A Web-Based Streaming Video Identification Method Using Self-Supervised Structural Embedding and Vector Similarity Search

Injae Yoo[1], Byeongchan Park[1], Sun-Jib Kim[2], Seok-Yoon Kim[1] and Youngmo Kim[1,*]

[1]*Department of Computer Science & Engineering, Soongsil University, Korea*
[2]*Department of Convergence Security, Hansei University, Korea*
E-mail: halo8024@outlook.com; pbc866@ssu.ac.kr; kimsj@hansei.ac.kr; ksy@ssu.ac.kr; ymkim828@ssu.ac.kr
[*]*Corresponding Author*

Received 10 March 2026; Accepted 30 May 2026

Abstract

Streaming video services have become a major channel for content distribution owing to the rapid growth of over-the-top (OTT) platforms and web-based media services. As a result, accurate video identification is increasingly required for copyright protection, audience measurement, content management, and search and recommendation services. However, conventional feature point-based methods often show limited robustness against transformations frequently observed in streaming environments, including compression, resolution change, re-encoding, frame-rate reduction, aspect-ratio conversion, overlays, rotation, and flipping. They also require the storage and comparison of many local descriptors, which can limit retrieval efficiency in large-scale content databases. In this paper, we propose a web-based streaming video identification method using self-supervised structural embedding and vector similarity search. The proposed method extracts representative

Journal of Web Engineering, Vol. 25_6, 1193–1214.
doi: 10.13052/jwe1540-9589.2567
© 2026 River Publishers

frames from original and query videos, applies preprocessing, generates 512-dimensional structural embeddings using a self-supervised copy detection model, and performs cosine similarity search in a vector database. To improve robustness against geometric transformations, transformed reference frames are additionally registered. For query videos, multiple representative frames are used, and the final video-level identification result is determined by majority voting over frame-level retrieval results. Each frame is represented by a fixed-length 512-dimensional vector, requiring 2048 bytes per frame and 10,240 bytes for a five-frame query. Experiments on 4000 query videos covering 14 transformation categories and 40 detailed transformation settings show that the proposed method achieves a recognition rate of 99.23%, a missed recognition rate of 0.38%, and a false recognition rate of 0.40%. These results demonstrate the feasibility of the proposed method for identifying transformed copies of the same source video in web-based streaming environments.

Keywords: Streaming video identification, self-supervised learning, structural embedding, vector similarity search, vector database.

1 Introduction

With the advancement of internet infrastructure and streaming technologies, video-based content consumption has rapidly increased and has become one of the dominant forms of media distribution [1]. In particular, OTT platforms and web-based video services such as Netflix, YouTube, and other streaming media platforms have changed how users access, consume, and share video content. According to Sandvine's Global Internet Phenomena Report, video streaming continues to account for a substantial portion of global internet traffic [2]. Nielsen's The Gauge also reported that streaming accounted for 40.3% of total TV usage in June 2024 [3]. These trends indicate that streaming media has become a central component of the modern web-based content distribution environment.

As streaming services continue to expand, accurate video identification has become increasingly important for copyright protection, audience measurement, user behavior analysis, content management, and search and recommendation services. In the broadcasting and media industries, identifying what content is being watched, when it is watched, through which devices, and for how long is essential for content scheduling, advertising optimization, viewing-pattern analysis, and service planning [4, 5]. In addition, traditional

audience measurement approaches have limitations in fully reflecting OTT and cross-platform viewing behaviors, which increases the need for automated content identification in web-based streaming environments [5, 6]. Automatic Content Recognition (ACR) technologies have also been adopted in smart TVs and streaming devices to identify displayed content and analyze viewing history, further emphasizing the importance of reliable video identification [7].

However, video identification in streaming environments is challenging because the same source content can appear in many different forms. During web-based distribution or redistribution, video content may undergo compression, re-encoding, resolution change, frame-rate reduction, aspect-ratio conversion, brightness or contrast adjustment, grayscale conversion, logo insertion, text overlay, rotation, or flipping. These transformations can change the visual appearance of frames while preserving the underlying content identity. Therefore, a practical streaming video identification method should be robust to such transformations while maintaining efficient storage and retrieval performance.

Conventional feature point-based methods such as SIFT, SURF, ORB, and AKAZE have been widely used for image matching, video retrieval, and copy detection tasks [8–11]. These methods detect local keypoints from video frames and compare local descriptors to determine visual similarity. Although they provide robustness against certain geometric and photometric transformations, they often generate a large number of descriptors per frame. As a result, storage cost and matching complexity can increase when they are applied to large video collections. In addition, their performance may degrade under streaming-specific transformations such as strong compression, resolution changes, frame loss, padding, overlays, and repeated re-encoding.

Recently, self-supervised visual representation learning has shown strong potential for robust content identification. In particular, Self-Supervised Copy Detection (SSCD) learns feature representations that remain relatively stable under various image transformations and has demonstrated effective performance in copy detection and large-scale image retrieval tasks [12]. Unlike local feature descriptors, SSCD-based embeddings represent each frame as a fixed-length vector, which is suitable for vector similarity search. At the same time, vector database technologies and approximate nearest neighbor search algorithms enable efficient retrieval of high-dimensional embeddings in large datasets [13, 14]. These developments provide a practical basis for applying structural visual embeddings to streaming video identification.

In this study, we propose a web-based streaming video identification method using self-supervised structural embedding and vector similarity search. The proposed method extracts representative frames from original and query videos, applies preprocessing, generates 512-dimensional structural embeddings, and performs cosine similarity search using a vector database. In the reference registration stage, transformed versions of reference frames are additionally registered to improve robustness against geometric transformations such as rotation and flipping. In the query identification stage, multiple representative frames are sampled from a suspected streaming video, and the final video-level decision is made by aggregating frame-level retrieval results through majority voting.

It should be noted that this study does not aim to develop a new self-supervised embedding backbone. Instead, the main focus is on adapting self-supervised structural embeddings to web-based streaming video identification through robustness-oriented reference registration, fixed-length vector retrieval, and multi-frame video-level decision. In this paper, identical content refers to video content derived from the same source video through transformations such as re-encoding, compression, resolution change, frame-rate reduction, aspect-ratio conversion, overlay insertion, rotation, flipping, grayscale conversion, and brightness or contrast adjustment. Heavily edited videos involving scene insertion, scene deletion, long partial clips, or semantic-level similarity between different videos are outside the current experimental scope and remain future research topics.

The main contributions of this paper are summarized as follows.

First, we present a web-based streaming video identification method that represents video frames as fixed-length 512-dimensional structural embeddings and performs content matching through vector similarity search.

Second, we introduce a robustness-oriented reference registration strategy in which transformed frames are additionally stored to improve identification performance against geometric transformations commonly observed in redistributed streaming videos.

Third, we apply a multi-frame video-level decision procedure that aggregates frame-level retrieval results, reducing the influence of temporary frame-level retrieval failures caused by compression artifacts, overlays, padding, or low-texture scenes.

Fourth, we evaluate the proposed method using 4000 query videos covering 14 transformation categories and 40 detailed transformation settings. The

experimental results show a recognition rate of 99.23%, a missed recognition rate of 0.38%, and a false recognition rate of 0.40%, demonstrating the feasibility of the proposed method for identifying transformed copies of the same source video under the evaluated streaming conditions.

The remainder of this paper is organized as follows. Section 2 reviews related studies on video fingerprinting, copy detection, feature-based video identification, self-supervised visual representation learning, and vector similarity search. Section 3 describes the proposed web-based streaming video identification method. Section 4 presents the experimental environment, performance evaluation, comparison results, and component-level analysis. Finally, Section 5 concludes the paper and discusses limitations and future research directions.

2 Related Work

2.1 Feature-Based Video Content Identification Technology

Feature-based video content identification is a technique that extracts feature information from video frames and determines whether videos are identical or similar by comparing the similarity between contents based on feature points. The overall process is illustrated in Figure 1.

In conventional video identification research, methods that detect feature points from video frames and represent them in the form of feature descriptors have been widely used. These approaches provide a certain level of robustness against transformations such as rotation, scale variation, and

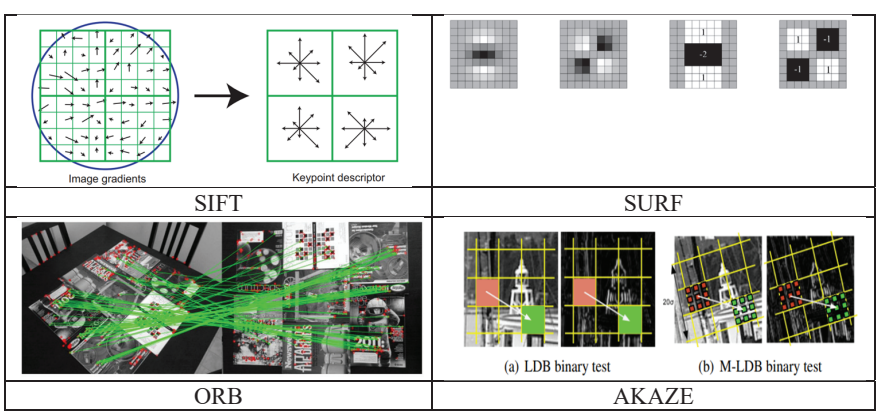

Figure 1 Existing image feature extraction methods.

illumination changes, and have been applied in image retrieval and copy detection tasks.

Representative feature extraction methods include SIFT, SURF, ORB, and AKAZE, as shown in Figure 1 [8–11]. These methods first detect keypoints in video frames and then represent the visual characteristics at each location as vectors. The similarity between these vectors is subsequently calculated to determine whether the contents are identical. In particular, ORB and AKAZE enable feature extraction with relatively low computational cost, making them suitable for real-time processing environments.

However, feature point-based methods generate a large number of keypoints and descriptors for each frame, which increases both storage requirements and computational cost. Moreover, recognition performance may degrade due to transformations frequently occurring in streaming environments, such as compression, resolution changes, frame loss, and screen editing. Therefore, more robust feature extraction methods are required for large-scale streaming environments.

2.2 Self-Supervised Video Feature Representation Learning

Recently, in the fields of video and image recognition, self-supervised visual representation learning techniques have been actively studied to learn effective feature representations in large-scale data environments, as illustrated in Figure 2.

Self-supervised learning is a method that learns features by utilizing the structural relationships within the data itself without relying on explicit

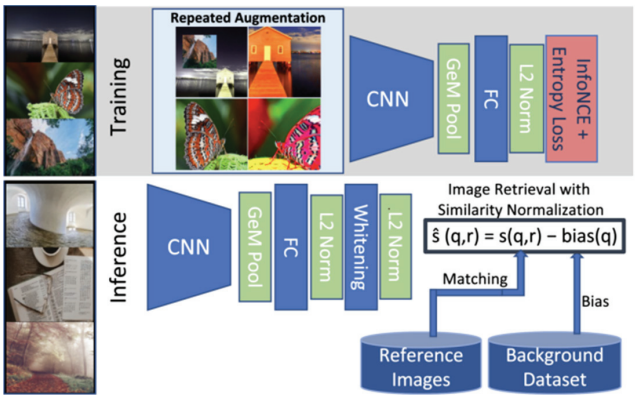

Figure 2 SSCD (Self-Supervised Copy Detection).

ground-truth labels. This approach has demonstrated strong performance across various visual recognition tasks. In particular, contrastive learning-based self-supervised representation learning defines transformed samples generated from the same video or image as positive pairs, while samples generated from different contents are defined as negative pairs, and learns the distance relationships in the embedding space accordingly. During this process, feature vectors generated from the same content are trained to be positioned close to each other, while feature vectors from different contents are placed farther apart. This approach has the advantage of providing relatively stable feature representations even under various visual transformations such as compression, cropping, color changes, and resolution variations.

In video content identification tasks, such self-supervised feature representations can be applied at the frame level. The embeddings generated from each frame reflect the structural visual information of the video, and the similarity between the original content and the query content can be computed based on these embeddings. In particular, recently proposed self-supervised copy detection models, as illustrated in Figure 2, generate embeddings that are robust to various visual transformations and have been effectively utilized for content identification and similarity retrieval tasks [12].

2.3 Vector Database-Based Similarity Search

In large-scale video data environments, similarity search techniques that can efficiently store extracted feature vectors and enable fast retrieval are essential. Traditional relational databases are suitable for structured data queries; however, they have limitations when handling high-dimensional vector-based searches. To address this issue, vector database-based retrieval technologies have recently been adopted, as illustrated in Figure 3 [13].

As shown in Figure 3, a vector database is a system designed to store high-dimensional feature vectors and efficiently perform similarity-based searches. In particular, by utilizing Approximate Nearest Neighbor (ANN)-based search algorithms, fast retrieval can be achieved even in large-scale vector datasets [14]. Vector database systems provide indexing techniques such as IVF (Inverted File Index) and HNSW (Hierarchical Navigable Small World), which support high-speed similarity searches over large volumes of embedding vectors. These architectures have been widely applied in various fields, including image retrieval, recommendation systems, and semantic search, and they can also be applied to video content identification tasks in a similar manner.

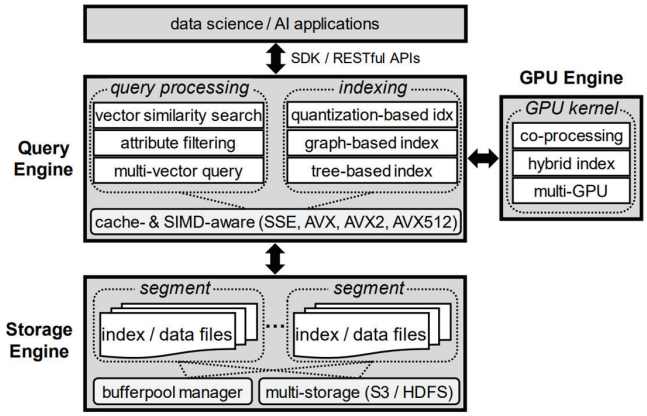

Figure 3 Vector database architecture.

In video identification systems, feature embeddings generated from original video frames are stored in a vector database, and similarity searches are performed with feature vectors generated from a query video to find the most similar content. This approach can provide high retrieval efficiency even when dealing with large-scale original content databases. Therefore, combining self-supervised feature embeddings with vector database-based retrieval can be considered an effective approach for large-scale streaming video identification problems.

3 Proposed Streaming Video Identification Method

3.1 Overview and Web-Based Data Flow

The overall procedure of the proposed web-based streaming video identification method using self-supervised structural embedding and vector similarity search is shown in Figure 4. The proposed method is designed to identify whether a query streaming video is a transformed copy derived from a registered original video. To this end, representative frames are extracted from videos, converted into fixed-length structural embeddings, and compared with reference embeddings stored in a vector database.

The proposed method is divided into two main procedures: reference content registration and query content identification. The reference content registration procedure is performed in advance to construct the reference vector database. First, I-frames are extracted from original videos using FFmpeg. The extracted frames are resized and preprocessed, and transformed versions

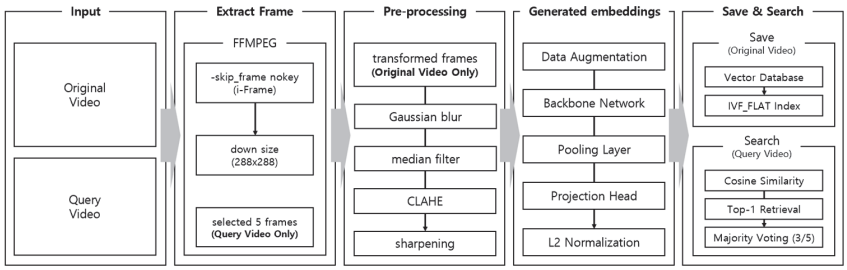

Figure 4 Web-based identification method for streaming videos using self-supervised structural embedding and vector similarity search.

of the frames are additionally generated to improve robustness against rotation and flipping. The original and transformed frames are then converted into 512-dimensional structural embeddings using a self-supervised copy detection model. Finally, the generated embeddings are stored in the vector database together with metadata such as video ID, frame index, and transformation type.

The query content identification procedure is performed when a suspected streaming video or query video is given. Representative frames are sampled from the query video according to temporal intervals. The sampled frames are resized, preprocessed, and converted into 512-dimensional embeddings using the same procedure as the reference frames. The generated query embeddings are used for cosine similarity search against the reference embeddings stored in the vector database. For each query frame, the Top-1 search result is obtained as a frame-level identification result.

The frame-level results are then aggregated to determine the final video-level identification result. In this study, multiple representative frames are used instead of a single frame, and the query video is identified as the corresponding reference video when a sufficient number of frames are matched to the same video ID. This multi-frame decision procedure improves identification stability by reducing the effect of temporary frame-level retrieval failures caused by compression artifacts, overlays, padding, low-texture scenes, or other distortions.

In a practical web-based environment, the query processing can be performed either on the client side or on the server side depending on service requirements. When client-side processing is used, only representative frame embeddings and metadata such as timestamps need to be transmitted to the server through standard HTTPS-based REST APIs. For near-real-time monitoring, WebSocket-based communication can also be used to transmit

sampled embeddings while the video is being played. Therefore, the proposed method does not require the entire video stream to be transmitted to the server for identification.

The proposed method focuses on identifying transformed copies derived from the same source video. In this study, the target transformations include compression, re-encoding, resolution change, frame-rate reduction, aspect-ratio conversion, logo or text overlay, grayscale conversion, brightness and contrast adjustment, rotation, and flipping. Heavily edited videos involving scene insertion, scene deletion, long partial clips, or semantic-level similarity between different videos are outside the current experimental scope.

3.2 Frame Extraction and Preprocessing

The reference content registration procedure corresponds to the left part of Figure 4. In this procedure, original videos are converted into frame-level structural embeddings and stored in the vector database as reference data for query identification. As shown in Figure 4, the procedure consists of I-frame extraction, transformed-frame generation, preprocessing, embedding generation, and vector database registration.

First, I-frames are extracted from each original video using FFmpeg with the -skip_frame nokey option. I-frames are used because they contain independently decodable visual information and provide stable visual structures for reference registration. The extracted frames are resized to 288 $\times$ 288 before preprocessing and embedding generation.

The set of I-frames extracted from a reference video is defined as shown in Equation (1).

$$F_r = \{f_1, f_2, \ldots, f_N\} \tag{1}$$

At this time, the sampling time $t_k$can be defined as shown in Equation (2).

$$t_k = t_s + \Delta\left(k - \frac{1}{2}\right), \quad k = 2, \ldots, 6 \tag{2}$$

In the preprocessing stage, noise suppression, local contrast enhancement, and edge enhancement are applied to the extracted frame . Specifically, in the original content registration stage, Gaussian blur, median filtering, contrast-limited adaptive histogram equalization (CLAHE), and sharpening are sequentially applied. The same preprocessing procedure is also performed in the query stage. If this process is represented by a preprocessing function $P(\cdot)$, it can be defined as shown in Equation (3).

$$\tilde{I}_i = P(I_i) \tag{3}$$

The preprocessed frames are subsequently used in the embedding generation stage, and the input channel normalization is performed as defined in Equation (4).

$$X_i(c,h,w) = \frac{\frac{\tilde{I}_i(c,h,w)}{255} - \mu_c}{\sigma_c} \tag{4}$$

Here, $\mu = [0.485, 0.456, 0.406]$ and $\sigma = [0.229, 0.224, 0.225]$ denote the channel-wise mean and standard deviation, respectively. The same normalization parameters are also used in the actual experiments.

In the query stage, frames are first normalized to a resolution of 288×288, followed by preprocessing steps including Gaussian blur, median filtering, CLAHE, and sharpening. In the original content registration stage, rotated and flipped transformed frames are additionally generated before applying the same preprocessing pipeline.

3.3 Feature Embedding Generation

The preprocessed frames are input into an SSCD-based embedding model, where they are transformed into high-dimensional structural feature vectors. In this study, an SSCD model in TorchScript format is used, and the input tensor is fed into the model with a batch dimension included. The feature embedding f_i for frame i can be defined as shown in Equation (5).

$$f_i = E(X_i) \tag{5}$$

Here, $E(\cdot)$ denotes the SSCD-based embedding function, and $f_i \in \mathbb{R}^{512}$ represents a 512-dimensional feature vector. In the query stage implementation, multiple frames are processed in batch units, and a 512-dimensional embedding is generated for each frame. The generated vectors are L2-normalized to improve the stability of cosine-based similarity search. The normalized vector $\hat{f}_i$ can be defined as shown in Equation (6).

$$F_i = \{E(P(v(I_i))) \mid v \in V\} \tag{6}$$

Here, $V = \{$“original”,“rotate90”,“rotate270”,“flip_ud”,“flip_lr”$\}$. The generated embeddings are stored sequentially in the same order.

3.4 Vector Database-Based Video Retrieval

The generated feature embeddings are stored in a vector database and used for nearest neighbor searches with query vectors. In this study, a collection

containing the video ID and a 512-dimensional embedding vector is created, where the embedding vector is stored as a FLOAT_VECTOR type. The search index uses IVF_FLAT, and the similarity metric is set to COSINE. In addition, nprobe $= 10$ is applied during the search process to control the exploration range of candidate partitions. The similarity between the query frame vector $\hat{q}_i$ and the reference frame vector $\hat{r}_j$ in the database is defined using cosine similarity, as shown in Equation (7).

$$Sim(\hat{q}_i, \hat{r}_j) = \frac{\hat{q}_i \cdot \hat{r}_j}{\|\hat{q}_i\| \|\hat{r}_j\|} \tag{7}$$

For each query frame, only the highest similarity score among all candidate frames in the database is used. Therefore, the representative matching score $s_i$for query frame $i$can be defined as shown in Equation (8).

$$s_i = max_j \; Sim(q_i, r_j) \tag{8}$$

In this study, Top-1 retrieval is performed using the query vector as input, returning the most similar video ID and its corresponding similarity score, which are used as the frame-level search results. The video-level aggregation score between the query video Q and the reference video R is defined as the average of the highest similarity scores for each query frame, as shown in Equation (9).

$$S(Q, R) = \frac{1}{N} \sum_{i=1}^{N} s_i \tag{9}$$

Here, N denotes the number of representative frames extracted from the query video and, in this study, $N = 5$ is used.

3.5 Video Identification Method

In web-based streaming environments, determining content similarity based on a single frame may lead to misrecognition due to various transformations. To reduce this issue, this study applies a multi-frame-based decision approach. As defined in Equation (10), if the highest similarity score s_i of a frame exceeds a predefined threshold τ, the frame is determined to be a matched frame.

$$m_i = \begin{cases} 1, & s_i \geq \tau \\ 0, & s_i < \tau \end{cases} \tag{10}$$

After that, if the number of matched frames among the total N representative frames is greater than or equal to the threshold value K, the query video and the reference video are determined to be the same content.

$$D(Q,R) = \begin{cases} 1, & \sum_{i-1}^{N} m_i \geq K \\ 0, & otherwise \end{cases} \tag{11}$$

In Equation (11), $D(Q,R) = 1$ indicates that the query video Q is identified as the same content as the reference video R. In this study, the number of representative frames is set to five, and a majority voting scheme is applied in which the content is finally determined to be identical if $K = 3$ or more frames are matched. This multi-frame-based decision strategy reduces the impact of search failures in individual frames on the overall video-level identification result. It also helps ensure the real-time performance and robustness required in web-based streaming environments.

4 Experiments and Results

4.1 Experimental Environment

To evaluate the performance of the proposed streaming video identification method, an experimental environment was constructed consisting of original content registration, query video processing, and vector database retrieval. In the original content registration stage, I-frames were extracted from the input videos, and SSCD-based feature embeddings were generated after preprocessing and then stored in a vector database. In this process, the FFmpeg-skip_frame nokey option was used to extract only I-frames, and the extracted frames were normalized to a resolution of 288 $\times$ 288. Subsequently, preprocessing steps including Gaussian blur, median filtering, CLAHE, and sharpening were applied, and 512-dimensional feature vectors were generated using the SSCD TorchScript model. In addition, during the original content registration stage, not only the original frames but also transformed frames including 90° rotation, 270° rotation, vertical flipping, and horizontal flipping were generated to improve the robustness of the original feature representations.

In the query video processing stage, up to five frames were sampled based on temporal intervals to ensure representative coverage of the entire query video. Each frame was normalized to a resolution of 288 $\times$ 288, followed

by the same preprocessing procedure, and feature embedding vectors were generated in batch units.

4.2 Dataset and Transformation Conditions

The experiments were conducted using original content and query content with various transformations applied. A total of 100 original videos were used to construct the reference vector database, and the frame feature embedding vectors extracted from each original video were stored in the vector database. The query videos consisted of transformed versions generated from the original videos. In total, 4000 test videos were used, including the original videos and 14 different types of transformations. The transformation conditions were designed to reflect the types of variations that may occur in web-based streaming environments, and the dataset configuration is presented in Table 1.

4.3 Evaluation Metrics

To evaluate the performance of the proposed method, recognition rate, false recognition rate, and missed recognition rate were measured. The recognition rate represents the proportion of query videos that were correctly identified as their corresponding original content among all query videos, as defined in Equation (12).

$$Recognition\ Rate = \frac{N_{correct}}{N_{total}} \times 100 \tag{12}$$

Here, N_{correct} denotes the number of correctly identified query videos, and N_{total} represents the total number of query videos. The false recognition rate represents the proportion of query videos that were incorrectly identified as a different original content, as defined in Equation (13).

$$Miss\ Recognition\ Rate = \frac{N_{miss}}{N_{total}} \times 100 \tag{13}$$

Here, N_{miss} denotes the number of query videos that were matched with an incorrect original ID. The missed recognition rate represents the proportion of query videos that were not identified as any original content, as defined in Equation (14).

$$False\ Recognition\ Rate = \frac{N_{false}}{N_{total}} \times 100 \tag{14}$$

In addition, to evaluate storage efficiency, the vector size per frame was calculated. In this study, the SSCD feature vector has 512 dimensions, and

Table 1 Dataset configuration and transformation conditions

Robustness Type	Robustness Details	Video Count
Original	Original	100
Logo insertion	Opacity 100%	100
Text insertion	Font size 12	100
	Font size 16	100
	Font size 20	100
Video compression	DivX 512 Kbps	100
	DivX 700 Kbps	100
Codec conversion	MPEG-4/AVC	100
	WMV	100
	Xvid	100
	HEVC/H.265	100
Aspect ratio conversion	16:9 → 4:3 (top and bottom black bars)	100
	4:3 → 16:9 (left and right black bars)	100
Frame rate reduction	20 fps	100
Rotation	Rotate 90°	100
	Rotate 180°	100
	Rotate 270° (−90°)	100
Flip	Horizontal flip	100
	Vertical flip	100
Grayscale conversion	I = 0.299xR + 0.587xG + 0.114xB	100
Brightness adjustment	9%	100
	−18%	100
	−9%	100
	18%	100
Contrast adjustment	Contrast (137%)	100
	Contrast (62%)	100
Combined transformation	DivX 512 Kbps/H.264/Font size 16	100
	DivX 512 Kbps/Xvid/Font size 16	100
	DivX 512 Kbps/WMV/Font size 16	100
	DivX 700 Kbps/H.264/Font size 16	100
	DivX 700 Kbps/Xvid/Font size 16	100
	DivX 700 Kbps/WMV/Font size 16	100
	DivX 512 Kbps/H.265/Font size 16	100
	DivX 700 Kbps/H.265/Font size 16	100
Resolution change	320 × 240	100
	640 × 480	100
	800 × 480	100
	1,920 × 1,080 (FHD) → 640 × 480 (VGA)	100
	1,920 × 1,080 (FHD) → 1,280 × 720 (HD)	100
	1,920 × 1,080 (FHD) → 3,840 × 2,160 (UHD)	100
	Total	4000

Table 2 Experimental results

Robustness Type	Total	Recog.	Recog. Rate	Miss Recog.	Miss Recog. Rate	False Recog.	False Recog. Rate
Original	100	100	100.00%	0	0.00%	0	0.00%
Logo insertion	100	99	99.00%	0	0.00%	1	1.00%
Text insertion	300	297	99.00%	1	0.33%	2	0.67%
Video compression	200	199	99.50%	1	0.50%	0	0.00%
Codec conversion	400	399	99.75%	1	0.25%	0	0.00%
Aspect ratio conversion	200	195	97.50%	2	1.00%	3	1.50%
Frame rate reduction	100	99	99.00%	1	1.00%	0	0.00%
Rotation	300	298	99.33%	0	0.00%	2	0.67%
Flip	200	200	100.00%	0	0.00%	0	0.00%
Grayscale conversion	100	98	98.00%	0	0.00%	2	2.00%
Brightness adjustment	400	397	99.25%	1	0.25%	2	0.50%
Contrast adjustment	200	200	100.00%	0	0.00%	0	0.00%
Combined transformation	800	799	99.88%	0	0.00%	1	0.13%
Resolution change	600	589	98.17%	8	1.33%	3	0.50%
Total	4000	**3,969**	**99.23%**	**15**	**0.38%**	**16**	**0.40%**

each dimension uses 4 bytes in the FLOAT_VECTOR format. Therefore, the vector size per frame can be defined as shown in Equation (15).

$$V_{frame} = 512 \times 4 = 2048\ bytes \tag{15}$$

When five representative frames are used for each query video, the vector size per query can be defined as shown in Equation (16).

$$V_{query} = 5 \times 2048 = 10,240\ bytes \tag{16}$$

4.4 Experimental Results

Experiments were conducted on a total of 4000 query videos using the proposed method. The results showed a recognition rate of 99.23%, indicating that most query videos were correctly identified as their corresponding original content. In addition, the false recognition rate was measured at 0.38%, while the missed recognition rate was 0.40%. These results demonstrate that most query videos were correctly identified even under various video transformation conditions. Table 2 presents the experimental results for each transformation type.

From the perspective of vector size, the feature embeddings used in this study have a relatively fixed size of 2048 bytes per frame and 10,240 bytes per query video. Unlike conventional methods that require storing hundreds to thousands of feature points per frame, the proposed approach maintains a simpler storage and retrieval structure, making it well suited for large-scale vector database-based search environments.

5 Conclusion

In this paper, we proposed a web-based streaming video identification method using self-supervised structural embedding and vector similarity search. The proposed method extracts frames from both original and query videos, generates structural feature embeddings through a preprocessing process, and determines content similarity by storing and retrieving these embeddings in a vector database. In addition, a multi-frame-based decision method is applied to query videos to reduce misrecognition caused by errors in individual frames and to enable stable identification at the video level.

Experiments were conducted using a total of 4000 query videos, including the original videos and 14 types of transformations. The results showed that the proposed method achieved a recognition rate of 99.23%, a false recognition rate of 0.38%, and a missed recognition rate of 0.40%, demonstrating strong robustness under various video transformation conditions. Furthermore, by using a fixed-length 512-dimensional feature vector, the storage size per frame could be maintained at a constant level, which is advantageous for storage and retrieval efficiency in large-scale content environments.

These results indicate that the proposed method can simultaneously provide high accuracy and practical processing efficiency in web-based streaming monitoring environments. Through this study, it was experimentally verified that combining self-supervised feature representation learning with a vector database-based retrieval structure enables stable identification even in real-world streaming environments where various transformations exist.

For future work, it will be necessary to enhance the sampling strategy for selecting representative frames from query videos. Currently, a fixed number of representative frames is used; however, applying an adaptive frame selection strategy that reflects scene changes or the amount of visual information could further reduce misrecognition results and improve identification stability.

Acknowledgment

This work was supported by the Software Copyright Research and Development Program funded by the Ministry of Culture, Sports and Tourism and managed by the Korea Institute of Culture Technology Evaluation and Planning (KCTEP) (Project Name: Development of Copyright Technology for OTT Contents Copyright Protection Technology Development and Application, Project Number: RS-2023-00225267, Contribution Rate: 100%).

References

[1] I. Yoo, Y. Kim, B. Park, S. Jang, and S. Kim, "A web-based identification method for illegal streaming videos using low-frequency components of the fast Fourier transform," *Journal of Korea Multimedia Society*, 2024.

[2] Sandvine, *Global Internet Phenomena Report 2024*. Sandvine Inc., 2024. [Online]. Available: https://www.sandvine.com/global-internet-phenomena-report.

[3] Nielsen, "Time spent streaming surges to 40.3% of total TV usage in June 2024," *The Gauge*, 2024.

[4] Nielsen, *Streaming Measurement*. Nielsen, 2024.

[5] Nielsen, "What's the difference between OTT, CTV and streaming?" Nielsen Insights, 2024.

[6] A. González-Neira and C. Quintas-Froufe, "Television audience measurement: The challenge posed by the rise of streaming and OTT platforms," *Comunicación y Sociedad*, 2020.

[7] G. Anselmi et al., "A first look at automatic content recognition tracking in smart TVs," *arXiv preprint arXiv:2409.06203*, 2024.

[8] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, 2004.

[9] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," *Computer Vision and Image Understanding*, vol. 110, no. 3, pp. 346–359, 2008.

[10] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "ORB: An efficient alternative to SIFT or SURF," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, 2011.

[11] P. F. Alcantarilla, J. Nuevo, and A. Bartoli, "Fast explicit diffusion for accelerated features in nonlinear scale spaces," in *Proc. British Machine Vision Conf. (BMVC)*, 2013.
[12] M. Douze, M. Furon, and H. Jégou, "Self-supervised copy detection for large-scale image retrieval," in *Proc. IEEE/CVF Int. Conf. Computer Vision (ICCV)*, 2021.
[13] J. Wang et al., "Milvus: A purpose-built vector data management system," in *Proc. ACM SIGMOD Int. Conf. Management of Data*, 2021.
[14] Y. Aumüller, E. Bernhardsson, and A. Faithfull, "ANN-benchmarks: A benchmarking tool for approximate nearest neighbor algorithms," *Information Systems*, vol. 87, 2020.

Biographies

Injae Yoo received his bachelor's degree in software engineering from The Cyber University of Korea in 2017, his master's degree in computer science and engineering from Soongsil University in 2022, and is currently pursuing a Ph.D. in computer science and engineering at Soongsil University since 2023. His research interests include lightweight video analysis, illegal streaming detection, and real-time web-based identification systems.

Byeongchan Park received the bachelor's degree in computer engineering through the Academic Credit Bank System, Korea, in 2015, and the master's and doctor of philosophy degrees in computer science and engineering from Soongsil University in 2018 and 2023, respectively. He is currently working as a Visiting Professor at the Department of Computer Science, Soongsil University. His research areas include copyright technology and the promotion of its utilization.

Sun-Jib Kim is currently working as a Professor at the School of IT, Hansei University, Korea. His research areas include information security, the Internet of Things (IoT), cloud computing, and AI system authentication.

Seok-Yoon Kim received the B.S. degree in electrical and electronic engineering from Seoul National University, Korea, in 1980, and the M.S. and Ph.D. degrees in electrical and computer engineering from the University of Texas at Austin, USA, in 1990 and 1993, respectively. From 1982 to 1987, he was a Researcher with the Electronics and Telecommunications Research Institute (ETRI). From 1993 to 1995, he worked as a Senior Researcher at Motorola. Since 1995, he has been a Professor at Soongsil University. His main research interests include copyright protection and the promotion of its utilization.

Youngmo Kim received the B.S., M.S., and Ph.D. degrees in computer engineering from Daejeon University, Korea, in 2003, 2005, and 2011, respectively. Since 2012, he has been a Professor at Soongsil University. His main research interests include copyright protection and the promotion of its utilization.