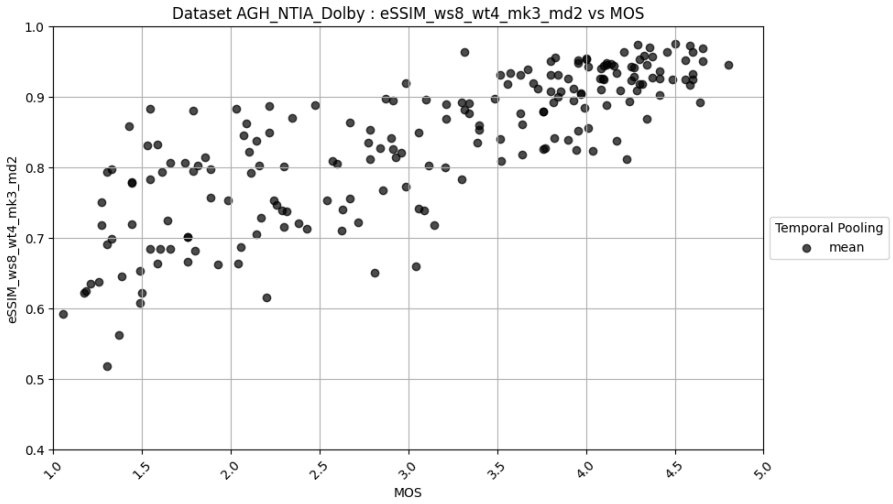

Figura 4.14: eSSIM(wsize:8,wstride:4, mink:3, mode:2) VS MOS per il dataset AVT-VQDB-UHD-1, test2

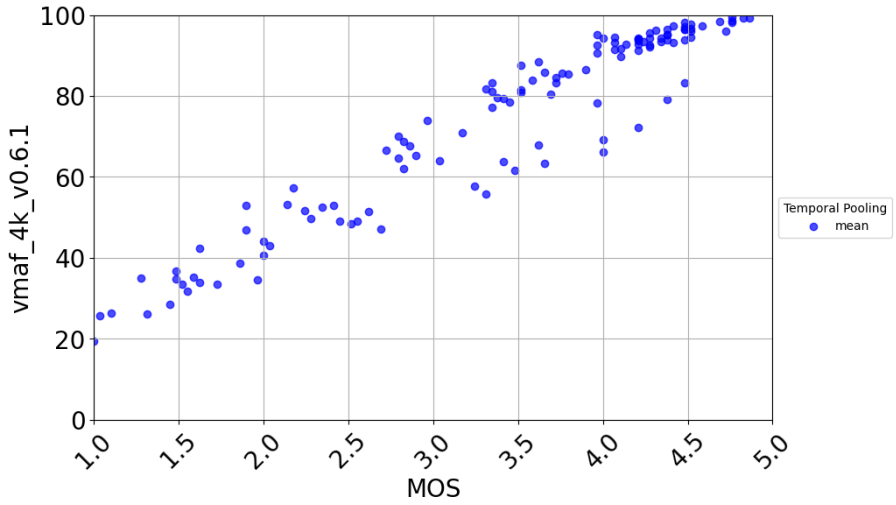

Figura 4.15: VMAF(modello vmaf_4k_v0.6.1) vs MOS per il dataset AVT-VQDB-UHD-1, test2

4.1.2 Incertezza del MOS

La variabilità del MOS è stata analizzata utilizzando scatter plot con barre di errore. VMAF consente di stimare la variabilità, fornendo così una misura della dispersione dei dati attorno al valore di VMAF. E' possibile estrarre i valori ci_p95_lo e ci_p95_hi per i due modelli b (VMAF_b_v0.6.3 e VMAF_float_VMAF_b_v0.6.3), estremi di un intervallo in cui il punto centrale è il VMAF. Il grafico mostra:

- in ascissa il valore di MOS
- in ordinata il valore di VMAF per corrispondente modello b
- sono rappresentate tutte le diverse PVS di un particolare dataset e per ognuna viene mostrato il valore della media

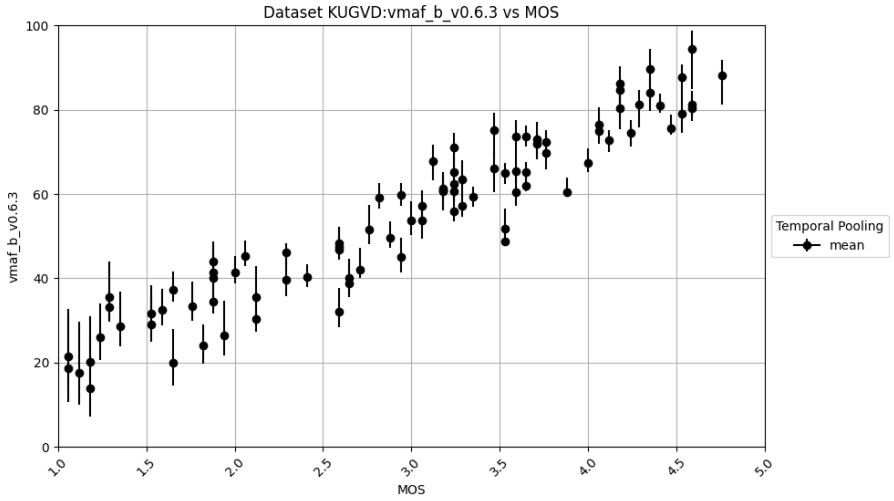

Figura 4.16: Error bar per il modello VMAF_b_v0.6.3 per il dataset KUGVD

Dall'analisi dei dati e dei grafici, è possibile notare un andamento significativo: l'incertezza, rappresentata dalla distanza tra i valori lo e hi (estremi delle barre di errore), tende ad aumentare al diminuire del Mean Opinion Score (MOS). In figura 4.24 sono visibili delle sovrapposizioni dovute a valori molto simili.

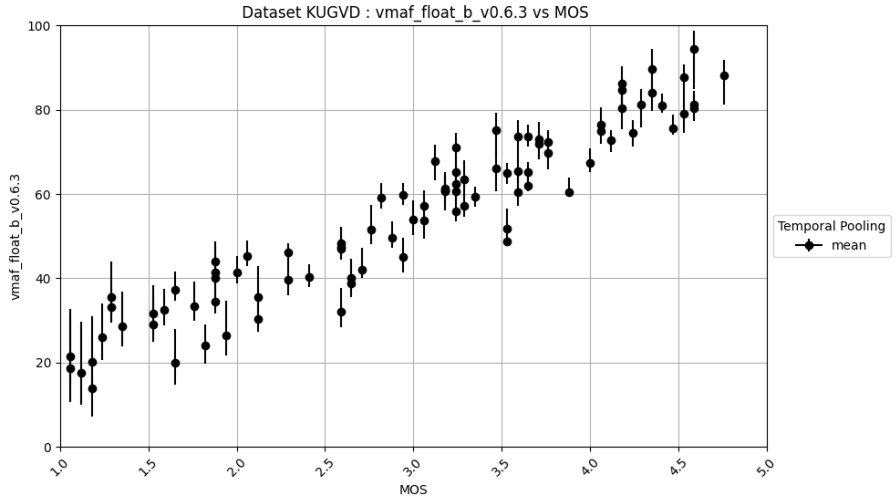

Figura 4.17: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset KUGVD

VMAF offre per questi due modelli anche il valore di deviazione standard, che rappresenta la dispersione dei dati attorno al valore medio. Aggiungendo e sottraendo questa deviazione standard al valore di VMAF, si ottiene un intervallo differente che fornisce un'ulteriore misura dell'incertezza della stima. I prossimi grafici presentano l'intervallo calcolato in questo modo, mostrando le barre di errore ottenute sulla base della deviazione standard.

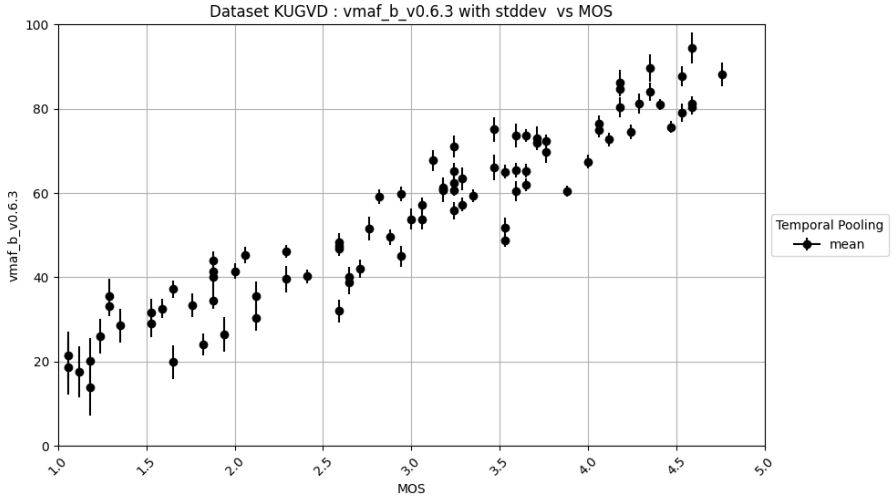

Figura 4.18: Error bar per il modello VMAF_b_v0.6.3 per il dataset KUGVD, dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

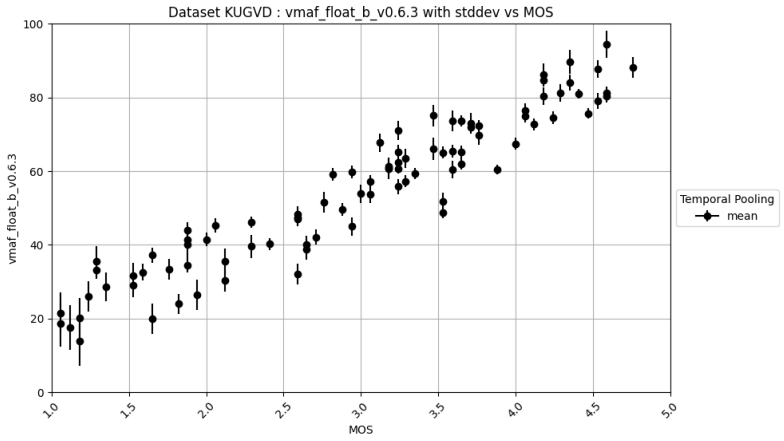

Figura 4.19: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset KUGVD , dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

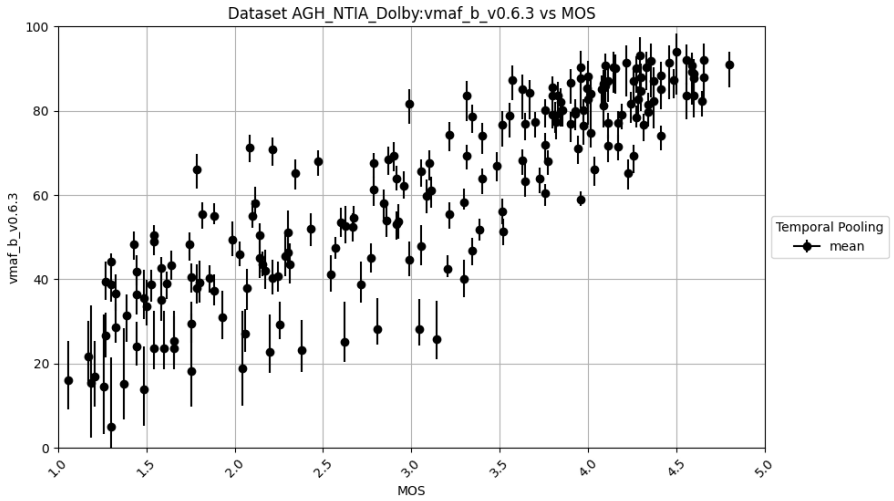

Figura 4.20: Error bar per il modello VMAF_b_v0.6.3 per il dataset AGH_NTIA_Dolby

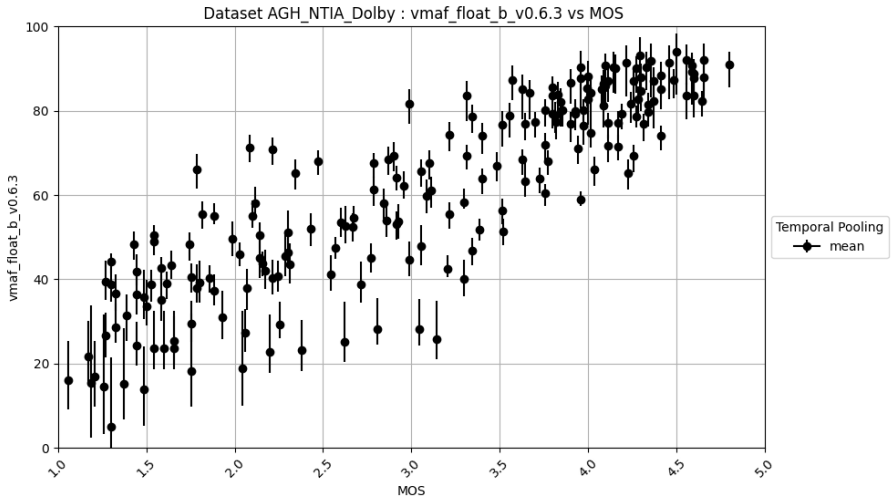

Figura 4.21: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset AGH_NTIA_Dolby

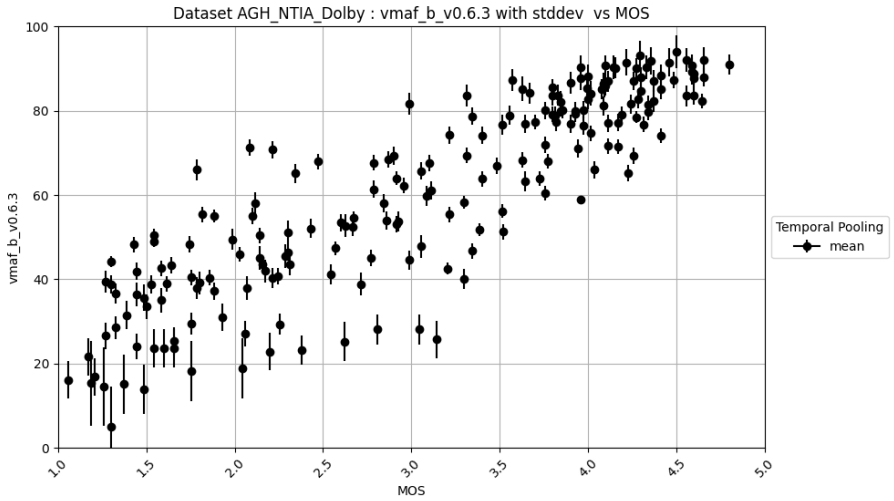

Figura 4.22: Error bar per il modello VMAF_b_v0.6.3 per il dataset AGH_NTIA_Dolby, dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

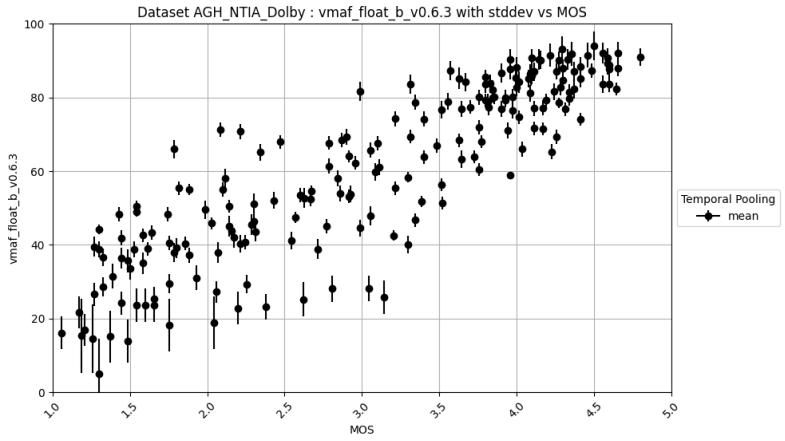

Figura 4.23: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset AGH_NTIA_Dolby, dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

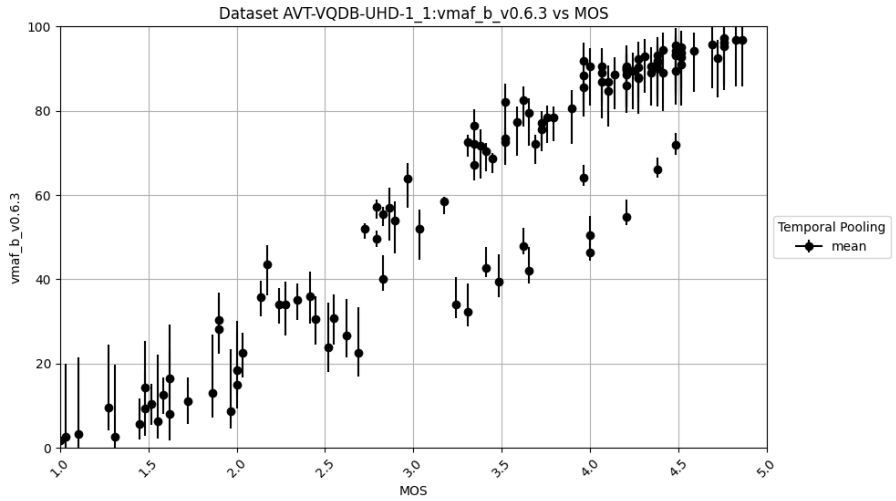

Figura 4.24: Error bar per il modello VMAF_b_v0.6.3 per il dataset AVT-VQDB-UHD-1, test1

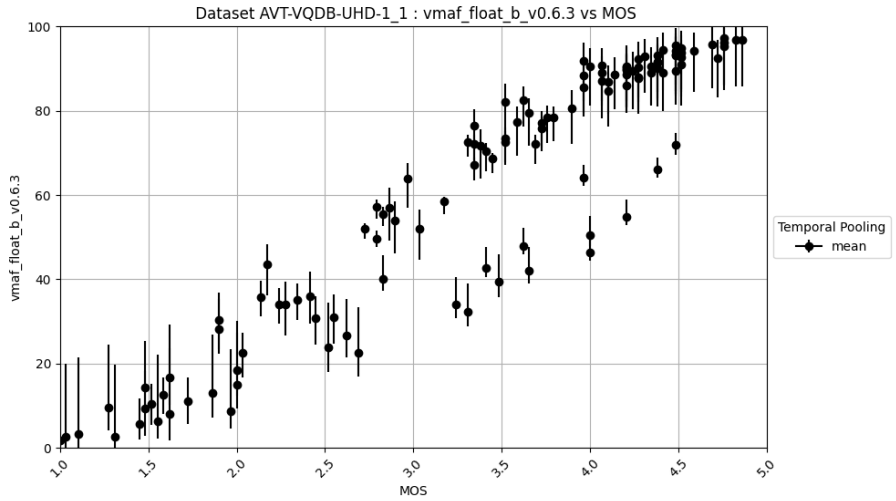

Figura 4.25: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset AVT-VQDB-UHD-1

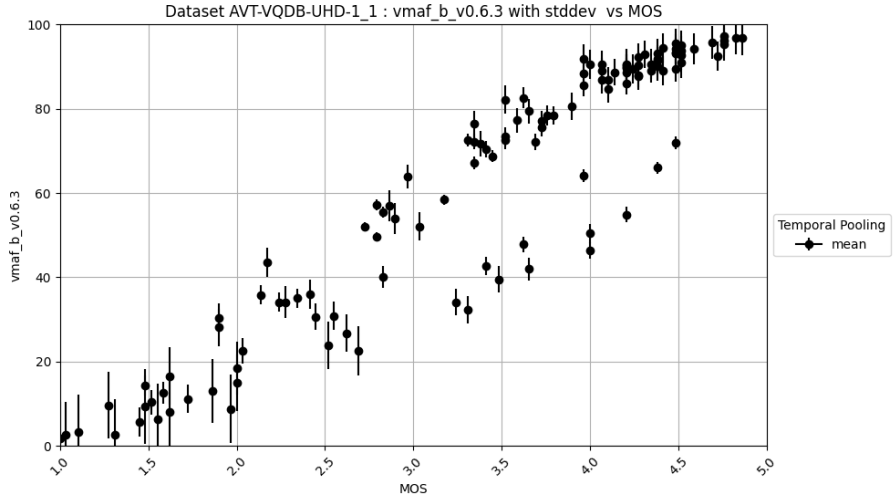

Figura 4.26: Error bar per il modello VMAF_b_v0.6.3 per il dataset AVT-VQDB-UHD-1, test1, dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

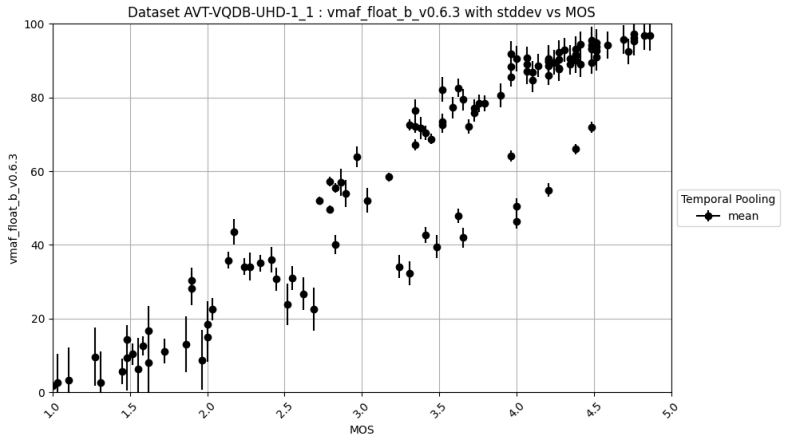

Figura 4.27: Error bar per il modello VMAF_float_b_v0.6.3 per il dataset AVT-VQDB-UHD-1, test1, dove le barre di errore rappresentano l'intervallo definito aggiungendo e sottraendo la deviazione standard al valore di VMAF

4.1.3 Differenze tra i Modelli di VMAF

Sono stati confrontati diversi modelli di VMAF per verificare eventuali differenze nelle valutazioni. Il grafico confronta il valore di valutazione soggettiva MOS in ascissa e in ordinata sono mostrati i valori di VMAF per due dei 9 possibili modelli. In figura 4.28 è mostrato il confronto dei risultati per il dataset KUGVD , con sequenze video in Full HD. I risultati si dimostrano essere molto simili. Risultano evidenti invece

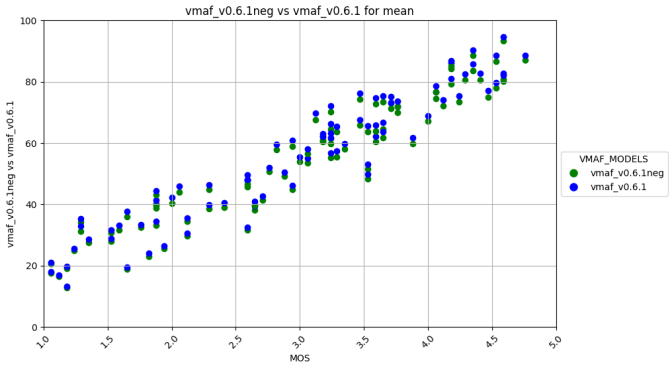

Figura 4.28: Confronto di valori della media rispetto al MOS per i due modelli VMAF_v0.6.1neg e VMAF_v0.6.1

le differenze per i modelli 4k utilizzati per sequenze in Full HD come mostrato in figura 4.29:

Dall'analisi condotta sui modelli VMAF, emerge un aspetto cruciale: i modelli 4k risultano efficaci solo per contenuti già in partenza con risoluzione 4k, come mostrato nel confronto tra le figure 4.30 e 4.31. I risultati riguardano il dataset AVT-VQDB-UHD-1 , analizzato per sequenze 4k. E' possibile notare come siano molto evidenti le differenze in termini di correlazione tra VMAF e MOS in base al modello scelto.

E' importante selezionare il modello VMAF appropriato in base alla risoluzione del contenuto video da valutare. Utilizzare un modello non adatto alla risoluzione target potrebbe portare a risultati non affidabili o fuorvianti nel confronto con il Mean Opinion Score (MOS). Pertanto, per garantire valutazioni accurate e coerenti, è essenziale prestare attenzione alla scelta del modello VMAF più adatto al contesto specifico. Solo in questo modo sarà possibile ottenere risultati affidabili e allineati con le aspettative qualitative espresse dal MOS.

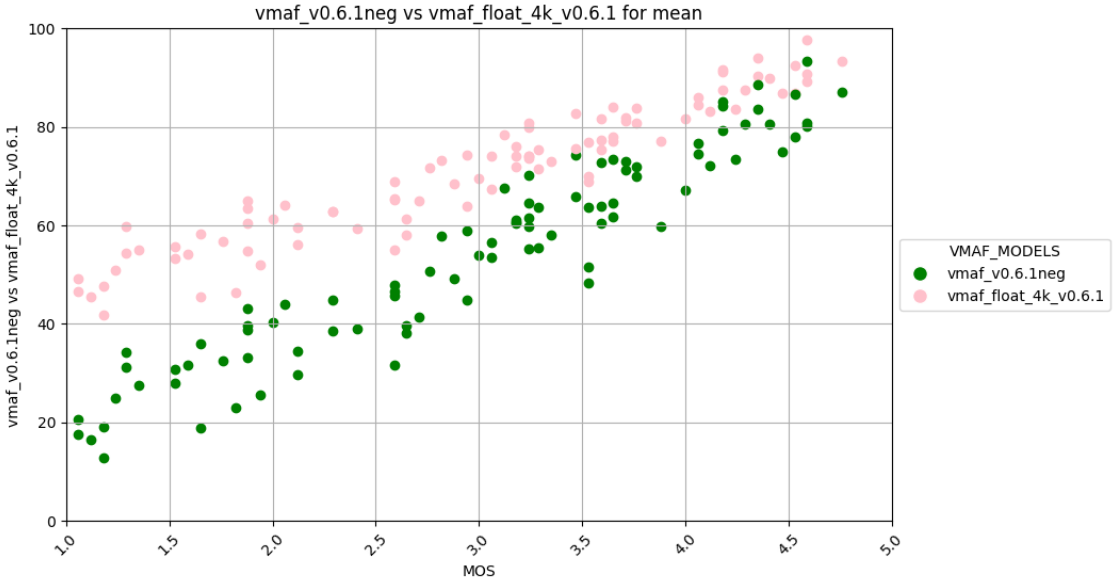

Figura 4.29: Confronto di valori della media rispetto al MOS per i due modelli VMAF_v0.6.1neg e VMAF_float4k_v0.6.1

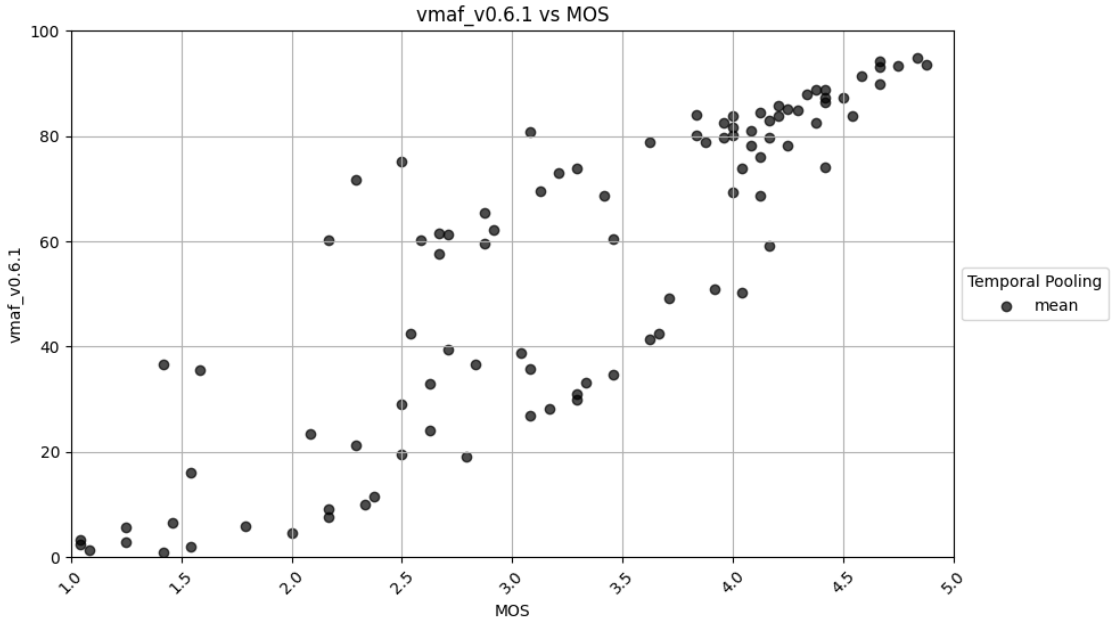

Figura 4.30: Valutazione VMAF per il modello v.0.6.1 rispetto a PVS in 4k

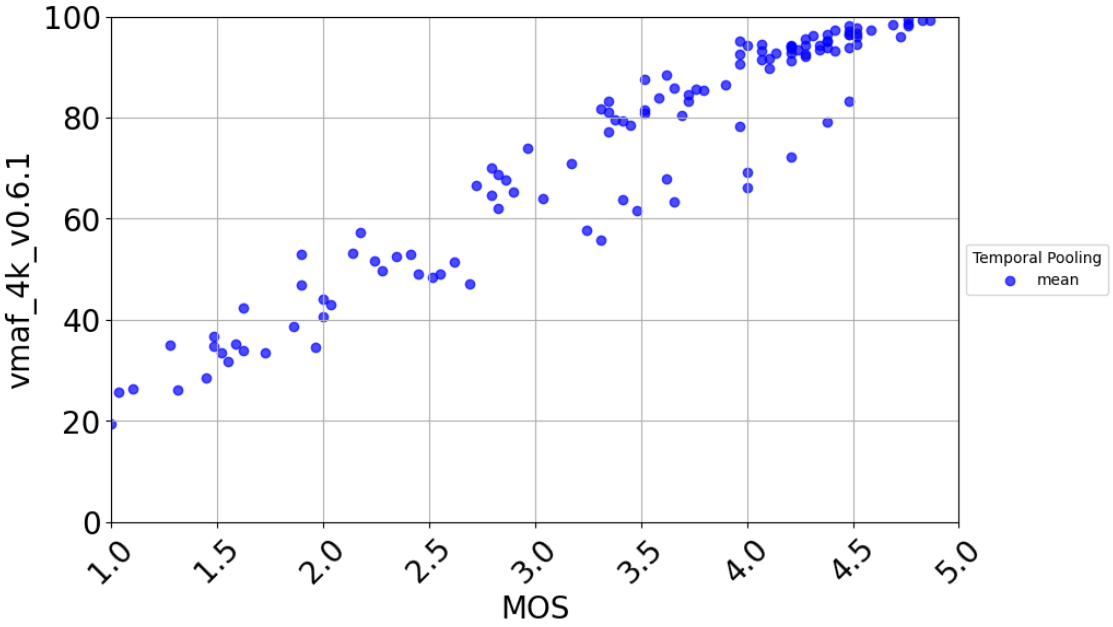

Figura 4.31: Valutazione VMAF per il modello 4k_v.0.6.1 rispetto a PVS in 4k

4.1.4 Metodi di Aggregazione Temporale

Sono stati analizzati diversi metodi di aggregazione temporale delle metriche:

- media:
$$M = \frac{1}{n} \sum_{i=1}^{n} x_i$$
- media armonica:
$$H = \frac{n}{\sum_{i=1}^{n} \frac{1}{x_i}}$$
- media geometrica:
$$G = \left(\prod_{i=1}^{n} x_i \right)^{\frac{1}{n}}$$
- percentili:

$$P_k = x_k \quad \text{dove} \quad P_k \text{ è il percentile corrispondente a } k\%.$$

- Il 5° percentile (P_5) è il valore sotto il quale si trovano il 5% dei dati.
$$P_5 = x_5$$

- Il 50° percentile (P_{50}), noto anche come mediana, è il valore che divide il 50% dei dati in due parti uguali.
$$P_{50} = x_{50}$$

- Il 95° percentile (P_{95}) è il valore sotto il quale si trovano il 95% dei dati.
$$P_{95} = x_{95}$$

- norme:
  - Norma L1 (Norma Manhattan o della somma assoluta):
$$\|x\|_1 = \sum_{i=1}^{n} |x_i|$$

– Norma L^2 (Norma Euclidea o quadratica):

$$\|x\|_2 = \left(\sum_{i=1}^{n} x_i^2\right)^{\frac{1}{2}}$$

– Norma L^3:

$$\|x\|_3 = \left(\sum_{i=1}^{n} |x_i|^3\right)^{\frac{1}{3}}$$

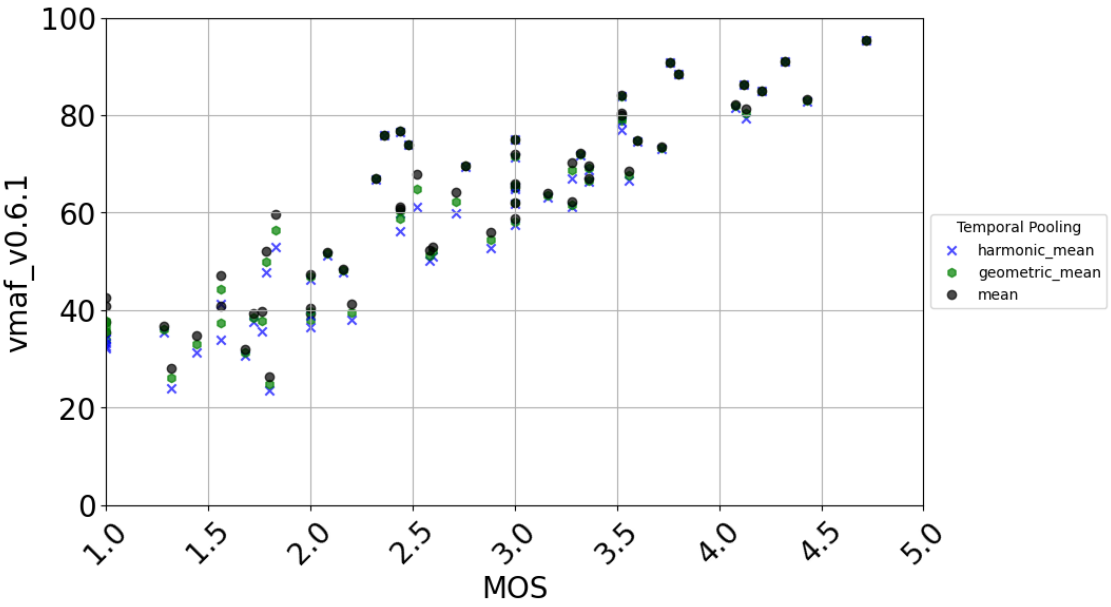

Figura 4.32: Scatter plot per il modello v.0.6.1 per diversi temporal pooling per il dataset GamingVideoSet

Per i vari metodi è stata confrontata l'efficacia in figura 4.32 e 4.33. Nonostante l'esistenza di numerose metodologie, i risultati indicano che le differenze tra i vari approcci sono minime. In particolare, la media si dimostra un metodo di aggregazione valido e affidabile, offrendo un buon compromesso tra semplicità e accuratezza dell'analisi.

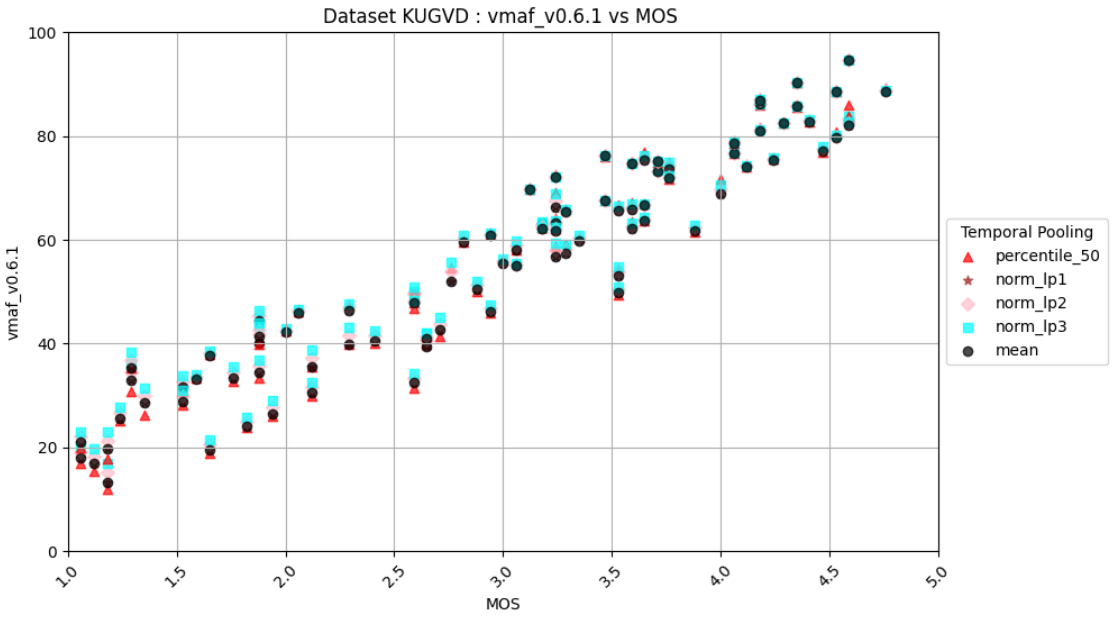

Figura 4.33: Scatter plot per il modello v.0.6.1 per altri temporal pooling per il dataset KUGVD

4.2 Codice

Il codice utilizzato per il framework che ha generato i risultati presentati in questo lavoro è disponibile pubblicamente su GitHub al seguente link: `https://github.com/robertogreco99/Tesi.git`.

Capitolo 5

Conclusioni

Obiettivo centrale del seguente lavoro è stato la realizzazione di un framework, semplice e flessibile da usare, per la realizzazione di esperimenti di valutazione oggettiva e il loro confronto con valori di valutazione soggettiva disponibili per i dataset analizzati. E' stata creata una pipeline automatizzata per l'esecuzione riproducibile delle codifiche e delle valutazioni di qualità.
A partire da un file di configurazione, si crea un'immagine Podman e su di essa sono stati fatti girare gli esperimenti su un insieme di datasets differenti e standardizzati per diverse metriche di valutazione oggettiva. Sono stati raccolti i risultati in un formato standard e prodotti degli script per analizzare i risultati.
Sono stati utilizzati differenti metodi di aggregazione temporale sui risultati prodotti e prodotti grafici per mostrare diversi confronti tra metriche oggettive e soggettive, quali correlazione con il MOS, variabilità del MOS e differenza nei risultati tra diversi modelli di VMAF. VMAF è risultata essere la metrica migliore per effettuare valutazioni oggettive ma anche eSSIM ha prodotto dei buoni risultati. Al diminuire del MOS si è trovata un aumento dell'incertezza per quanto riguarda i valori ottenuti da VMAF per i due modelli b, per i quali sono stati estratti intervallo e valore medio. Si è mostrato come l'utilizzo di un modello o di un altro influisca sull'esito della valutazione. La media è risultata essere un buon metodo di aggregazione temporale.

Ringraziamenti

Vorrei ringraziare sinceramente la mia famiglia per il costante supporto durante tutto il mio percorso di studi. Ogni passo che ho fatto è stato accompagnato dalla vostra presenza, e vi sarò per sempre grato.
Un sentito ringraziamento va ai miei amici di sempre: Vincenzo, Daniele, Rosario, Davide, Paolo e Mattia che mi sono stati vicini negli anni, sostenendomi e permettendomi di arrivare fino a questo traguardo.
Un affettuoso saluto anche ai miei amici di Torino, che mi hanno accompagnato in questo lungo e significativo viaggio, rendendolo speciale grazie alla loro compagnia: Gaia, Samuele, Miriana, Elisa, Martino, Foros, Danilo, Claudia, Angelo, Davide, Chiara, Lucio e Giuseppe. Un saluto va inoltre alla mia amica Sarah, per aver condiviso con me una parte del percorso e averlo reso più facile insieme.
Infine, desidero esprimere la mia sincera gratitudine al mio relatore, Enrico Masala, per il prezioso supporto e la guida costante durante lo sviluppo di questo lavoro.

Bibliografia

[1] Nabajeet Barman, Saman Zadtootaghaj, Steven Schmidt, Maria G. Martini, and Sebastian Möller. Gamingvideoset: A dataset for gaming video streaming applications. In *2018 16th Annual Workshop on Network and Systems Support for Games (NetGames)*, pages 1–6, 2018.

[2] Sandvine. Global internet phenomena report 2024. `https://www.sandvine.com/global-internet-phenomena-report-2024`, 2024.

[3] International Telecommunication Union. Recommendation ITU-T P.910: Subjective video quality assessment methods for multimedia applications. Technical Report P.910, International Telecommunication Union, 2008. Section 6.1 'Absolute Category Rating (ACR)', page 6.

[4] International Telecommunication Union. Recommendation ITU-R BT.500-13: Methodology for the subjective assessment of the quality of television pictures. Technical Report BT.500-13, International Telecommunication Union, 2012. Annex 2, Section 2.1, Formula (1), page 34.

[5] ITU Recommendation BT. 500-14, methodologies for the subjective assessment of the quality of television images. *Geneva: International Telecommunication Union*, 2019.

[6] International Telecommunication Union. Recommendation ITU-T P.910: Subjective video quality assessment methods for multimedia applications. Technical Report P.910, International Telecommunication Union, 2008.

[7] Zhou Wang, A.C. Bovik, H.R. Sheikh, and E.P. Simoncelli. Image quality assessment: from error visibility to structural similarity.

IEEE Transactions on Image Processing, 13(4):600–612, 2004.

[8] Guan-Hao Chen, Chun-Ling Yang, Lai-Man Po, and Sheng-Li Xie. Edge-based structural similarity for image quality assessment. *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2:II–II, 2006.

[9] Zhi Li, Anne Aaron, Ioannis Katsavounidis, Anush Moorthy, and Megha Manohara. Toward a practical perceptual video quality metric. *Netflix Technology Blog*, 2016.

[10] Anne Aaron, Zhi Li, Megha Manohara, Jan De Cock, and David Ronca. Per-title encode optimization. *Netflix Tech Blog*, 2015.

[11] IGVQM Group. Implementer's guide to video quality metrics (igvqm). `https://docs.google.com/document/d/1w3rgGxDHFehwdwtVXENtngAPsMhGnPlAsiU2v0Tu6nU/edit?usp=sharing`, 2023.

[12] VQEG. Igvqm. `https://vqegjeg.github.io/jeg-hybrid/igvqm`.

[13] VQEG. Jeg hybrid. `https://vqegjeg.github.io/jeg-hybrid/`.

[14] VQEG. Video quality experts group (vqeg). `http://vqeg.org`.

[15] Rakesh Rao Ramachandra Rao, Steve Göring, Werner Robitza, Bernhard Feiten, and Alexander Raake. Avt-vqdb-uhd-1: A large scale video quality database for uhd-1. In *2019 IEEE ISM*, pages 1–8, Dec 2019.

[16] Nabajeet Barman, Emmanuel Jammeh, Seyed Ali Ghorashi, and Maria G Martini. No-reference video quality estimation based on machine learning for passive gaming video streaming applications. *IEEE Access*, 7:74511–74527, 2019.

[17] Margaret H Pinson. Its4s: A video quality dataset with four-second unrepeated scenes. Technical report, Institute for Telecommunication Sciences, 2018.

[18] Lucjan Janowski, Ludovic Malfait, and Margaret H. Pinson. Evaluating experiment design with unrepeated scenes for video quality subjective assessment. *Quality and User Experience*, 4(2), June 2019.

[19] Netflix. Vmaf: Video multimethod assessment fusion. `https://github.com/Netflix/vmaf`, 2025.

[20] Facebook Research. Essim: Efficient super-resolution image models. `https://github.com/facebookresearch/essim`, 2021.

[21] FFmpeg Developers. FFmpeg – A complete, cross-platform solution to record, convert and stream audio and video. `https://www.ffmpeg.org/`, 2025.

[22] ImageMagick. Imagemagick examples resampling filters. `"https://usage.imagemagick.org/filter/#lanczos"`.

[23] Frank Stenger. *Numerical methods based on sinc and analytic functions*, volume 20. Springer Science & Business Media, 2012.

[24] Ken Turkowski. Filters for common resampling tasks. In *Graphics gems*, pages 147–165. 1990.

[25] Red Hat contributors. What is podman? `https://www.redhat.com/en/topics/containers/what-is-podman`, 2025.

[26] Docker. Python 3.12-bookworm. `https://hub.docker.com/layers/library/python/3.12-bookworm/images/`sha256-3e7c0f87d7c085a6ec7511b2fa69194b21ea54bc3a110c35e37ba20cbd0aef7e.