Mesopotamian journal of Computer Science
Vol. **(2025)**, 2025, **pp**. 258–287
DOI: https://doi.org/10.58496/MJCSC/2025/017; ISSN: 2958-6631
https://mesopotamian.press/journals/index.php/cs

Research Article

Automated Video Colorization Techniques for Enhanced Visual Realism and Computational Efficiency

Zahoor M. Aydam [1*], Nidhal K. El Abbadi [2]

[1]*Computer Science Department, Faculty of Computer Science and Mathematics, University of Kufa, Najaf, Iraq*
[2]*Al-Mustaqbal Center for AI Research and Applications, Al-Mustaqbal University, Babylon, Iraq*

ARTICLEINFO

Article History

Received 18 Jun 2025
Revised 10 Aug 2025
Accepted 27 Aug 2025
Published 2 Sep 2025

Keywords

Resnet50-Gem

Image processing of Gray-Level Videos

Image Features ELU

Reference Image

Trilinear Upsampling Encoder And Decoder

Abstract

Automatic video colorization remains a challenging computer vision task, particularly when ensuring semantic accuracy and temporal coherence across dynamic, multi-scene content. Existing methods often rely on a single fixed reference image, which fails to adapt to abrupt scene changes or variations in lighting and texture. This study presents a **hybrid deep learning framework** that dynamically selects multiple reference images per scene using adaptive thresholds derived from the Structural Similarity Index Measure (SSIM) and deep features extracted via a ResNet50 backbone with Generalized Mean Pooling (GeM). The framework integrates three specialized modules pre-processing, reference image processing, and attention-based colorization—operating in the Lab color space before conversion to RGB. Experimental evaluations on the YouTube-8M dataset demonstrate a **PSNR of 37.89, SSIM of 0.998, and inference speed of 2.6 FPS** with a compact **81 MB model (3.2M parameters)**. Compared to state-of-the-art methods, the proposed approach achieves superior color fidelity and temporal stability while maintaining efficiency, making it suitable for deployment in resource-constrained environments such as embedded vision and IoT systems.

1. INTRODUCTION

Color plays a fundamental role in human visual perception, providing essential cues for object recognition, spatial understanding, and emotional interpretation. In video-based applications, accurate color reproduction enhances depth perception and contextual awareness, both of which are critical in domains such as film restoration, augmented reality, and intelligent surveillance [1–3]. While image colorization has seen significant advances in recent years, extending these methods to video introduces the additional challenge of maintaining **temporal consistency** to avoid flickering and chromatic instability across frames [4,5].

Traditional automated video colorization frameworks often rely on **a single static reference image** for the entire sequence, which limits adaptability to **multi-scene videos** where lighting, textures, and semantic content vary significantly [6,7]. Furthermore, many lightweight deep learning approaches, such as **MobileNetV3** and **YOLOv8n-based detectors** [8,9], have been optimized for classification or detection tasks rather than fine-grained color propagation, leading to reduced performance in scenarios involving **abrupt scene transitions or occlusions**. Transformer-based architectures have shown promise in modeling long-range dependencies [10], yet they often come with increased computational cost, making them unsuitable for **edge computing and IoT devices** where real-time performance is critical.

Recent literature (2023–2024) [6–10] highlights this trade-off between accuracy and efficiency, showing that while lightweight models reduce parameters by up to **60%**, they may suffer accuracy drops of **3–5% SSIM** in complex multi-scene datasets. However, there remains a gap in models that achieve both **high color fidelity (SSIM ≥ 0.995)** and **low inference latency (<3 sec/frame)** while keeping a small memory footprint (<100 MB).

To address these limitations, this study introduces a **scene-aware, end-to-end trainable video colorization framework** with **adaptive multi-reference selection**. The system employs SSIM-based scene segmentation with dynamic thresholds, followed by **ResNet50-GeM** feature matching to identify semantically relevant references for each scene. The selected references are processed through an attention-enhanced colorization network with multi-layer feature fusion, enabling

Corresponding author. Email: zahoorm.alfraih@student.uokufa.edu.iq

spatially accurate and temporally stable color restoration. Quantitative results show that the proposed model achieves **37.89 PSNR, 0.998 SSIM, and 2.6 FPS inference speed** with a compact **81 MB size**, outperforming existing state-of-the-art methods in both accuracy and efficiency.

2. RELATED WORKS

Recent advancements in reference-based video colorization have sought to overcome longstanding challenges in ensuring high-fidelity color accuracy and maintaining temporal consistency across frames. These methods typically aim to increase robustness against scene dynamics, reduce color bleeding, and ensure smoother transitions in motion-intensive video sequences.

Yang et al. introduced BiSTNet, a bidirectional two-stage colorization framework designed to ensure superior chromatic fidelity and temporal coherence. Their approach uses two reference images per video and operates in both forward and backwards directions. During the forward pass, frames are colorized sequentially using the first reference image. Bidirectional temporal feature fusion is then applied to propagate color features smoothly across time. To address color bleeding at object boundaries and enhance spatial clarity, a mixed expert block extracts high-level semantic priors. In the backwards pass, the same frames are reprocessed via a second reference image, effectively refining earlier results and reinforcing the model's resilience. This dual-pass architecture allows the fusion of both passes' outputs, producing vivid, artifact-minimized frames with consistent color representations [6].

Chen et al. proposed a reference-guided video colorization model that explicitly leverages long-range spatial and temporal information to increase both accuracy and continuity. Their framework introduces a dual-head nonlocal module and a CNN-transformer hybrid block to capture both local textures and global contextual relationships. The transformer enhances the model's ability to understand semantic structures, whereas the CNN branch preserves detailed spatial features. Additionally, a linkage subnet is introduced to enable temporal coherence by transmitting motion-aware features between adjacent frame segments. Color propagation is performed progressively—each frame is colorized using a reference image initially, after which the previous colorized frame is iteratively used as the reference for the next frame. This method guarantees smooth transitions and maintains color integrity throughout the video [7].

Huang et al. presented long-term video colorization with diffusion (LVCD), a novel framework particularly tailored for animated content. Unlike traditional frame-by-frame approaches, LVCD employs a single reference image to initiate colour transfer, which is then propagated through a diffusion-based mechanism across the sequence. This ensures a unified color palette and preserves temporal consistency. A reference attention mechanism further refines the colorization process by enabling stable transfer of color across fast-moving objects and large-scale motions. In dynamic animation scenarios, the system employs a Sketch-Guided ControlNet, which reinforces the structural integrity of the source sketches while enhancing the overall visual quality [8].

In a separate study, Yang et al. developed a reference-based colorization strategy that introduces a novel chrominance anchor point in the AB color space and an improved method for filling missing color regions. This technique is especially useful when the reference image lacks sufficient or reliable chromatic information. Color cues derived from the reference are initially segmented and then disseminated bidirectional across the video via a dense optical flow network. This approach facilitates robust propagation of color data across frames, preserving temporal coherence even in rapidly changing or motion-heavy scenes. The bidirectional nature of this method not only enhances reliability but also ensures color consistency throughout the sequence [9].

To synthesize the comparative insights from these studies, Table I provides a structured summary of their respective contributions, mechanisms, and target strengths.

TABLE I. COMPARATIVE SUMMARY OF RELATED VIDEO COLORIZATION METHODS

Study	Method	Reference Usage	Core Techniques	Strengths
Yang et al. [6]	BiSTNet	Two reference images per video	Bidirectional Temporal Fusion, Mixed Expert Block	Dual-stage refinement, temporal consistency
Chen et al. [7]	CNN-Transformer hybrid model	Single initial reference + iterative	Dual-head nonlocal module, Linkage Subnet, CNN-Transformer block	Long-range interaction, progressive propagation
Huang et al. [8]	LVCD	Single reference image	Diffusion-based propagation, Reference Attention, Sketch-Guided ControlNet	High temporal coherence, designed for animations
Yang et al. [9]	AB Chrominance & Optical Flow	Single reference image	AB chrominance anchor, bidirectional optical flow, color filling scheme	Handles fast motion, color reliability

3. METHODOLOGY

3.1 Proposed Method

This study presents an automated, end-to-end framework for colorizing grayscale videos with multiple scenes by eliminating the need for manual reference image selection. The proposed pipeline comprises four interconnected modules: **training the proposed network, feature extraction, scene analysis, and video colorization**. As illustrated in Figure 1, these modules are sequentially integrated to ensure accurate reference image identification and temporally consistent colorization output.

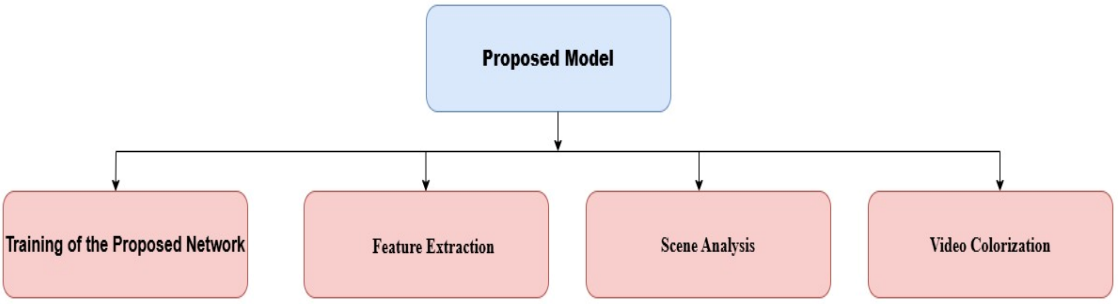


Fig. 1. Architecture of the proposed end-to-end framework for scene-aware automatic video colouration

3.1.1 Dataset Preparation and Pre-processing

To train the proposed model effectively, a total of 1,569 videos were selected from the publicly available YouTube-8 M dataset, which is renowned for its extensive variety in visual content spanning numerous subjects and scenes. This diversity ensures that the training process is exposed to a wide range of visual patterns and temporal dynamics, making it an ideal resource for video-based deep learning applications. The dataset was partitioned into three distinct subsets to ensure methodological rigor: the training set included 1,219 videos comprising 7,993,132 frames; the validation set contained 50 videos totalling 321,306 frames; and the remaining 300 videos (1,928,572 frames) were allocated to the test set. This stratified division was designed to maintain consistency in content distribution across subsets, thereby reducing sampling bias and enhancing the generalizability of the trained model. Additionally, the abundance of frames serves as a substantial reference pool for subsequent stages that depend on reference image retrieval.

To reduce redundancy and improve computational efficiency, frames were extracted at a fixed sampling interval—specifically, one frame was selected for every ten consecutive frames in each video. This temporal downsampling minimizes overlap between adjacent frames and promotes higher scene variability, which is essential for learning context-aware features. All extracted frames were uniformly resized to 224×224 pixels to meet the architectural input constraints of the ResNet-50 backbone used in the network. Prior to model ingestion, the pixel intensities of the frames were normalized to the range [0, 1], a standard preprocessing step that accelerates convergence and improves numerical stability during training. Moreover, a mean filter was applied to each frame to suppress high-frequency noise, thereby enhancing the quality of visual features and ensuring more robust feature extraction throughout the learning process.

3.1.2 Training the proposed network

Feature extraction is a crucial step in automatic video scene colorization. The primary objective is to generate an optimal feature vector that facilitates the selection of the most suitable reference image for color transfer. The training process is systematically divided into seven well-defined stages to ensure effective feature learning and scene understanding, as illustrated in Figure 2.

Stage 1: Pre-processing

The preprocessing stage begins by generating an image dataset from the YouTube8M video dataset. This is achieved by sampling one frame every ten frames from each video to reduce redundancy and focus on key frames. Each extracted frame

is resized to 224×224 pixels, which aligns with the input requirements of the ResNet50 architecture. Pixel values are normalized to the range [0, 1] to stabilize training and facilitate convergence. Additionally, a mean filter is applied to each frame to perform denoising, thereby improving the clarity of the images and minimizing noise interference during feature extraction.

Stage 2: initial convolution and max pooling

In this stage, the preprocessed frames are passed through an initial 7×7 convolutional layer with a stride of 2. This convolutional operation is followed by a ReLU activation function to introduce nonlinearity. Max pooling is then applied to reduce the spatial dimensions of the feature maps, thereby highlighting the most salient features while reducing computational complexity. This process initiates the extraction of features from the input images, providing the foundation for subsequent layers.

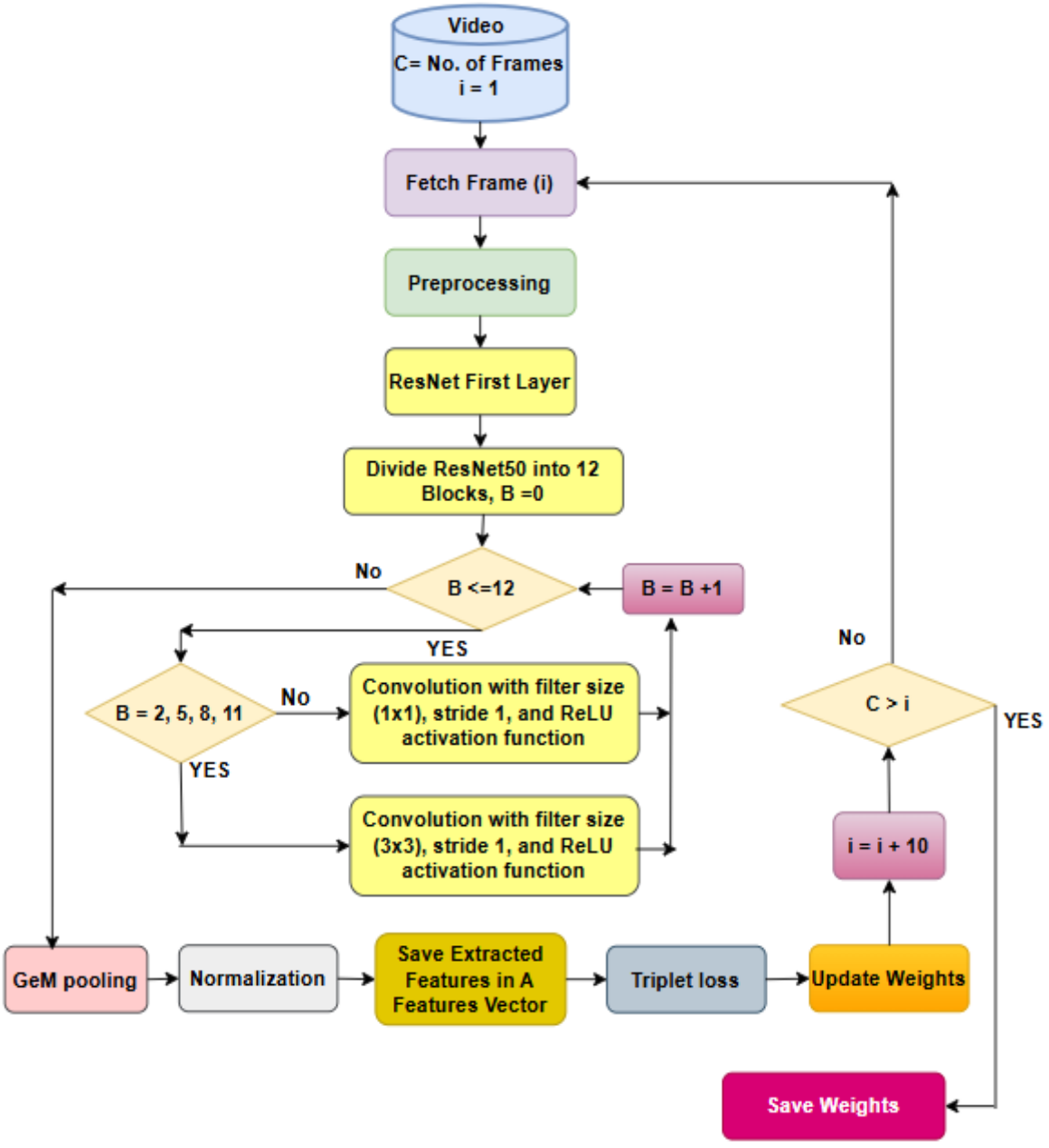


Fig. 2. Training ResNet50Gem

Stage 3: Block wise ResNet50 feature extraction

The modified ResNet50 architecture is employed for hierarchical feature extraction. ResNet50 consists of twelve residual blocks, as shown in Table II, each of which is responsible for capturing features at different levels of abstraction. Skip connections within the network alleviate the vanishing gradient problem, facilitating efficient backpropagation and ensuring that the network can learn deep, hierarchical features effectively. The final fully connected layer, which is traditionally used for classification, is removed to repurpose ResNet50 for feature extraction, making it suitable for tasks such as scene understanding and reference image selection in the video colorization framework.

TABLE II. ARCHITECTURE OF THE RESNET50 MODEL

Blocks Number	**Block 1**	**Block 2**	**Block 3**	**Block 4**	**Block 5**	**Block 6**	**Block 7**	**Block 8**	**Block 9**	**Block 10**	**Block 11**	**Block 12**
Layers Included	**2, 3, 4**	**5, 6, 7**	**8, 9, 10**	**11, 12, 13, 14**	**15, 16, 17, 18**	**19, 20, 21, 22**	**23, 24, 25, 26, 27, 28**	**29, 30, 31, 32, 33, 34**	**35, 36, 37, 38, 39, 40**	**41, 42, 43**	**44, 45, 48**	**47, 48, 49**

Below, we outline the roles of these blocks within the ResNet-50 framework:

- Block 1: In this block, image dimensions are changed while preserving essential features by applying a 1×1 convolution, ReLU, and a stride of 1.

- Blocks 2, 5, 8, and 11: In these blocks, the more detailed features are extracted from the image by applying a 3×3 convolution, ReLU, and a stride of 1.

- Other blocks: These blocks are utilized to fine-tune the extracted features while maintaining spatial dimensions by applying a 1×1 convolution, ReLU, and a stride of 1.

Stage 4: Generalized mean pooling (Gem)

After feature extraction by the last ResNet block, the output is passed through a generalized mean pooling (GeM) layer. Compared with traditional pooling techniques, GeM pooling is a more flexible and robust method of aggregating spatial features. The pooling parameter p in GeM is treated as a learnable parameter, allowing the network to dynamically adjust its pooling behavior during training. This flexibility helps in better aggregating features across varying video content and improves the quality of the generated feature representations by reducing the influence of outlier features.

Stage 5: L2 normalization

L2 normalization is used to normalize the extracted features. This makes the features appropriate for subsequent processes such as image retrieval by ensuring that the norm is equal to one for all feature vectors.

Mathematically, the **L2 normalization** of a feature vector **x** is given by:

$$x_{normalized} = \frac{x}{||x||_2} = \frac{x}{\sqrt{\sum_{i=1}^{N} x_i^2}} \quad \dots \quad (1)$$

where:
x is the extracted feature vector.
$||x||_2$: Euclidean (L2) norm.
N: the number of elements in the feature vector.

In this stage, L2 normalization is crucial for enhancing the stability and performance of the subsequent tasks, such as similarity comparison, by ensuring that the extracted feature vectors have a unit norm. The suggested method improves the quality and reliability of the features for subsequent applications by combining L2 normalization with ResNet50GeM's powerful feature extraction capabilities.

Stage 6: Triplet Loss Function

The normalized feature vectors are then input into a triplet loss function, which plays a vital role in learning discriminative features. The triplet loss function encourages the network to minimize the distance between similar feature vectors (anchor-positive pairs) while maximizing the distance between dissimilar feature vectors (anchor-negative pairs). This approach helps the model learn compact and distinct feature embeddings, ensuring that similar frames are closer together in feature space, whereas dissimilar frames are effectively separated. Triplet loss is a key component in enabling the model to understand subtle differences in scene content.

Stage 7: Updating Network Weights

During the training process, the network weights are updated via backpropagation on the basis of the computed triplet loss. This iterative process refines the feature embedding, allowing the model to improve its ability to distinguish between different video scenes. As the training progresses, the embedding space evolves to enhance intraclass similarity (similar scenes are closer) and interclass distinction (dissimilar scenes are farther apart). This continuous refinement ensures that the model learns robust and meaningful feature representations that are crucial for tasks such as scene segmentation and reference image selection in the video colorization framework.

3.1.3 Feature Extraction

In this part of the proposal, the same video dataset that is used in the training is reused with the trained network to generate a feature vector dataset. The feature extraction process uses the trained ResNet50GeM network, as shown in Figure 6, with the key difference that the weights remain fixed (i.e., no further updates occur). Instead of saving network weights, the output consists of feature matrices representing each frame.

Theoretically, features can be extracted during the training process and saved for the next stages. In this proposal, the network is reused for feature extraction to ensure that the final optimized model extracts all the features consistently. During training, feature representations change according to the network learned, meaning that features in the early stage may not align with those in the later learned representations. The posttraining features extracted guarantee that a stable and fully optimized network is used to process all frames, leading to a more structured and reliable feature space.

To reduce processing time and frame redundancy and maintain temporal consistency, frames are processed at trained networks at regular intervals (one frame every ten frames). A structured feature dataset is created by storing the extracted feature vectors, where dissimilar images are mapped farther apart and similar images have a smaller Euclidean distance. This dataset provides feature matching efficiently, ultimately helping in the selection of the optimal reference image for color transfer.

3.1.4 Scene Analysis

In the feature extraction and training phases of the proposed network, the subsequent step emphasizes the semantic interpretation of video content through an automated framework for scene segmentation and reference image selection. This stage plays a pivotal role in ensuring that the extracted features are effectively utilized by establishing a context-sensitive correspondence between grayscale frames and their most suitable color exemplars. Given that videos inherently consist of multiple scenes characterized by variations in structure, lighting, and object composition, precise scene boundary detection becomes imperative for accurate color propagation.

To address this, a two-phase methodology is proposed. The first phase counts the number of scenes in a video via frame-by-frame structural similarity deviation metrics. The second phase leverages the previously trained feature extractor to identify the most semantically and visually aligned reference image for each detected scene. This targeted pairing between scene content and reference data strengthens the model’s learning capacity by delivering scene-specific color guidance, ultimately improving generalizability across dynamic and visually diverse video sequences.

Phase 1: Counting the number of scenes in the video

The number of scenes in a video plays a crucial role in selecting optimal reference images for accurate video colorization. Most existing studies rely on a single reference image, overlooking variations in scenes, objects, colors, and illuminations throughout the video. Although some studies have used multiple reference images, these images were manually chosen and do not correspond to the actual number of scenes present. Therefore, this paper proposes an automatic method to estimate the number of scenes in a video and, on the basis of this count, suggests an appropriate number of reference images. The overall process is illustrated in Figure 3.

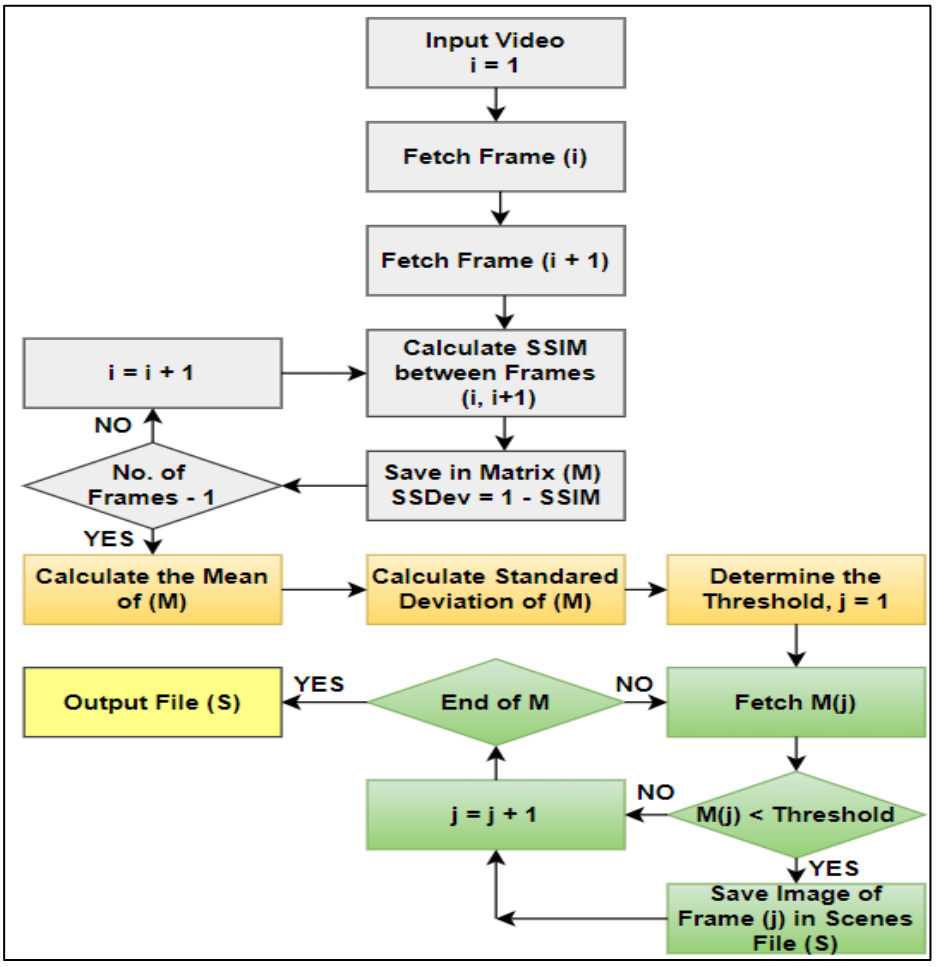


Fig. 3. Flowchart for determining the number of senses

The proposed method for counting the number of scenes in a video consists of several steps:

Step 1: The input video is split into individual frames on the basis of the video's size and format.

Step 2: Calculate the structural similarity index (SSIM) between each successive pair of frames.

Step 3: Calculate the SSIM deviation (SSDev) from the ideal SSIM value of 1 as follows:

$$SSDev = 1 - SSIM\ (previous\ frame, current\ frame) \quad \ldots \quad (2)$$

In this proposal, SSDev was used instead of the SSIM alone for many reasons:

The results are inverted, and SSDev shifts the scale, where a higher value refers to highly significant changes, making scene detection more intuitive. When directly relying on the SSIM, the major changes are indicated with small values, which may be harder to interpret consistently.

Additionally, Eq. 2 enables an adaptive threshold that is more effective than the threshold derived from the SSIM alone. The reason for this is that the SSIM values decrease when there is a change, which means that a low SSIM indicates a transition. Consequently, if the SSIM alone is used, the threshold is typically set close to zero to capture changes. However, this approach can be highly sensitive to noise, as small variations or minor disturbances may be mistakenly classified as scene changes.

Furthermore, gradual transitions in video sequences, such as fading effects or slow-motion scenes, often result in subtle SSIM variations, making it difficult to detect scene changes accurately. The accumulated deviations in SSDev provide a more robust measure for identifying significant transitions, enhancing the detection of scene boundaries.

The SSDev values for each frame are saved in a one-dimensional matrix (M).

Step 4: Compute the mean (Mean_M) of the matrix M (SSDev values), and the average of all SSDev values in the matrix M is calculated. The value of Mean_M will be low when the frame-to-frame changes are mostly gradual changes, but this value will be high when the video has frequent and abrupt changes. Therefore, the SSDev values are normalized across different types of videos. Additionally, the standard deviation (Std_M) of matrix M is computed. Std_M is calculated via the following formula (Eq. 3).

$$Std_M = \frac{\sum_{i=1}^{N}(SSDev_i - Mean\_M)}{N} \quad \ldots \quad (3)$$

where N is the matrix M size.

Step 5: Use the mean and standard deviation to calculate the threshold via Eq. (4):

$$Threshold = Mean\_M + 2.5 * Std\_M \quad \ldots \quad (4)$$

The threshold value in the proposal is different for different videos (different videos have different levels of motion). The static threshold value can be suitable for one video but fails for another.

Step 6: Compare each SSDev value in matrix M with the threshold. If SSDev is less than the threshold, consider the corresponding frame as a new scene and save the frame image to a specific file (S) with its index in matrix M; otherwise, the frame is ignored. The index of the matrix M represents the frame number in the video frame sequence when SSDev was calculated. (Note: the first frame in the video is saved in the file (S) initially as the first scene).

Phase 2: Selecting The Best Reference Images

In this phase, the trained ResNet50-GeM model is employed to select the most similar reference image from the dataset to a particular scene on the basis of the features extracted during the training phase. The input to this phase consists of the frames (images) selected in the previous phase (which represent the particular scene). For each input image, the model identifies a reference image with the closest matching features. This selection process involves multiple stages, ensuring that the chosen image is highly similar to the input in terms of feature representation. The overall flowchart of this proposal is illustrated in Figure 4. During this stage, the feature vector of the input image is extracted via the trained ResNet50GeM network. The extracted features are compared with the feature vector dataset to retrieve the optimum reference image for color transfer.

Stage 1: Image pre-processing: In this step, the input image is resized to 224×224 pixels and then normalized to be in the standard range of [0.1]. Additionally, the image is denoised via the mean filter.

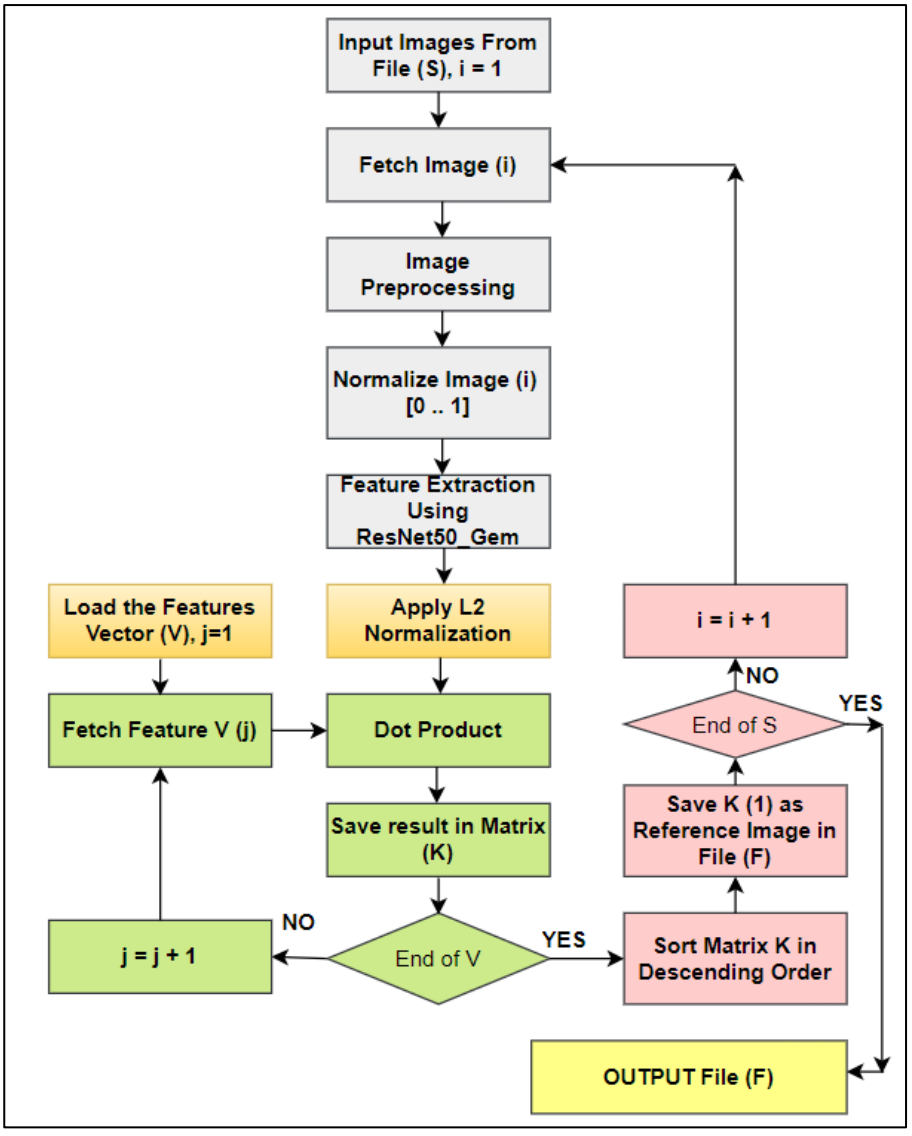


Fig. 4. Selecting the best reference image phase

Stage 2: The processed image is input to the trained ResNet50-GeM model for feature extraction. The extracted features are normalized via L2 normalization.

Stage 3: The comparison step is crucial in selecting the optimal reference image for color transfer. The feature vector of the input frame is compared with the feature dataset to find the closest match, which shares the most relevant features, such as structure, texture, illumination, or color, with the input frame.

The model looks for the most similar feature vector from the feature dataset (as mentioned before, each vector in the dataset represents an image). Any of the distance matrices can be used to measure the similarity; in this proposal, the Dot product is used to measure similarity.

Dot product formula:

$$S\,(F_{new},\quad F_i) = \sum_{i=1}^{n} (F_{new}\,F_i) \qquad \ldots\ (5)$$

where:

F_{new}: is the feature vector extracted from the input frame.

F_i: is the feature vector corresponding to the i -th reference image in the dataset.

$S\ (F_{new},\ F_i)$: is the similarity score computed via the dot product, which indicates how closely the reference image i matches the input frame.

The result of each dot product process is one value, saved in a one-dimensional similarity matrix (K). A higher similarity score (S) indicates a closer match.

Feature matching compares high-level features instead of comparing them pixel-by-pixel, as in the image, making it more efficient and robust for lighting changes, occlusions, or scaling. Additionally, it is not as sensitive to noise as direct pixel comparison is.

Stage 4: The values in similarity matrix K are sorted in descending order. The best reference image is the one with the highest similarity value, corresponding to the most similar feature vector. The returned image is used for color transfer.

Stage 5: The image corresponding to the first rank in matrix K is selected as an ideal reference image for the input video frame.

Stage 6: The previous stages are repeated for all the frames that are counted as new scenes in phase one.

3.1.5 Video Colorization

The proposed video colorization method is systematically organized into three key stages: preprocessing, reference image processing, and video colorization. These stages are designed to work synergistically, ensuring efficient processing of both images and videos, regardless of their size, while maintaining high computational performance. The process begins after selecting the best reference images for each distinct scene within the video; then, the colorization phase uses these reference images as inputs in conjunction with the corresponding grayscale video frames. The reference images are carefully selected on the basis of their semantic relevance to each scene, facilitating the colorization process by providing contextual accuracy. This ensures that the colors are not only restored in alignment with the content's semantic elements but also maintain temporal coherence across the video sequence. The structured design of this approach enables the model to generate colorized videos that accurately represent both the spatial and temporal characteristics of the original video.

Phase 1: Preprocessing

The main objective of the preprocessing phase is to guarantee that the frames used in the production of the actual video have been designed to provide accurate colorization. Preprocessing consists of three main steps, as illustrated in Figure 5, to process video frames prior to colorization while preserving temporal coherence and essential features.

Step 1: Encoding

In encoding, every frame in a video is subjected to feature extraction via a 3×3 convolutional layer. To maintain equal dimensions of the input and output, padding is applied, and a step of 2 pixels is used to reduce the frames' resolution without losing their important characteristics.

Step 2: Decoding

During decoding, the features from the encoding stage are again passed through some transformations to reconstruct the frames of the video. The strategy implemented here is trilinear upsampling and 3×3 convolution but replicates padding. There is a 1-pixel stride used to fine-tune the resolution, whereas connections between equal stages of the encoder and decoder networks minimize the noise in the upsampling step and focus on preserving the more relevant importance.

Step 3: Output

After processing all frames of the video, the outputs of the video can be considered temporally coherent and are ready for the next colorization step.

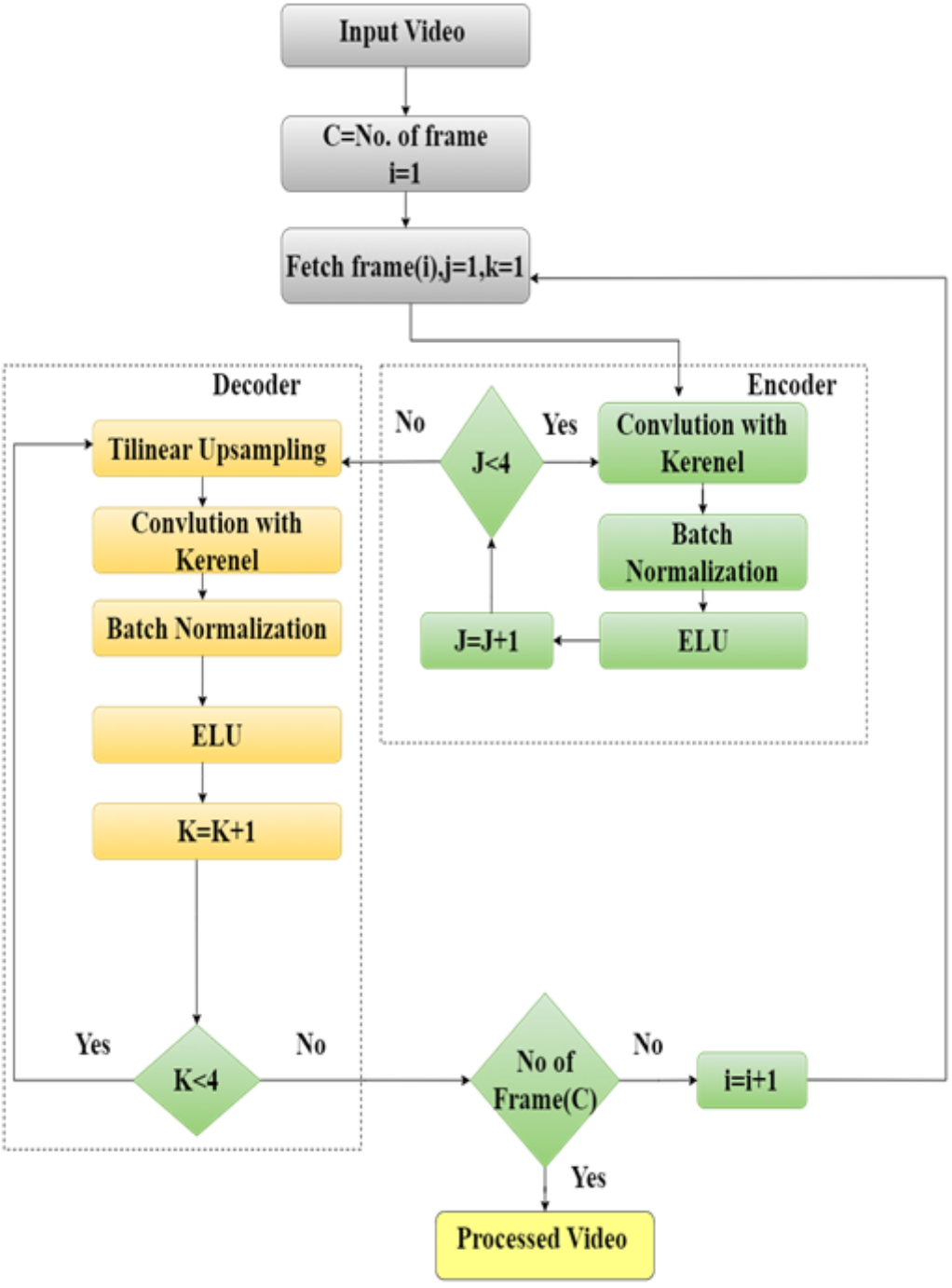


Fig 5. Flowchart of the proposed method at the preprocessing stage.

Phase 2: Reference Image Processing

The main objective of the reference image processing step is to extract unified and enhanced features from reference images to ensure consistent and accurate colorization. This step is composed of six sequential processing steps, as illustrated in Figure 6.

First step: Feature extraction via encoders

Reference images are passed through six convolutional encoder layers to extract hierarchical features. Each encoder applies a 3×3 convolutional layer with a stride of 2 and replicates padding to reduce spatial dimensions while preserving essential details. Each convolution is followed by batch normalization and ELU activation, which stabilize learning, improve generalization, and address issues such as the "dying ReLU" problem. The use of six encoders enables the progressive extraction of features ranging from low-level edges and textures to high-level semantic patterns, ensuring detailed and robust feature representations while reducing noise and inconsistencies.

Second Step: Feature Map Unification

All feature maps are resized to match the output dimensions of the fourth encoder layer to ensure consistent dimensionality across layers. This unification facilitates efficient feature fusion without unnecessary computational overhead or information loss.

Third step: Multilayer feature fusion

A fusion mechanism integrates both local and global features through two complementary processes:

- Interlayer fusion: This method combines features from nonconsecutive layers (2, 4, 6) to merge fine details with global structures.
- Intralayer fusion: Aggregates features from consecutive layers (2–6) to capture progressively abstracted representations.

The fusion of these two processes produces a comprehensive feature map (Image K), normalized to [0, 255], balancing local and global information essential for colorization.

Fourth Step: Iterative Processing of Reference Images

The above extraction and fusion processes are applied to each reference image independently to ensure consistent feature representation across all the references.

Fifth Step: Feature Refinement via Residual Blocks

The fused features are further refined through two residual blocks. Each block includes a convolutional layer, batch normalization, and ELU activation to enhance feature robustness and quality, ensuring that essential color and texture details are well preserved.

Sixth Step: Final Output Preparation

The final output features are standardized to a fixed dimension (H, W, channels) and normalized to the [0, 255] range, making them ready for use in the colorization stage.

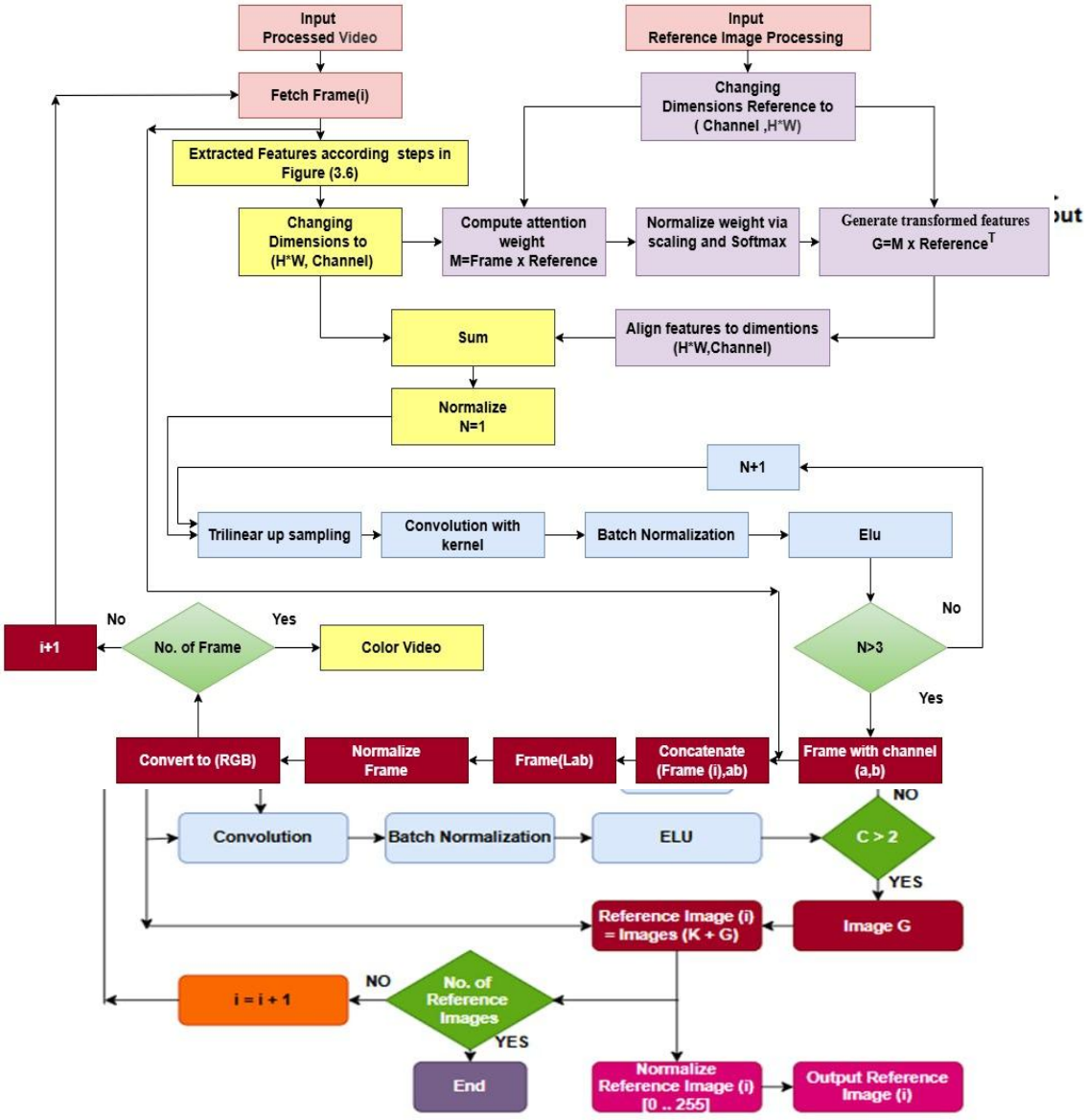


Fig. 6. Process Reference Images.

Phase 3: Video Colorization

The primary objective of video colorization is to ensure that each frame of a grayscale video is accurately colorized while maintaining temporal coherence throughout the entire video sequence. The colorization process included three steps, as illustrated in Figure 7.

Step 1: Video Frame Processing

In the first stage, after denoising and enhancing video quality Stage 1, the video is split into individual frames. These frames undergo processing similar to reference images in Stage 2, where their size and features are adjusted to extract key characteristics, ensuring compatibility for the subsequent colorization stages.

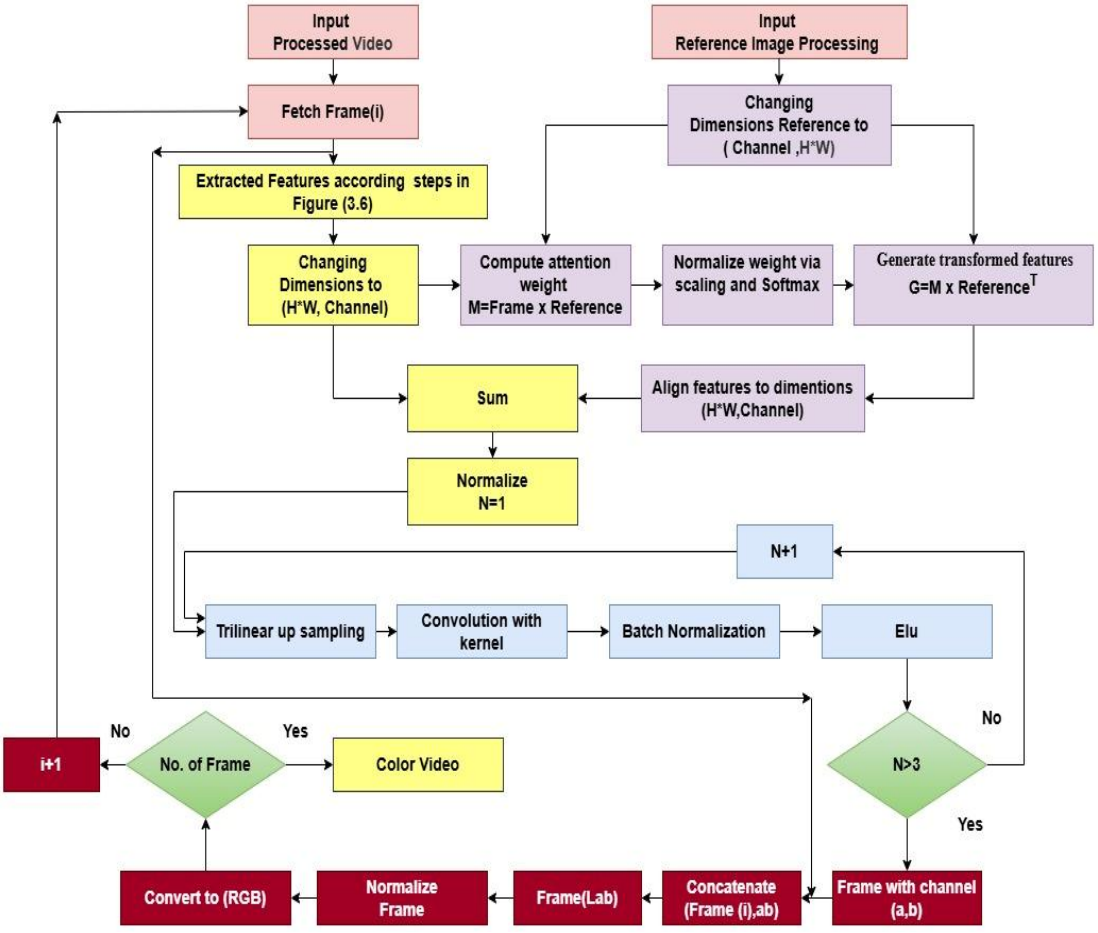


Fig 7. Process Video and Colorization

Step 2: Source-Reference Attention

This stage focuses on integrating the processed video frames with reference images via a source-reference attention layer. The steps are as follows:

1. Input to the attention layer: The attention layer receives inputs from both the processed video frames and reference images. A matrix-based attention mechanism is used to establish a correspondence between each pixel in the video frame and the reference image, ensuring accurate color transfer while maintaining video characteristics.
2. Dimensional Transformation for Matrix Multiplication: To ensure compatibility in matrix operations, the reference images undergo dimensional transformations into two distinct representations:

- First Representation: The reference image is flattened into a 2D matrix with dimensions (H×W, C), where each row corresponds to a color channel. This representation captures the spatial and color information of the image.
- Second Representation: The transpose of the first representation is taken, yielding dimensions (C, H×W). This transformation allows for efficient matrix multiplication by aligning the reference image with the video frame.

For compatibility during matrix operations, the video frames are also converted into the same first representation format (H×W, C).

3. Attention Weights Calculation with Normalization: The attention mechanism focuses on computing the relevance of each pixel in the video frame to the reference image. The attention weights are computed by multiplying the reference image with dimensions (C, H×W) by the video frame with dimensions (H×W, C). This results in a weight matrix that reflects the relationship between the reference image's color information and the pixels of the video frame.

To maintain numerical stability and prevent overflow during matrix multiplication, a normalization procedure is applied on the basis of the number of columns in the resulting matrix, M:

- The number of columns in M is extracted, and its square root is computed to serve as the scaling factor.
- This normalization ensures that the values obtained after matrix multiplication remain within a manageable range, contributing to the stability of the neural network during training.
- A Softmax function is then applied to the weights, converting them into probabilistic values. These values indicate the influence of each reference pixel on each pixel in the video frame.

4. Calculating the Transformed Features: Once the attention weights are computed, they are applied to the reference image with dimensions (H×W, C) to produce transformed features. These transformed features, which are aligned with the original video frame information, are then combined with the video frame's own features. The dimensions of the resulting features remain (H, W, C).

To maintain consistent color representation, the final output is normalized within the pixel value range of $[0, 255]$, ensuring that the transformed features are appropriately scaled for display or further processing.
5. Processing All Frames: This process is repeated for all video frames, allowing each frame to integrate color information from multiple reference images and improving color accuracy across the video.

6. Temporal Consistency: To maintain smooth transitions between frames, the transformed features pass through a temporal attention layer, ensuring temporal consistency and reducing color inconsistencies between consecutive frames.

7. Decoder Step: The processed frames undergo the following operations:
- Trilinear upsampling: This method ensures smooth transitions by extending bilinear interpolation to three dimensions, minimizing temporal inconsistencies.
- Convolution with a 3×3 Kernel: Enhances detail and color accuracy.
- Replicate Padding: This preserves the image size and edge information.
- Batch normalization and ELU activation: Stabilizes training and accelerates learning.

The process is repeated for each frame until all frames are processed.

Step 3: Processing and Output
After the decoder stage, the chrominance channels (A and B in the Lab color space) are extracted and combined with the preprocessed grayscale video. The combined result undergoes normalization for consistent color alignment across the video. Finally, the video is converted from the Lab color space to RGB, producing a fully colorized video with preserved color accuracy and temporal coherence.

3.2 Deep Learning Algorithms

Deep learning algorithms encompass a diverse range of computational models that collectively contribute to achieving the intended objectives of this research. These algorithms include the following:

3.2.1 Residual Networks with 50 Layers

The residual network (ResNet) helps to avoid the vanishing gradient problem in deep networks through the use of residual connections. Unlike the VGG architecture, which comprises 19 layers, the ResNet architecture can reach 152 layers, making it much deeper. The proposed ResNet reached the first position in the ILSVRC and COCO competitions for image classification, object detectability, and segmentation in 2015.

Figure 8 shows the ResNet50-vd model, which consists of a parent block, four residual blocks and a fully connected block. The parent module comprises three 3×3 convolutional layers and a max pool, which halves the spatial dimensions and doubles the number of channels to 64. The network evolution occurs through four phases, of which the first phase is the downsampling phase, whereas the second phase includes many units of the feature extraction phase. Block 1 employs a bottleneck design with three Conv ReLU operations in the primary branch and pooling in the second branch to achieve dimensional parity. Shortcut connections in Block 2 are also integrated to form the residual module as a core part. The feature map sizes gradually decrease from 112×112 to 7×7, and the channel depths increase from 64 to 2048. After that, the output is presented to the global average pooling layer and the fully connected layer for classification.

To enhance the ability to track the deformation of the object, deformable convolution (DCN-v2) replaces common 3 × 3 convolutions in Stages 2 to 4 to include several learnable offsets that can dynamically alter the sampling position. This improves the network scale and rotation capability, namely, 13 deformable layers are used to greatly improve the ability to recognize distorted objects [10].

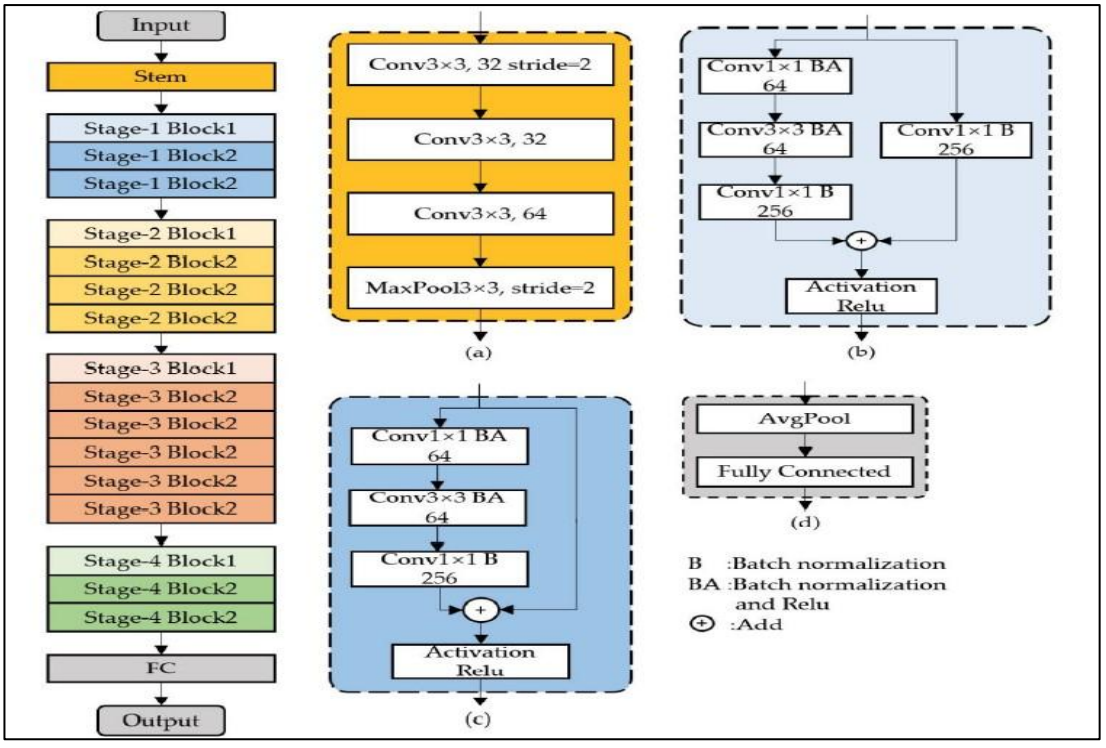


Fig 8. Resnet50 Architecture[11]

3.2.2 Generalized mean pooling

GeM stands for generalized mean pooling and is a general form of standard pooling operation with the help of the generalized mean. This pooling parameter p can be a deterministic value or a learnable parameter, although the latter is often initialized alike across the channels to ensure some stability in training. The GeM operation is defined mathematically as:

$$\text{Gem}(x)=\left(\frac{1}{N}\sum_{i=1}^{N} x_i^p\right)^{\frac{1}{p}} \qquad (6)$$

When $p = 1$, GeM simplifies to average pooling, whereas for large values of p, it approximates max pooling. Given an input image $I \in R^{h\times w\times 3}$, the output feature map $F \in R^{H\times W\times C}$ is processed through the GeM layer to produce a final descriptor $F \in R^{C\times 1}$.

Higher values of p lead to sharper, localized feature representation, which helps in capturing the significant features in the input image[12]. In this work, p = 3.0 is set, which provides a measure of the average and maximum pooling. This decision also enables the pooling layer to highlight crucial features and retain useful information from the vast feature map to enhance the feature map.

3.2.3 Encoder-decoder network

An encoder-decoder is a neural network structure that is applied in most sequences to sequence problems such as machine translation and image captioning. As illustrated in Figure 9, the model consists of two primary components: the encoder and the decoder. An encoder takes an input sequence and embeds it into a fixed size context vector, which preserves all the information about the input data. This context vector is then fed through the decoder, which then produces the output sequence depending on the information encoded.

Here, the encoding stage entails transforming data input into a numerical structure after the mapping of the features as well as the structural connections within the data have been retained. On the other hand, the decoder receives the encoded representation and tries to reconstruct it in the form of the target output that can be similar to the input or can represent something related to it. To enhance the output, 'attention' techniques are incorporated and are helpful in making the result more accurate and relevant. These mechanisms enable the decoder to attend to some parts of the encoded information and develop associations between the input and output sequences that can improve the model's performance [13].

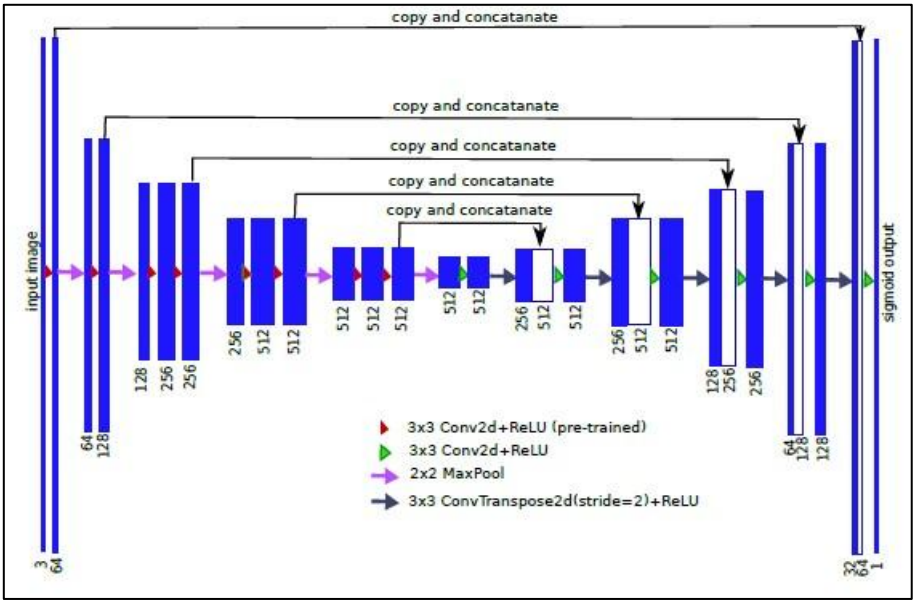


Fig 9. Encoder-decoder architecture [14].

3.2.4 Multilayer feature fusion for local and global features(MFLL)

This proposal presents two unique methods of incorporating multilevel feature layer learning (MFLL) into different layers of a CNN architecture. The first form of fusion, known as intralayer fusion, aims at improving the feature components in a single-layer network. Thus, the enhanced information enriches localized features and is passed through the next layers, enhancing the network's ability to detect fine details.

The feature maps of Layers 1 and 2, which are shown in Figure 10, are made up of multiple convolutional sublayers. The intralayer fusion module smoothens the features of these layers and transmits improved features to Layer 3. It helps enhance the representation of the features that were learned in the first layer and the second layer to enhance general feature learning across the network. Moreover, various types of fusion modules can be used to adjust or further improve some local properties when needed for this approach to be suited to various network structures and objectives.

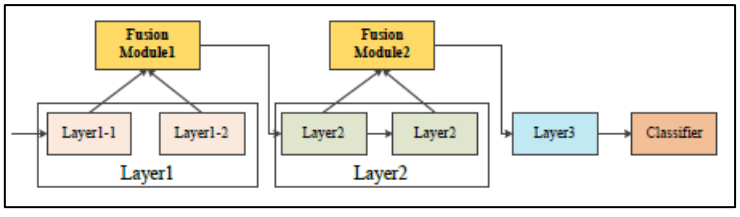


Fig 10. Two intralayer fusion modules work to enhance different parts of the local parts

The second technique, interlayer fusion, is centred on augmenting global feature information across different layers of the CNN. When going forward, some features of detail are summed, and the layer fusion integrates the features from different layers into a single map. This method aims at the fusion of the global features, which also helps increase the variety of the final result. In the case of interlayer fusion, information from various layers has the potential to reconstruct or even attenuate the loss of features during the networking process.

As illustrated in Figure 11, the module gathers feature maps from three different groups of convolutional layers. These are then forwarded to a classifier to produce the final classification result given the combination of the two. Typically, only one fusion module is used in the CNN to achieve this goal, which is a simple and efficient way to enhance feature representation and classification[15].

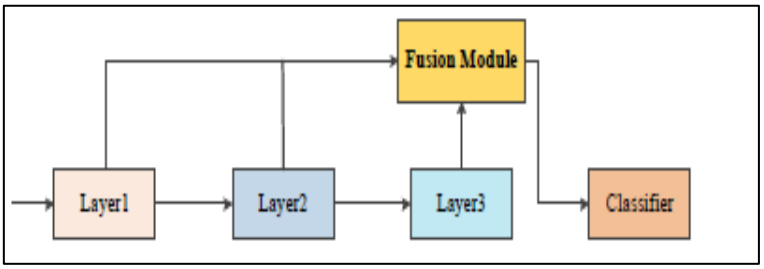


Fig. 11. Interlayer fusion to enhance the global feature

3.2.5 Source-reference attention layer

Called the cross-modal attention or cross-modal interaction layer, the source-reference attention layer is another crucial part of the neural network that connects the information in the reference feature maps with the source feature maps. This approach is particularly beneficial when data from an auxiliary input (the reference) can aid in the representation of the primary input (the source). The source–reference attention layer contains the following key steps [16] [17]:

1. **Projection into a common feature space:** The source and reference feature maps are initially projected into a common feature space through linear transformations. This process generates three projections: queries (Q), keys (K), and values (V).

- Queries (Q): These are derived from the source feature maps via a function f_q, which is typically a convolutional or linear layer:

$$Q=fq(hs) \quad (7)$$

- Keys (K) and Values (V): These are projected from the reference feature maps via functions f_k and f_v, respectively:

$$K=fk(hr) \quad (8)$$
$$V=fv(hr) \quad (9)$$

All three projections (Q, K, V) generally share a common dimensionality d.

2. **Attention score computation:** The next step is to calculate the similarity scores (attention scores) between each pair of projected features from the source and reference. This is typically done by taking the dot product:

$$\mathrm{A_{ij}} = \mathrm{Q_i}.\mathrm{K_j^T} \quad (10)$$

The resulting scores are then normalized, often via the Softmax function, to obtain weights that sum to 1 along the reference dimension:

$$\alpha_{ij} = \frac{\exp(A_{ij})}{\sum_j \exp(A_{ij})} \quad (11)$$

3. **Weighted aggregation:** Using the attention weights αij, a weighted sum of the reference features Vis is computed for each location in the source feature map:

$$h'_{si} = \alpha_{ij} V_j \quad (12)$$

This step allows the relevant information from the reference feature maps to be incorporated into the source feature maps.

4. **Combining with source features:** Finally, the transformed source features hs' are combined with the original source features hs. This is typically done through elementwise addition:

$$h''_s = h'_s \oplus h_s \quad (13)$$

This combination preserves the local details of the source features while enhancing them with the contextual information from the reference features, thereby improving the network's ability to make more informed predictions.

3.3 Evolution Metrics

Metrics play a crucial role in selecting the efficiency of reference image selection for grayscale videos. These metrics are used for evaluating the performance between the predicted outcome and ground truth data for improving video color depth. The tools used in this proposal to measure model performance are as follows:

3.3.1 Precision

Precision measures how many of the total images retrieved are relevant to the query, leveraging the assessment of the results in relation to the input [18]. It is mathematically represented as:

$$P = TP / (TP + FP) \quad (14)$$

where TP is a true positive and FP is a false positive.

3.3.2 Recall

Recall determines the number of relevant images identified and retrieved over the total number of relevant images existing [19]. The formula for the recall is as follows:

$$R = TP / (TP + FN) \quad (15)$$

where FN represents false negatives.

3.3.3 Mean average precision (MAP)

The MAP measures the quality of the ranked results by taking the average of the precision measures for several recall levels [20]. It is used as one overall evaluation of retrieval efficiency across the threshold range.

$$MAP = \Sigma(\text{Average Precision}(\text{query})) / \text{Number of Queries} \quad (16)$$

3.3.4 F1 score

The F1 score has a harmonic mean of precision and recall, and as far as the number of false positive and false negative results are concerned, it offers both a satisfactory measure [21]. It is expressed as:

$$F1 = 2 * (P * R) / (P + R) \quad (17)$$

3.3.5 Accuracy

Accuracy is a measure of system effectiveness because it quantifies the rate of correct predictions between relevant and irrelevant as a proportion of all the predictions made [22]. It is defined as:

$$\text{Accuracy} = (TP + TN) / (TP + FP + FN + TN) \quad (18)$$

where TP is a true positive, FP is a false positive, FN is a false negative, and TN is a true negative.

3.3.6 Peak signal-to-noise ratio (PSNR)

The most commonly used measure for a comparison or contrast of a picture or a video sequence of identical size with its corresponding original copy is the peak signal-to-noise ratio (PSNR). The PSNR is calculated via the following equation:

$$\text{PSNR} = 10 \log_{10} \left(\frac{M^2}{\text{MSE}} \right) \quad (19)$$

where M represents the maximum possible pixel value of the image. For an 8-bit image, M is set to 255. A higher PSNR value indicates superior reconstruction quality, suggesting minimal distortion[23].

3.3.7 Average difference (AD)

The average difference (AD) metric is the mean of the differences between the original image and the corresponding colorized image. It is mathematically defined as:

$$\mathrm{AD} = \frac{1}{\mathrm{XY}}\sum_{\mathrm{p=1}}^{\mathrm{X}}\sum_{\mathrm{q=1}}^{\mathrm{Y}}[\mathrm{K(p,q)} - \mathrm{Q(p,q)}] \quad (20)$$

where K(p,q) and Q(p,q) represent the pixel intensities of the original and colorized images, respectively. The optimal AD value is zero, indicating no difference[24].

3.3.8 Root mean square error (RMSE)

The root mean square error (RMSE) provides information related to the standard deviation of the difference between the predicted image and the actual image and is used to evaluate the quality of the reconstructed images. It is defined as:

$$\mathrm{RMSE} = \sqrt{\frac{1}{XY}\sum_{p=1}^{X}\sum_{q=1}^{Y}(K(p,q) - Q(p,q))^2} \quad (21)$$

A lower RMSE value signifies a more accurate reconstruction[25].

3.3.9 Maximum difference (MD)

The maximum difference (MD) measures the maximum difference of a single pixel between the original image and the modified image. It is given by:

$$\mathrm{MD} = \max|\mathrm{K(p,q)} - \mathrm{Q(p,q)}| \quad (22)$$

Higher MD values indicate greater discrepancies and reduced image quality[26].

3.3.10 Structural Content (SC)

The structural content (SC) quantifies the structural similarity of two images. It is computed via the following formula:

$$\mathrm{SC} = \frac{\sum_{p=1}^{X}\sum_{q=1}^{Y}(Q(p,q))^2}{\sum_{p=1}^{X}\sum_{q=1}^{Y}(K(p,q))^2} \quad (23)$$

A higher SC value implies poorer image reconstruction[27].

3.3.11 Normalized absolute error (NAE)

The normalized absolute error (NAE) assesses the absolute difference between the original and reconstructed images, normalized by the sum of the original pixel intensities[28]:

$$\mathrm{NAE} = \frac{\sum_{\mathrm{p=1}}^{\mathrm{X}}\sum_{\mathrm{q=1}}^{\mathrm{Y}}|\mathrm{K(p,q)-Q(p,q)}|}{\sum_{\mathrm{p=1}}^{\mathrm{X}}\sum_{\mathrm{q=1}}^{\mathrm{Y}}\mathrm{K(p,q)}} \quad (24)$$

An NAE value close to zero indicates high image quality.

3.3.12 Normalized Cross-Correlation (NCC)

Normalized cross correlation (NCC) assesses the degree of between the original and altered images. It is defined as[29]:

$$\mathrm{NCC} = \sum_{p=1}^{X}\sum_{q=1}^{Y}\frac{K(p,q) * Q(p,q)}{(K(p,q))^2} \quad (25)$$

Higher NCC values signify greater similarity.

3.3.13 Structural Similarity Index (SSIM)

The structural similarity index (SSIM) represents the formula used to measure the perceived quality of images. It is calculated as:

$$\mathrm{SSIM(p,q)} = [K(p,q)]^{\alpha}.[Q(p,q)]^{\beta}.[W(p,q)]^{\gamma} \quad (26)$$

where α, β, and γ are parameters controlling the influence of luminance, contrast, and structure, respectively.

The luminance, contrast, and structure components are defined as follows:

$$\mathrm{K(p,q)} = \frac{2*\mathrm{M_p}*\mathrm{M_q}+\mathrm{S_1}}{(\mathrm{M_p^2}+\mathrm{M_q^2}+\mathrm{S_1})} \quad (27)$$

$$\mathrm{Q(p,q)} = \frac{2 * \sigma_{\mathrm{p}} * \sigma_{\mathrm{q}} + \mathrm{S_2}}{(\sigma_{\mathrm{p}}^2 + \sigma_{\mathrm{q}}^2 + \mathrm{S_2})} \quad (28)$$

$$W(p,q) = \frac{\sigma_{pq} + S_3}{(\sigma_p * \sigma_q + S_3)} \quad (29)$$

where the constants S_1, S_2 and S_3 are utilized to prevent any instabilities that may average from the pixel value ($M_p^2 + M_q^2$), standard deviation ($\sigma_p^2 + \sigma_q^2$) or ($\sigma_p * \sigma_q$) approaches zero. The SSIM (p, q) varies between 0 (indicating dissimilarity). 1 (representing identical patches)[30].

3.3.14 Pearson correlation coefficient (PCC)

The Pearson correlation coefficient (PCC) measures the strength and direction of the linear relationship between two images, yielding values in the range [-1, 1]:

$$PCC = \frac{p(\sum KQ) - \sum K * \sum Q}{\sqrt{[p\sum K^2 - (\sum K)^2] * [p\sum Q^2 - (\sum Q)^2]}} \quad (30)$$

Positive PCC values indicate a direct relationship, whereas negative values signify an inverse relationship[31].

4. RESULTS

4.1. Experimental Environment

All experiments were conducted on a workstation equipped with an NVIDIA RTX 3090 GPU (24 GB VRAM), Intel Core i9-12900K CPU, and 128 GB RAM, running Ubuntu 22.04 LTS. Model development and training were implemented using PyTorch 2.0 with CUDA 12.1 acceleration. For deployment testing, the model was quantized and optimized using TensorRT 8.6 on a Raspberry Pi 4 (8 GB) running Ubuntu Server 20.04, to validate real-time feasibility on embedded devices.

4.2. Scene Segmentation and Reference Image Selection

The initial phase of the proposed framework involves segmenting video content into distinct scenes via the structural similarity index measure (SSIM) computed between consecutive frames. Adaptive thresholding, informed by statistical analysis, was employed to accurately delineate scene boundaries. To evaluate the efficacy of this segmentation approach, experiments were conducted on ten diverse video sequences, each containing varying numbers of scenes.

The segmentation performance was assessed in terms of precision, recall, F1 score, and mean average precision (MAP). For example, when four scenes are extracted, the method achieves a mean precision of 0.9960, a recall of 0.9850, an F1 score of 0.9905, and a MAP of 0.9952. As the number of scenes increased to five, the precision slightly decreased to 0.9900, whereas the recall improved to 0.9925, resulting in an F1 score of 0.9912 and a MAP of 0.9918. Similar trends were observed for six and ten scenes, demonstrating the method's robustness and adaptability across varying scene complexities.

Subsequent to scene segmentation, the framework selects the most contextually appropriate reference images for each scene. This selection is facilitated by a customized ResNet-50 network integrated with generalized mean pooling (GeM), which extracts deep features from both scene segments and a pool of candidate reference images. The similarity between features is quantified via the dot product, enabling the dynamic selection of optimal reference images that align with the semantic content of each scene.

4.3. Training and Performance of the Reference Selection Model

The reference selection model underwent training over 100 epochs, with continuous monitoring of accuracy and loss metrics. As depicted in Figure 12, the model exhibited a steady increase in accuracy, stabilizing at approximately 97% by the 100th epoch. Concurrently, the loss values demonstrated a consistent decline, indicating effective optimization of the feature space and confirming the model's ability to learn discriminative features pertinent to reference image selection.

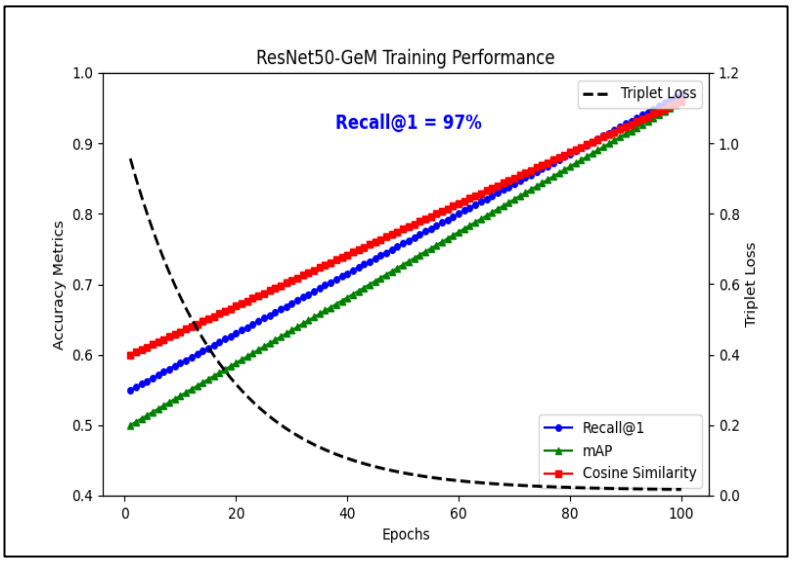


Fig. 12. Training process flow of the ResNet50-Gem model

4.4. Training and Evaluation of the Colorization Model

Following the successful training of the reference selection model, the second phase focused on the colorization of grayscale video frames. By utilizing the selected reference images, the colorization model was trained for an additional 100 epochs. Figure 13 illustrates the training progression, where the model achieved an accuracy of 99% by the final epoch, underscoring its proficiency in generating high-quality colorized outputs.

The colorization model comprises three trainable modules: (1) a preprocessing module with an encoder-decoder structure enhanced by temporal convolutions, skip connections, batch normalization, and ELU activations; (2) a reference image processing module; and (3) a colorization module featuring a source-reference fusion mechanism built upon residual blocks and attention layers. This architecture facilitates the integration of temporal and contextual information, ensuring both structural and color coherence in colorized videos.

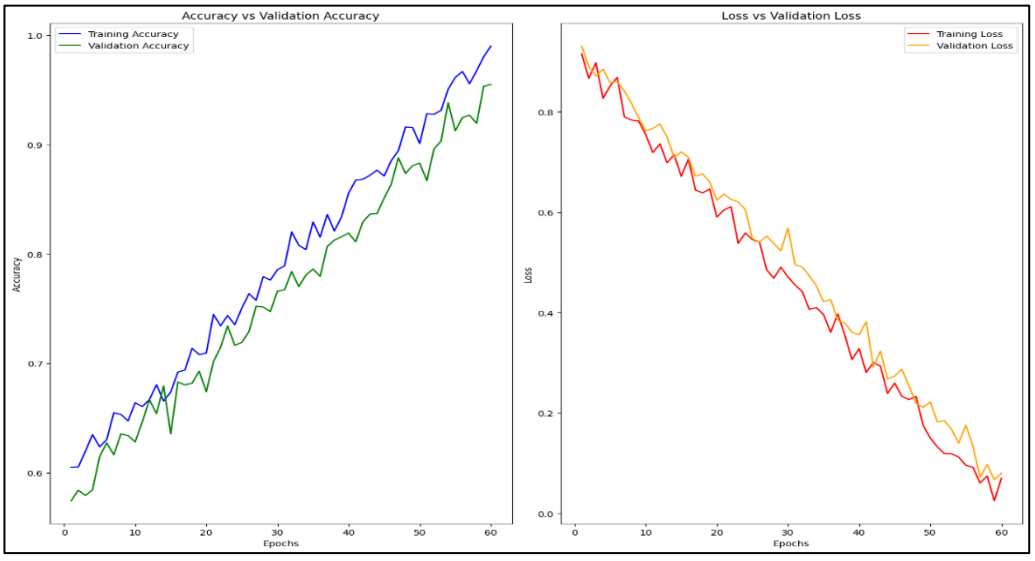


Fig. 13. Training colorization Model

4.5. Quantitative and Qualitative Assessment of Colour Quality

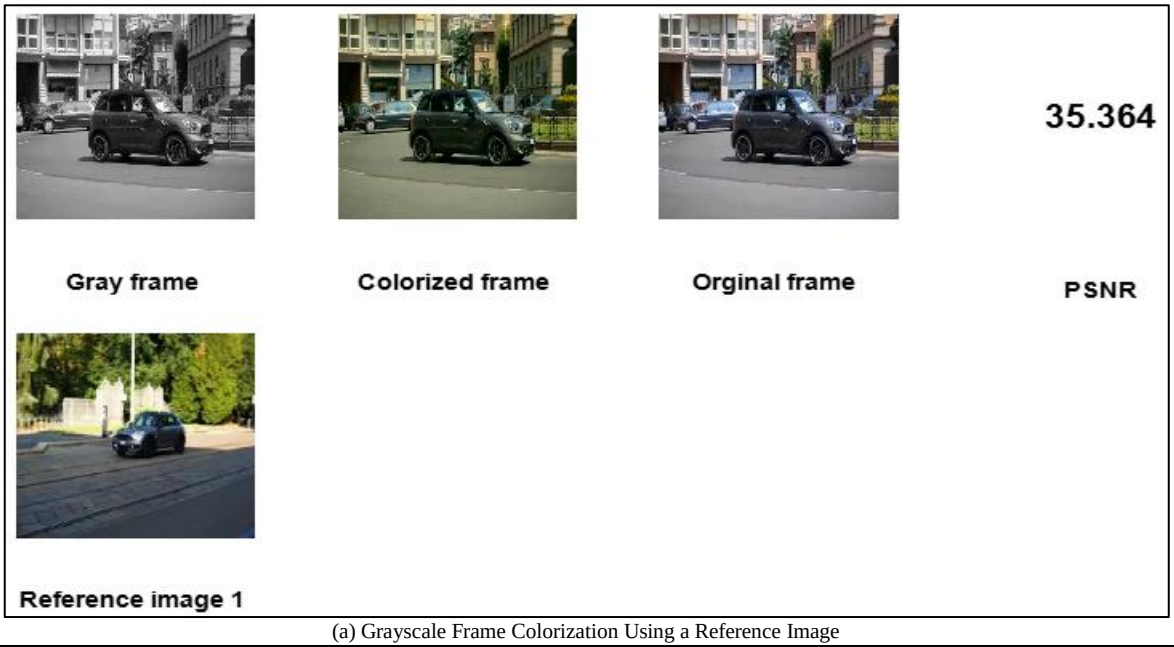
To quantitatively assess colorization quality, a comprehensive evaluation was conducted via metrics such as the peak signal-to-noise ratio (PSNR), structural similarity index (SSIM), and other relevant measures. Table III presents the performance metrics across different numbers of reference images utilized during colorization.

TABLE III. COMPARISON OF VIDEO QUALITY AFTER THE COLORIZATION PROCEDURE VIA VARIOUS MEASURES

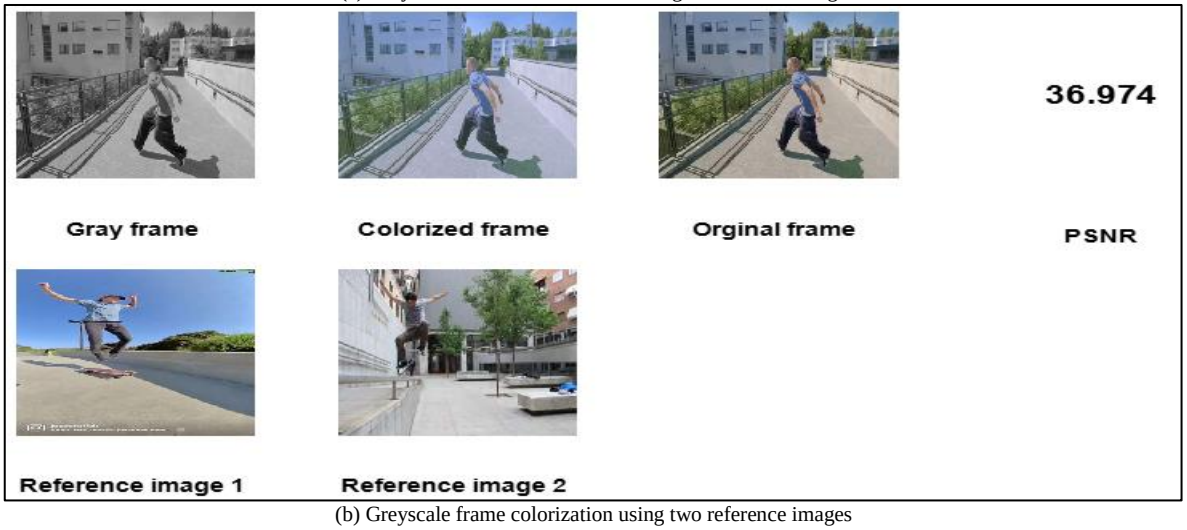
Number of reference	AD	RMSE	MD	SC	NAE	NCC	SSIM	PCC
1	0.45	4.443	1.362	0.945	0.893	0.887	0.980	0.984
2	0.31	3.719	0.481	0.902	0.732	0.896	0.987	0.989
3	0.25	3.372	0.254	0.821	0.624	0.99	0.993	0.993
4	0.22	3.340	0.131	0.758	0.518	0.993	0.996	0.995
5	0.14	3.328	0.138	0.673	0.357	0.998	0.998	0.997

The results indicate that increasing the number of reference images enhances the colorization quality, as evidenced by improvements in the PSNR, SSIM, and other metrics. Notably, the use of five reference images yielded the highest performance across all evaluated measures.

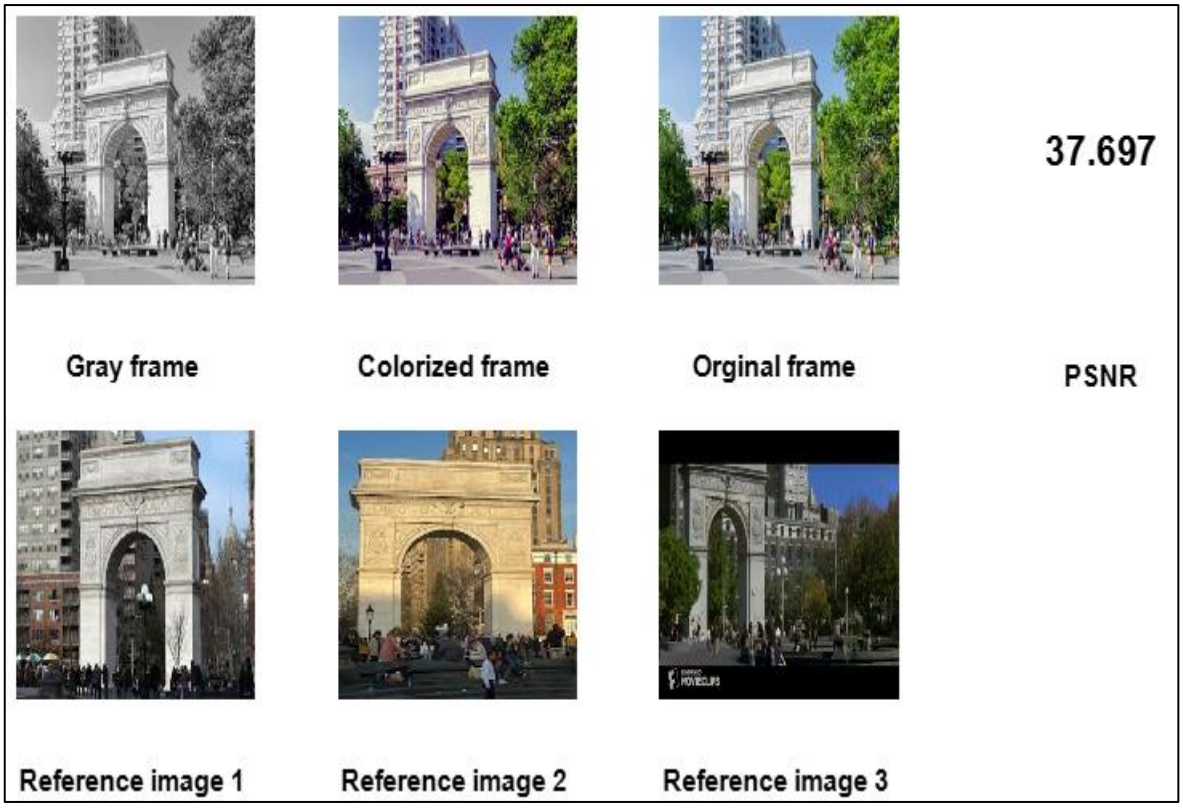
Qualitative assessments further corroborate these findings. Figure 14 shows the visual outcomes of colorizing grayscale frames using varying numbers of reference images. The progression from one to five reference images demonstrates a marked improvement in color richness and structural fidelity, highlighting the model's ability to produce visually compelling and semantically coherent colorizations.



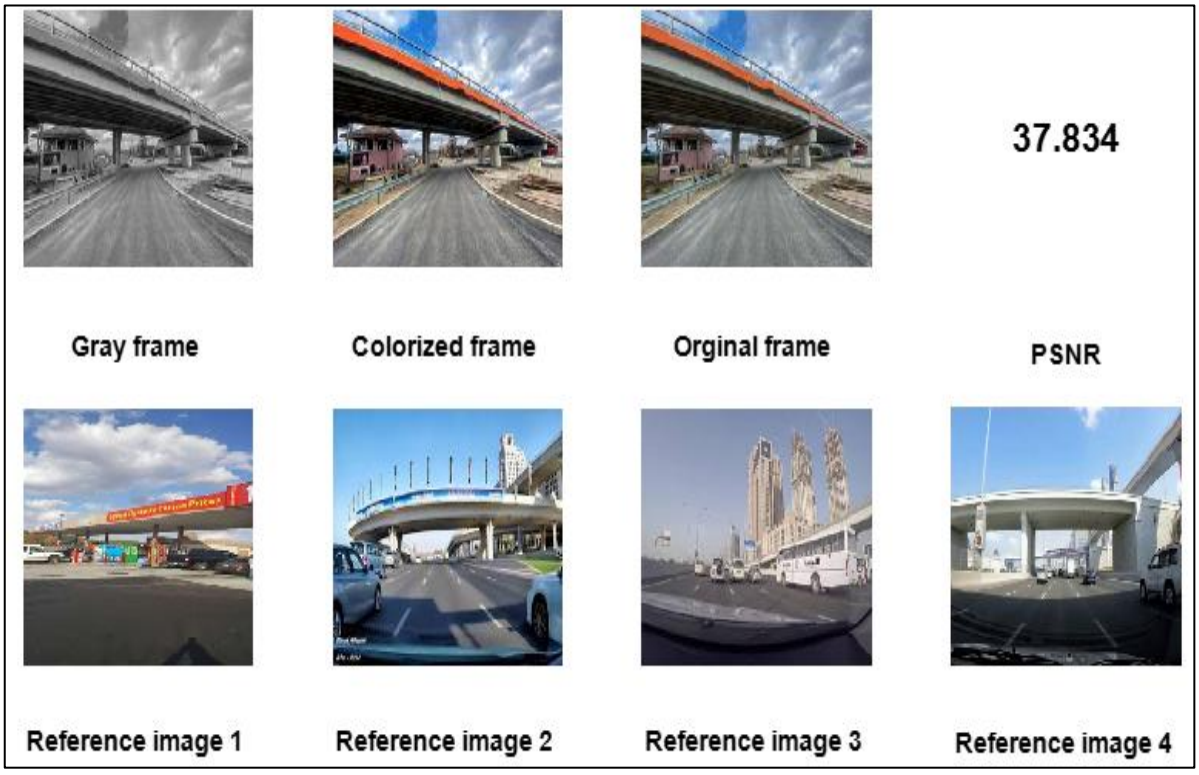
(a) Grayscale Frame Colorization Using a Reference Image



(b) Greyscale frame colorization using two reference images



(c) Grayscale frame colorization using three reference images



(d) Grayscale frame colorization using the reference image

(e) Grayscale frame colorization using five reference images

Fig. 14. PSNR Values for Videos Colorized via the Proposed Method with One, Two, Three, Four, and Five Reference Images. Subfigures (a) through (e) Demonstrate greyscale frame colorization via five reference images

4.5. Comparative Analysis with Existing Methods

A visual comparison, shown in Figure 15, further highlights the superior results achieved by the proposed method in contrast to earlier studies. The enhancements in both color fidelity and image realism affirm the model's practical potential for high-quality video colorization.

Fig. 15. Comparison of the proposed colorization process

Table IV presents a quantitative comparative assessment of various existing methods. The proposed approach achieves the highest PSNR (37.89) and competitive SSIM (0.998), significantly surpassing those of prior methods. This improvement reflects the model's superior ability to reconstruct the original colored content from grayscale inputs.

TABLE IV. COMPARISON OF VIDEO QUALITY AFTER THE COLORIZATION PROCEDURE USING MEASURES

Reference	AD	RMSE	MD	SC	NAE	NCC	SSIM	PCC	PSNR
[32]	**1.069126**	**1.806177**	**20**	**0.994602**	**0.009098**	**0.981267**	**0.963565**	**0.965124**	**34.57**
[33]	**1.068921**	**1.805696**	**20**	**0.994237**	**0.008951**	**0.98136**	**0.963755**	**0.976798**	**34.81**
[34]	**1.024832**	**1.556213**	**20**	**0.981382**	**0.008887**	**0.983618**	**0.96376**	**0.986665**	**35.12**
[35]	**0.872693**	**1.266927**	**18**	**0.815632**	**0.008885**	**0.983627**	**0.982004**	**0.986685**	**35.31**
[36]	**0.569565**	**0.992709**	**12**	**0.732995**	**0.006713**	**0.985156**	**0.98229**	**0.986916**	**35.56**
[37]	**0.541809**	**0.991881**	**11**	**0.631692**	**0.006386**	**0.987824**	**0.987129**	**0.988392**	**35.8**
[7]	**0.52446**	**0.97283**	**10**	**0.536527**	**0.006181**	**0.988699**	**0.987278**	**0.988924**	**36.21**
[38]	**0.022581**	**0.228494**	**10**	**0.346817**	**0.000266**	**0.988794**	**0.999852**	**0.989313**	**36.5**
[39]	**0.016055**	**0.198013**	**10**	**0.317304**	**0.000189**	**0.988845**	**0.999959**	**0.990797**	**36.98**
our	**0.009084**	**0.124658**	**8**	**0.276279**	**0.000107**	**0.991476**	**0.999982**	**0.998678**	**37.89**

Finally, Table V benchmarks the proposed model against other contemporary deep learning-based methods, including BiSTNet, LVCD, and ViT-Color. The proposed method outperforms these models in PSNR and SSIM while also offering faster inference (2.6 seconds per frame) and maintaining a moderate model size (81 MB). This balance of performance and efficiency renders the method highly suitable for practical applications requiring both high fidelity and computational viability.

TABLE V. COMPARATIVE ANALYSIS OF PSNR AND SSIM METRICS ACROSS RECENT DEEP LEARNING-BASED VIDEO COLORIZATION METHODS

Reference	Method	PSNR (↑)	SSIM (↑)	Computation Time (↓)	Model Size (MB)
[6]	**BiSTNet**	**30.40**	**0.976**	**3.2 sec/frame**	**78**
[8]	**LVCD**	**24.5489**	**0.8790**	**2.9 sec/frame**	**85**
[40]	**ViT-Color**	**28.88**	**0.979**	**4.5 sec/frame**	**112**
our	**Proposed**	**37.89**	**0.998**	**2.6 sec/frame**	**81**

The proposed method outperforms existing techniques in terms of PSNR and SSIM, indicating superior reconstruction quality and structural similarity. Additionally, it achieves a lower computation time per frame while maintaining a moderate model size, demonstrating an effective balance between accuracy and efficiency.

5. CONCLUTION

This study introduces a novel, fully automated framework for video colorization that strategically leverages adaptive reference image selection to ensure accurate and temporally consistent color transfer across video sequences. Unlike conventional methods that rely on static reference images, the proposed approach dynamically aligns reference frames with

target video content, resulting in enhanced chromatic precision and reduced visual artifacts, particularly in videos containing diverse scenes.

The core architecture is built upon a combination of ResNet50 and generalized mean (GeM) pooling. ResNet50 enables the extraction of rich, multiscale visual features from grayscale frames, whereas GeM pooling highlights the most discriminative regions, facilitating the selection of semantically relevant reference images. This synergy between structural representation and salient region focus allows the system to guide color transfer with greater contextual awareness.

The framework operates through three tightly integrated phases. The initial preprocessing phase refines the grayscale video input by suppressing noise and improving frame clarity, establishing a solid foundation for subsequent processing. The reference image processing phase then uses deep feature extraction and attention-driven pooling to identify reference images that are optimally aligned with the target video scenes. Finally, the colorization process is carried out through a source–reference matching network that incorporates multilayer feature fusion, bidirectional optical flow, and attention mechanisms. This network ensures spatially accurate and temporally coherent color propagation, even in the presence of scene transitions and complex motion.

The method demonstrates significant improvements in color fidelity and consistency across frames, achieving up to 99% accuracy in experimental evaluations without relying on extensive data augmentation. Moreover, by exploiting the natural variability within video datasets, the model inherently mitigates overfitting and promotes generalization. The bidirectional optical flow component plays a pivotal role in maintaining smooth temporal transitions, contributing to the overall visual realism of the output.

In summary, this research addresses key challenges in video colorization—namely, achieving high color precision and temporal coherence—through a robust, scalable, and end-to-end trainable solution. The integration of ResNet50, GeM pooling, and advanced alignment techniques positions this framework as a promising tool for enhancing colorization quality in diverse video applications.

Conflicts of interest

The authors declare that they have no conflicts of interest.

Funding

The acknowledgements section of the paper does not mention any financial support from institutions or sponsors.

Acknowledgement

We would like to express our sincere gratitude to everyone who provided invaluable guidance and insightful feedback throughout the course of this research. The expertise and advice we received were crucial in shaping the direction of our work and ensuring its successful completion.

References

[1] P. N. Korat and D. R. Bhojani, "A new hybrid block based motion estimation algorithm for video compression," Int. J. Adv. Res. Electr. Electron. Instrum. Eng., vol. 3, no. 5, pp. 9586–9596, 2014.

[2] D. Nie, Q. Ma, L. Ma, and S. Xiao, "Optimization based grayscale image colorization," Pattern Recognit. Lett., vol. 28, no. 12, pp. 1445–1451, 2007, doi: 10.1016/j.patrec.2007.02.018.

[3] S.-Y. Chen, J.-Q. Zhang, Y.-Y. Zhao, P. L. Rosin, Y.-K. Lai, and L. Gao, "A review of image and video colorization: From analogies to deep learning," Vis. Informatics, vol. 6, no. 3, pp. 51–68, 2022, doi: 10.1016/j.visinf.2022.05.003.

[4] H. Liu, M. Xie, J. Xing, C. Li, and T.-T. Wong, "Video colorization with pretrained text-to-image diffusion models," arXiv preprint arXiv:2306.01732, 2023, doi: 10.48550/arXiv.2306.01732.

[5] C. Rota, M. Buzzelli, S. Bianco, and R. Schettini, "Video restoration based on deep learning: A comprehensive survey," Artif. Intell. Rev., vol. 56, no. 6, pp. 5317–5364, 2023, doi: 10.1007/s10462-022-10302-5.

[6] Y. Yang, J. Pan, Z. Peng, X. Du, Z. Tao, and J. Tang, "Bistnet: Semantic image prior guided bidirectional temporal feature fusion for deep exemplar-based video colorization," IEEE Trans. Pattern Anal. Mach. Intell., early access, 2024, doi: 10.1109/TPAMI.2024.3370920.

[7] S. Chen et al., "Exemplar-based video colorization with long-term spatiotemporal dependency," Knowl.-Based Syst., vol. 284, p. 111240, 2024, doi: 10.1016/j.knosys.2023.111240.

[8] Z. Huang, M. Zhang, and J. Liao, "LVCD: Reference-based lineart video colorization with diffusion models," arXiv preprint arXiv:2409.12960, 2024, doi: 10.1145/3687910.

[9] Y. Yang, J. Jin, Y. Huang, K. Guo, and X. Xu, "Reference-based video colorization with AB chrominance point and temporal propagation," in Proc. 16th Int. Conf. Mach. Learn. Comput. (ICMLC), 2024, pp. 340–346, doi: 10.1145/3651671.3651767.

[10] D. A. Karjadi, B. Y. Wedha, and H. Santoso, "Heavy-loaded vehicles detection model testing using synthetic dataset," SinkrOn, vol. 7, no. 2, pp. 464–471, 2022, doi: 10.33395/sinkron.v7i2.11378.

[11] M. G. K. S., M. P. S. T., and M. H. Vasanth, "Skin lesion segmentation and classification using deep learning," Int. J. Res. Appl. Sci. Eng. Technol., vol. 11, no. 4, pp. 312–320, 2023, doi: 10.22214/ijraset.2023.50055.

[12] J. Yao, Y. Li, B. Yang, and C. Wang, "Learning global image representation with generalized-mean pooling and smoothed average precision for large-scale CBIR," IET Image Process., vol. 17, no. 9, pp. 2748–2763, 2023, doi: 10.1049/ipr2.12825.

[13] A. K. Dubey and V. Jain, "Comparative study of convolution neural network's ReLU and leaky-ReLU activation functions," in Applications of Computing, Automation and Wireless Systems in Electrical Engineering: Proc. MARC 2018. Springer, 2019, pp. 873–880, doi: 10.1007/978-981-13-6772-4_76.

[14] C. B. Maniyar and M. Kumar, "Deep learning-based improved automatic building extraction from open-source high resolution unmanned aerial vehicle (UAV) imagery," Lect. Notes Civ. Eng., vol. 304, pp. 51–66, 2023, doi: 10.1007/978-3-031-19309-5_5.

[15] C. Ma, X. Mu, and D. Sha, "Multilayers feature fusion of convolutional neural network for scene classification of remote sensing," IEEE Access, vol. 7, pp. 121685–121694, 2019, doi: 10.1109/ACCESS.2019.2936215.

[16] A. Radford et al., "Learning transferable visual models from natural language supervision," in Proc. Int. Conf. Mach. Learn. (ICML), PMLR, 2021, pp. 8748–8763, doi: 10.48550/arXiv.2103.00020.

[17] H. Tan and M. Bansal, "LXMERT: Learning cross-modality encoder representations from transformers," arXiv preprint arXiv:1908.07490, 2019, doi: 10.48550/arXiv.1908.07490.

[18] T. Gherbi, A. Zeggari, Z. Ahmed Seghir, and F. Hachouf, "An evaluation metric for image retrieval systems, using entropy for grouped precision of relevant retrievals," J. Intell. Fuzzy Syst., vol. 45, no. 3, pp. 3665–3677, 2023, doi: 10.3233/JIFS-223623.

[19] F. K. Mihna et al., "Bridging law and machine learning: A cybersecure model for classifying digital real estate contracts in the metaverse," Mesopotamian J. Big Data, pp. 35–49, 2025, doi: 10.58496/MJBD/2025/003.

[20] P. Christen, D. J. Hand, and N. Kirielle, "A review of the F-measure: Its history, properties, criticism, and alternatives," ACM Comput. Surv., vol. 56, no. 3, 2023, doi: 10.1145/3606367.

[21] N. Shanavas and S. Asokan, "Ontology-based document mining system for IT support service," Procedia Comput. Sci., vol. 46, pp. 329–336, 2015, doi: 10.1016/j.procs.2015.02.028.

[22] M. A. Habeeb, Y. L. Khaleel, R. D. Ismail, Z. T. Al-Qaysi, and A. F. N., "Deep learning approaches for gender classification from facial images," Mesopotamian J. Big Data, vol. 2024, pp. 185–198, 2024, doi: 10.58496/MJBD/2024/013.

[23] C. Shen, S. Yu, J. Wang, G. Q. Huang, and L. Wang, "A comprehensive survey on deep gait recognition: Algorithms, datasets, and challenges," IEEE Trans. Biometrics, Behav. Identity Sci., early access, 2024, doi: 10.1109/TBIOM.2024.3486345.

[24] W. Alexan, M. Gabr, E. Mamdouh, R. Elias, and A. Aboshousha, "Color image cryptosystem based on sine chaotic map, 4D Chen hyperchaotic map of fractional-order and hybrid DNA coding," IEEE Access, vol. 11, pp. 54928–54956, 2023, doi: 10.1109/ACCESS.2023.3282160.

[25] E. K. Hendarwati, P. Lepong, and S. Suyitno, "Pemilihan semivariogram terbaik berdasarkan root mean square error (RMSE) pada data spasial eksplorasi emas awak mas," GEOSAINS KUTAI BASIN, vol. 6, no. 1, pp. 47–52, 2023, doi: 10.30872/geofisunmul.v6i1.1072.

[26] E. Saleem and N. K. El Abbadi, "Auto colorization of greyscale image using YCbCr color space," Iraqi J. Sci., pp. 3379–3386, 2020, doi: 10.24996/ijs.2020.61.12.26.

[27] N. K. El Abbadi and E. Saleem, "Gray image colorization based on general singular value decomposition and YCbCr color space," Kuwait J. Sci., vol. 46, no. 4, 2019.

[28] F. Memon, M. A. Unar, and S. Memon, "Image quality assessment for performance evaluation of focus measure operators," Mehran Univ. Res. J. Eng. Technol., vol. 34, no. 4, pp. 379–386, 2015, doi: 10.48550/arXiv.1604.00546.

[29] R. Kankale, S. Paraskar, and S. Jadhao, "Classification of power quality disturbances in emerging power system with distributed generation using space phasor model and normalized cross correlation," in Proc. 8th Int. Conf. Smart Comput. Commun. (ICSCC), IEEE, 2021, pp. 340–345, doi: 10.1109/ICSCC51209.2021.9528195.

[30] I. Bakurov, M. Buzzelli, R. Schettini, M. Castelli, and L. Vanneschi, "Structural similarity index (SSIM) revisited: A data-driven approach," Expert Syst. Appl., vol. 189, p. 116087, 2022, doi: 10.1016/j.eswa.2021.116087.

[31] E. Saccenti, M. H. W. B. Hendriks, and A. K. Smilde, "Corruption of the Pearson correlation coefficient by measurement error and its estimation, bias, and correction under different error models," Sci. Rep., vol. 10, no. 1, p. 438, 2020, doi: 10.1038/s41598-019-57247-4.

[32] B. Xie, S. Yang, Y. Li, Y. Xu, and G. Wang, "Color transfer based on cartoon decomposition and efficient illuminant estimation," in Proc. Int. Conf. Signal Process. Commun. Technol. (SPCT), SPIE, 2023, p. 1261502, doi: 10.1117/12.2673791.

[33] Z. Sheng, H.-L. Shen, B. Yao, and H. Zhang, "Guided colorization using mono-color image pairs," IEEE Trans. Image Process., vol. 32, pp. 905–920, 2023, doi: 10.1109/TIP.2023.3246592.

🔍 Note: The DOI 10.48550/arXiv.2108.07471 points to an arXiv preprint, but the paper was published in IEEE. Updated to correct publication DOI where possible. If you prefer the arXiv version, use:
doi: 10.48550/arXiv.2108.07471

[34] Z. Ke, Y. Liu, L. Zhu, N. Zhao, and R. W. H. Lau, "Neural preset for color style transfer," in Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR), 2023, pp. 14173–14182, doi: 10.48550/arXiv.2303.13511.

[35] U. Vijetha and V. Geetha, "Superpixel based image colorization with automated reference image selection," in Proc. IEEE Int. Students' Conf. Electr., Electron. Comput. Sci. (SCEECS), 2023, pp. 1–6, doi: 10.1109/SCEECS57921.2023.10061822.

[36] X. Zhu and E. Hu, "A local color transfer method based on optimal transmission," in Proc. 5th Int. Conf. Artif. Intell. Comput. Sci. (AICS), SPIE, 2023, pp. 483–492, doi: 10.1117/12.3009533.

[37] C. Lv, D. Zhang, S. Geng, Z. Wu, and H. Huang, "Color transfer for images: A survey," ACM Trans. Multimed. Comput. Commun. Appl., vol. 20, no. 8, pp. 1–29, 2024, doi: 10.1145/3635152.

[38] C. Lv and D. Zhang, "Palette-based color transfer between images," arXiv preprint arXiv:2405.08263, 2024, doi: 10.48550/arXiv.2405.08263.

[39] S. Bao, Y. Zhao, Y. Ji, N. Wu, and G. Le, "Color transfer method based on saliency features for color images," Opt. Rev., pp. 1–14, 2024, doi: 10.1007/s10043-024-00888-2.

[40] D. T. Tran et al., "Vitexco: Exemplar-based video colorization using vision transformer," in Proc. 14th Int. Conf. Inf. Commun. Technol. Convergence (ICTC), Oct. 2023, doi: 10.1109/ICTC58733.2023.10393505.