IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 2 August 2022, accepted 20 September 2022, date of publication 26 September 2022, date of current version 3 October 2022.

Digital Object Identifier 10.1109/ACCESS.2022.3209275

RESEARCH ARTICLE

LTC-SUM: Lightweight Client-Driven Personalized Video Summarization Framework Using 2D CNN

GHULAM MUJTABA[1,3], ADEEL MALIK[2], AND EUN-SEOK RYU[3], (Senior Member, IEEE)

[1]C-JeS Gulliver Studios, Seoul 10390, Republic of Korea
[2]Department of Communication System, EURECOM, 06904 Sophia-Antipolis, France
[3]Department of Computer Science Education, Sungkyunkwan University, Seoul 03063, Republic of Korea

Corresponding author: Eun-Seok Ryu (esryu@skku.edu)

This work was supported in part by the National Research Foundation of Korea (NRF) Grant funded by the Korean Government through MSIT under Grant 2022R1F1A1074935 and Grant NRF-2020R1A2C1013308, and in part by the Gachon University Research Fund of 2019 under Grant GCU-2019-0776.

ABSTRACT This paper proposes a novel lightweight thumbnail container-based summarization (LTC-SUM) framework for full feature-length videos. This framework generates a personalized keyshot summary for concurrent users by using the computational resource of the end-user device. State-of-the-art methods that acquire and process entire video data to generate video summaries are highly computationally intensive. In this regard, the proposed LTC-SUM method uses lightweight thumbnails to handle the complex process of detecting events. This significantly reduces computational complexity and improves communication and storage efficiency by resolving computational and privacy bottlenecks in resource-constrained end-user devices. These improvements were achieved by designing a lightweight 2D CNN model to extract features from thumbnails, which helped select and retrieve only a handful of specific segments. Extensive quantitative experiments on a set of full 18 feature-length videos (approximately 32.9 h in duration) showed that the proposed method is significantly computationally efficient than state-of-the-art methods on the same end-user device configurations. Joint qualitative assessments of the results of 56 participants showed that participants gave higher ratings to the summaries generated using the proposed method. To the best of our knowledge, this is the first attempt in designing a fully client-driven personalized keyshot video summarization framework using thumbnail containers for feature-length videos. Our code and trained models are publicly available at https://github.com/iamgmujtaba/LTC-SUM.

INDEX TERMS Client-driven, personalized media, video summarization, thumbnail containers, 2D CNN.

I. INTRODUCTION

In recent years, we have witnessed an exceptional growth in multimedia content, with a significant proportion of multimedia content comprised of videos. This growth trend is believed to continue in the future at even higher rates mainly because of two factors: (i) a steady increase in users' engagement with smart and computationally powerful video recording devices, and (ii) the widespread use of social media networks and video sharing platforms as a means of communication for billions of users [1]. This tremendous growth has increased the demand for technologies that enable users to quickly browse through vast and ever-growing videos and retrieve the content of their interest. The development of autonomous video summarizing techniques is one way to achieve these goals. These techniques produce a short version of a full-length video that conveys meaningful segments. Accordingly, viewers can quickly obtain an overview of the entire story without watching the full-length video. For instance, a 90-min video of a soccer match can be summarized in a few minutes, highlighting meaningful events such as free kicks and penalty shootouts.

Over the last few decades, several approaches have been proposed to automate video summarization. In general, these techniques fall into two categories: keyframes [2], [3], [4], [5] and keyshots [6], [7], [8], [9], [10], [11], [12]. Keyframes are

The associate editor coordinating the review of this manuscript and approving it for publication was Dian Tjondronegoro.

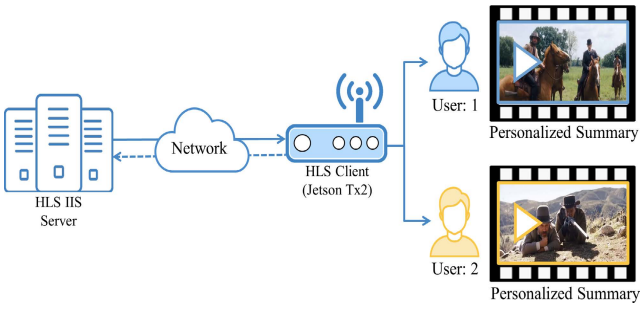


FIGURE 1. Conceptual diagram of proposed LTC-SUM framework using 2D CNN. It can generate distinct video summaries concurrently according to user preferences.

also known as static stories, representative frames, or static image summaries; in contrast, keyshots can be referred to as video skims, dynamic storyboards, or dynamic image summaries. The keyframe-based method selects a small number of image sequences from the original video, which presents an approximate visual representation. The keyshots consisted of typical continuous video segments of the full-length video that were shorter than the original video. The keyframe can be obtained from the keyshot summary in some cases [7]. In general, keyframe-based summaries are lighter in size than keyshot-based summaries. However, this gain is achieved at the cost of neglecting valuable information during the summarization process. For example, obtaining the context of the previous frame in a keyframe-based summary is challenging. In addition, it lacks the original sound. Consequently, keyshot-based video summarization methods are predominantly selected to overcome these challenges.

Keyshot-based video summarization methods are used to produce subsets for short-form videos (i.e., user-generated TikToks and news) or long-form videos (i.e., feature-length films and soccer matches). Generally, the lengths of short- and long-form videos are lesser and more than 10 min, respectively [13]. As the playback duration of short-form videos is already very concise, it is impractical and may be ineffective in generating keyshot-based subsets for such videos. Moreover, the playback duration for long-form videos can exceed 90 min, specifically for movies or sports videos. Keyshot-based summary methods are more practical and effective for such video categories to provide a quick glimpse to users.

Computational Bottleneck: Each video contains a variety of information such as character appearance, motion, interactions between objects, events, and scenes. Considering a 1-h long-form video at 25 frames per second (FPS), it comprises thousands of frames. Existing approaches require extensive computational resources to properly process the entire video data (i.e., the frames) [7], [8], [9], [12]. If the video has an overly high definition, the demand for computing resources will increase. Deep learning-based methods also require segmented processing of long-form videos, further increasing the number of processing steps and computational complexity [2]. Thus, these types of approaches may not be suitable for resource-constrained devices, as the device must process all frames, which increases the overall computational time. Considering that computational resources are limited, lightweight keyshot-based summarization methods for the long-form are lacking.

Privacy Bottleneck: Video summarization is a daunting task because of its subjectivity. This is because every user has different preferences, even for similar video content. The personalized video summarization method provides precise solutions to this problem [14]. The algorithm is aimed at generating customized content for every user according to their interests. However, personalized video summaries with optimal lengths for new long-form videos (e.g., sports matches) are not immediately available. With current approaches [2], [3], generating personalized summaries in real time requires enormous computational resources to process user preference data and video content. Centralized dedicated servers can provide real-time, personalized video summaries. However, server-based personalized solutions would require the server to have access to users' preference data along with video content, which could lead to users' privacy concerns.

In the context of computational and privacy bottlenecks, we propose a client-driven approach to create personalized keyshot video summaries on resource-constrained devices.

A. OUR CONTRIBUTION

This paper proposes a novel client-driven framework called LTC-SUM that uses lightweight thumbnail containers in the summarization process. It handles the complex process of detecting personalized events (such as penalty shoot-out in soccer videos) from lightweight thumbnails. This makes the proposed approach computationally efficient because the entire video is not processed. In addition, the technique is efficient in terms of communication (between the server and the client) and storage requirements, as the entire video does not need to be transmitted over the network and stored. Contrary to previous keyshot-based methods [6], [7], [8], [9], [10], [11], [12], this study was aimed at generating subsets for long-form videos such as movies and documentaries. The proposed approach is a fully client-driven application that can generate distinct video summaries separately for concurrent users according to their interests (see Figure 1 for an example). The main contributions of this study are summarized as follows:

- A novel thumbnail-based client-driven framework is proposed to generate keyshot video summaries according to user preference. The proposed LTC-SUM framework aims to resolve the bottlenecks of computation resources and user privacy.
- To the best of our knowledge, this is the first study to develop a complete client-driven technique for creating personalized video summaries using thumbnail containers.
- A lightweight two-dimensional convolutional neural network (2D CNN) model was designed to identify personalized events from thumbnails.

- Quantitative and qualitative evaluations were conducted on full long-form eighteen videos (approximately 32.9 h in duration). Extensive quantitative experiments showed that the proposed method is more computationally efficient than the SoA baseline methods for the same client device configurations (Section IV-C). The qualitative evaluations were conducted with the collaboration of 56 participants (Section IV-D).

The remainder of this paper is organized as follows. Section II provides a summary of related work. Section III discusses the proposed lightweight client-driven personalized video summarization approach. A detailed implementation of the video summarization framework along with the experimental results and discussion is presented in Section IV. Finally, Section V summarizes the paper and provides concluding remarks.

B. NOTATIONS AND DEFINITIONS

The following definitions are used throughout this paper:

- **Segment (*Seg*):** A video is a combination of sequences of distinct segments *Seg* (or chunks), where the duration of each segment is a few seconds.
- **Frame:** The video consists of a sequence of individual moving images, each of which is called a frame.
- **Event/Action:** Event/action corresponds to certain types of activity, such as penalty shoot-out in a soccer match or horse-riding in western movies.
- **Thumbnail container (*ThuCon*):** Thumbnail container *ThuCon* is a collection of thumbnails extracted from the video. The sequence of all *ThuCon* covers the entire video length.
- **Thumbnail (*Thum*):** Thumbnail *Thum* is obtained from the video frame. A single *ThuCon* has 25 *Thum*, which is used in the video player to instantaneously preview the video.[1]

II. RELATED RESEARCH

This section briefly reviews existing works on keyshot-based and personalized video summarization methods. It also reviews existing action recognition methods, which are important for identifying personalized events from thumbnails in our proposed method.

A. KEYSHOT BASED SUMMARIZATION

The main idea of video summarization is to generate a short version of the original video. Video summarization techniques are divided into two categories i) keyframes [2], [3], [4], [5] and ii) keyshots [6], [7], [8], [9], [10], [11], [12]. The keyframe summarization methods generate a quick glimpse of the video as a set of images, however a significant amount of valuable information is omitted. Keyshot summarization methods attempt to overcome this challenge and provide more informative summaries in video form. Wang *et al.* [6] proposed a web-based event-driven video summarization method using tag localization and keyshot mining. Initially, the tags associated with each video are localized and included in its shots, and the relevance of the shots for the event query is estimated. A set of keyshots is then classified from the shots by performing near duplicate keyframe detection. Song *et al.* [7] used an aesthetic measurement to detect the segmentation point changes from a video. They used the K-nearest neighbor algorithm for clustering to remove redundant frames. A sequential decision-making method for a video summarization task was proposed in [8]. The deep summarization network (DSN) was designed to obtain the probability and selection of every frame in the video. Fajtl *et al.* [9] proposed a summarization method using a soft self-attention mechanism with two fully connected layers, with a sequence-to-sequence network. The network is used to process the CNN features of video frames and compute the frame-level importance scores. Later, this information was utilized in the relevant segment selection process from the video. A deep side semantic embedding (DSSE) model that leverages queries as a side information method is proposed in [10]. The DSSE architecture consists of two subnetworks, each with a unimodal autoencoder. One DSSE autoencoder encoded the video frames as input, and the other encoded the side information of the textual information associated with the video. The keyshot summary is generated by minimizing the distance between the selected video frame and side semantic information in the latent subspace. The importance of the sequence of video frames is measured using the proposed supervised-based encoder-decoder network [11]. This information is used to generate a series of keyshots containing humans as output. The encoder uses a bidirectional long short-term memory (LSTM) network to encode contextual information between the input video frames. The decoder uses two attention-based LSTM networks. The encoder-decoder model is used to convert the frame-level importance score into a shot-level score in the summarization process. Recently, another summarization approach was proposed using a generative adversarial network (GAN) [12]. An embedded actor-critic with a GAN model is designed to select the most important frames from the video. Subsequently, the selected frames were combined to generate a video summary.

B. PERSONALIZED VIDEO SUMMARIZATION

Although the above-mentioned keyshot summarization techniques are valuable, they miss an influential element in the summarization process; that is user preferences. The personalized summarization techniques utilize preference-based events, shots, and features to create personalized summaries that correspond to users' interests. Wei *et al.* [15] proposed a personalization method that adapted video content based on both client devices' resource constraints and user-provided keywords. In [16], long-form first-person tourist videos and user preferences were analyzed as input, and a subset of the video was returned as output. For each video, shot

[1]The number of *Thum* in a *ThuCon* can be varied. However, the number of *Thum*s was fixed to 25 in this study, based on our study on web-based YouTube player.

TABLE 1. Comparison of proposed framework with existing well-known video summarization methods.

Methods	Content Analysis				Video length		Summary Type	
	Video	Frames	*Segs*	*ThuCon*	Short-form	Long-form	Key-frame	Key-shot
Song, Yale, et al. [7]		✓			✓		✓	✓
Yuan, Yitian, et al. [10]		✓			✓			✓
Lei, Jie, et al. [2]			✓			✓	✓	
Thomas, Sinnu Susan, et al. [3]	✓				✓		✓	
Zhou, Kaiyang, et al. [8]		✓			✓			✓
Fajtl, Jiri, et al. [9]		✓			✓			✓
Huang, Cheng, et al. [4]	✓				✓		✓	
Ji, Zhong, et al. [11]		✓			✓			✓
Ma, Mingyang, et al. [5]		✓			✓		✓	
Apostolidis, Evlampios, et al. [12]		✓			✓			✓
Proposed				✓		✓		✓

boundaries were detected using clustering and personalized saliency calculated by comparing the video segment to the user profile preferences and visual attention score. Li *et al.* [17] used short- and long-term audio-visual temporal features to detect substories from movies. The length of the generated summary was adjusted according to user preference. The technique proposed in [18] allowed users to select content, type of shots, and summary length in the personalized movie summarization process. However, movie content and user preferences are equated at the feature stage instead of at the semantic level. The emotions of viewers and their attention are used to create summaries in [19]. The viewer's mood is identified with the help of different facial expressions such as blinking, head, and eye movements observed while the viewer is watching a video. A summarization technique that relies on user-generated comments was proposed in [20]. They used real-time comments from the movie created by the audience on different timestamps. The number of comments showed the excitement of the audience, and the content of the comments provided an idea of the current scene. Recently, another method proposed to generate personalized content using an end-to-end automated directing system for multi-camera sports broadcasts, driven entirely by the semantic understanding of sporting events [21].

C. ACTION RECOGNITION METHODS FROM VIDEOS

In the context of the personalized summarization process, detecting shots or events from a video according to the user's preferences is a crucial and challenging task. Thus, the detection of relevant events from thumbnails is essential in the proposed summarization method. Current action recognition techniques are used in the proposed method to complete this phase. In this context, some prominent SoA-CNN techniques were reviewed. CNN models have surpassed conventional approaches in recent studies [22], [23], [24], [25]. This is because they are more reliable and generalizable for extracting holistic features compared than are handcrafted. For this, variants and extensions of 2D CNNs and three-dimensional CNNs (3D CNNs) are applied to pictures. 2D CNNs perform only spatial operations on a single image. However, 3D CNNs can perform spatial and temporal operations while maintaining temporal dependencies between input video frames [22]. In [23], researchers used a 3D CNN with a support vector machine (SVM) and an independent subspace analysis (CNN-ISA) to identify human actions from the video. Similarly, another variant of a CNN network C3D was used to extract later-fed video features to SVM to identify the action [22]. Unlike previous methods, another CNN-based SoA action recognition method uses two types of streams, namely, the spatial and temporal streams [24], [25]. The video decomposes into spatial (RGB representation) and temporal (optical flow representation) components. Subsequently, video frames were fed into two separate 3D CNNs.

The autonomous video summarization process for long-form videos in real-time on a client device is still an open problem. This is mainly because the current summarization techniques use segments [2], or entire video/frames [3], [4], [5], [8], [9], [10], [11], [12] data in a process that requires enormous computational resources, as shown in Table 1. However, most modern end-user devices have low computational resources, and processing the entire video or frames takes a significantly long time to generate a summary on the client device. This is not a feasible real-time requirement. In addition, video summarization methods that require semantic information in the process may require mining and pre-processing steps [10] to obtain useful information, and information may not be publicly available for all videos, such as scripts and subtitles [2]. This further increases the demand for computational resources and processing time. Thus, in this proposed work, lightweight thumbnail containers are used in the summarization process, which makes the computation process, communication, and storage efficient to create a real-time summary on the user end. Consequently, our proposed approach aims to resolve the computational and privacy bottlenecks of the personalized video summarization technique.

III. PROPOSED LTC-SUM VIDEO SUMMARIZATION FRAMEWORK

A normal video is a combination of continuous moving frames at a rate of 25 FPS. As described in Section II-A, the well-known techniques that process the entire video (i.e., all frames) to generate summaries are not computationally efficient. Intuitively, for any given unit of time

(i.e., one second), there exists a significant amount of redundancy in frames [26]. Thus, generating a summary by processing all frames is inefficient because unnecessarily redundant frames will also be processed, thus wasting a significant portion of the limited computational resources. Considering that computational resources are limited and expensive, it would be desirable to avoid processing frames that have a high correlation (i.e., redundant frames). In this context, a novel thumbnail-based approach is proposed to generate personalized video summaries with the aim of reducing the waste of computational resources and reducing computation time. The use of lightweight thumbnails instead of frames enables us to generate summaries within the limit of acceptable computation time for end-user devices such as the Nvidia Jetson TX2.[2]

Figure 2 illustrates the high-level system architecture of the proposed lightweight video summarization framework. The system comprises two main parts: the HLS IIS server and HLS client. In the following, we explain the configuration and role of each component of the proposed framework.

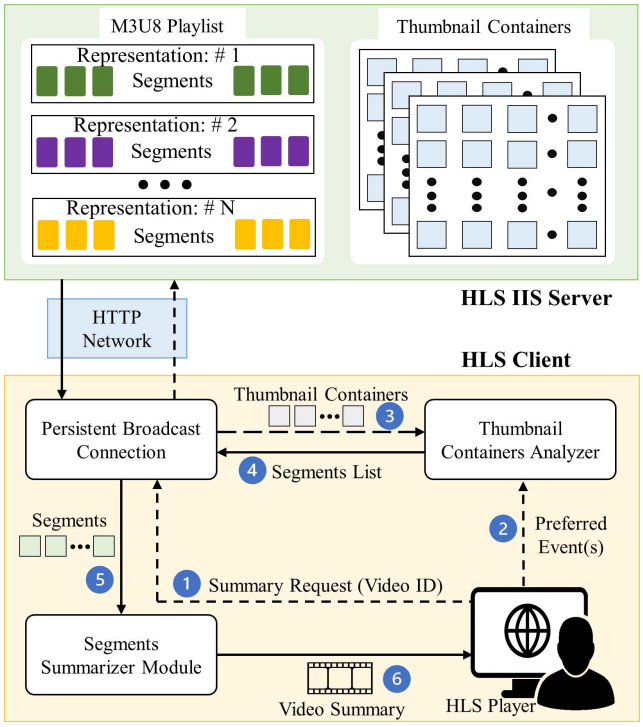


FIGURE 2. High-level system architecture of the proposed lightweight video summarization framework.

A. HLS IIS SERVER

The first component of the system architecture was the HLS server. The HLS server was configured locally on Microsoft Windows 10 Internet Information Services (IIS). This configuration allows multiple heterogeneous devices to concurrently download *ThuCon* and *Seg* for a given video from the HLS IIS server. The IIS supports a wide range of network protocols such as HTTP, HTTPS, and FTPS (refer to [27] for the complete list). Initially, by using FFmpeg [28], the entire video is encoded as the H.264/AAC MPEG-2 transport stream (.ts) segments. The MPEG-2 transport stream is suitable for transmission when there is a potential corruption or loss of data packets [29]. Each *Seg* consists of approximately a playback portion of 10 seconds of the video, with a continuous timestamp. The text-based playlist file (M3U8) contains a list of *Seg*s according to their playback order. Each bitrate playlist contains URLs pointing to the *Seg* files.

[2]Note that the approaches that require the processing of the entire video can also be deployed on the end-user device, but this will lead to a significantly higher computation time. This is discussed in detail in Section IV-C.

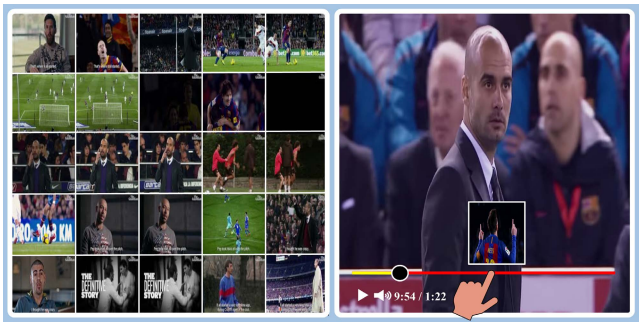


FIGURE 3. Orientation of thumbnails on a single thumbnail container image (left), and the thumbnail usage for instant preview in the client web-based YouTube video player (right).

In addition to the *Seg*s, the HLS IIS server also contains *ThuCon*s, which are extracted from the corresponding video using FFmpeg [28]. Each *ThuCon* has 25 *Thum*s, where a single *Thum* of a video corresponds to the first frame of each second of the video. Thus, each *Thum* represents one second of the video, and a single *ThuCon* represents 25 seconds of the video. The sequence of all the *ThuCon* covers the entire video length. Based on our study on YouTube web-based player, in this work, the size of each *Thum* was fixed at 160×90 (*width* $\times$ *height*) pixels and *ThuCon* to 800×450 (*width* $\times$ *height*) pixels. Figure 3 illustrates an example of a *ThuCon* of a documentary received in the client web-based YouTube video player (left), and a *Thum* previewing a particular duration (right).[3] Next, the configuration and the role of the HLS client are discussed.

B. HLS CLIENT

The purpose of the HLS client is to process video-related information obtained from the HLS IIS server using the end-user computational resources and to locally generate a personalized summary of the corresponding video. For this purpose, a Nvidia Jetson TX2, which has an embedded AI computing device, is configured as an end-user computational resource. It is a GPU-based board with a Nvidia Pascal 256 CUDA core architecture along with a 64-bit hex-core ARMv8 CPU; stacked with a memory of 8 GB, and

[3]The documentary was Take The Ball, Pass The Ball, and it can be obtained from URL https://www.youtube.com/watch?v=VfKls9Eo1ZI.

59.7 GB/s 128-bit interface of memory data transfer capacity [30]. The Jetpack 4.3 SDK is used to automate the basic installations on Nvidia Jetson TX2, which includes board support packages and libraries, especially for deep learning and computer vision. The Nvidia Jetson TX2 supports several energy profiles and the max-n profile used in the proposed approach. The HLS client consists of four major components: the persistent HTTP connection to download *ThuCon*s and personalized *Seg*s from the HLS IIS server; the deep learning-based action recognition model to recognize personalized events from thumbnails; the summarizer module to aggregate the different timestamp segments; and the web-based HLS video player for the user interface to generate a personalized summary.

1) HTTP PERSISTENT CONNECTION

During the generation of personalized video summary, the client initiates several requests to obtain *ThuCon*s and *Seg* of the corresponding video from the HLS IIS server. For this purpose, a cost-effective HTTP 2.0 persistent connection was used to download *ThuCon*s and *Seg* from the HLS IIS server. This connection enabled the exchange of numerous requests, and it returned data simultaneously in a single TCP connection. An open connection is faster for frequent data exchanges, as it remains open for HTTP requests and responses rather than closing after a single exchange. The performance of the persistent connection adaptive streaming was evaluated in [31]. Using a persistent connection has several advantages; for example, the overall CPU usage and round trips are reduced because of fewer new connections and TLS handshakes [32].

2) THUMBNAIL CONTAINERS ANALYZER

The main task of the thumbnail container analyzer is to detect the preferred events from *Thum*s. Then, based on the selected *Thum*s, generate a list of personalized *Seg* and use them to produce a personalized summary. For this purpose, a lightweight 2D CNN model was designed to detect personalized events from each thumbnail. To detect the personalized thumbnail based on the preferred events of the user from each *Thum* with high accuracy, the CNN model must be trained using thousands of images, which requires high processing GPU power. In this context, transfer learning [33] is useful, in which a pre-trained model is used for other purposes. This method was applied to train the EfficientNet-B0 [34], which was trained on a large-scale ImageNet [35] dataset to extract the frame-level spatial features of each thumbnail. Compared with other ConvNets, EfficientNet outperforms state-of-the-art architectures on ImageNet and has fewer parameters and FLOPS [34]. This makes EfficientNet [34] a suitable candidate for detecting personalized events from lightweight *Thum*s. The backbone of the proposed network was based on EfficientNet-B0 [34]. Figure 4 shows the proposed action recognition model used to process all the *Thum*s and detect personalized events. An attention module was used to improve the performance of the proposed network [36]. The attention module is more effective by using multibranch convolution with different dilation rates to aggregate contextual information. Different dilation rates can effectively improve the receptive field, consequently acquiring multi-level contextual information. The architecture of the vortex pooling used as an attention module is depicted in Figure 5.

The proposed 2D CNN model was trained on the UCF101 dataset, which is a well-known action recognition dataset [37]. It consists of 13320 videos taken from YouTube, which are divided into 101 action categories [37]. In the proposed approach, data augmentation is applied [38] to reduce overfitting; this method has been proven to be very effective. To train the model using the UCF101 dataset, each video was subsampled down to 40 frames. Before being provided as input to the network, all images were preprocessed by first cropping the center region, and then resizing them to 244×244 pixels. A shear transformation was also performed at an angle of 20°, horizontal and vertical shift of 0.2, random rotation of 10°, and random horizontal flipping of images.

The dataset is split into two subsets: training and testing – as suggested in [37]. The model is trained using a variant of stochastic gradient descent (SGD) with a momentum of 0.9, and a learning rate of 0.01 – using the default weight decay value (SGDW) [39]. In the experiments, an early stopping mechanism was applied in the training process with a patience of ten epochs. The Keras toolbox was used for deep feature extraction, and a GeForce RTX 2080 Ti GPU was used for implementation. The training data were fed in mini batches with a size of 32 and a learning rate of 0.001 for cost minimization, and there were one thousand iterations for learning the sequence patterns in the data. The action recognition accuracy analysis of the model is presented in Section IV-B. In the following section, the third component of the HLS client is described, which is the segments summarizer module.

3) SEGMENTS SUMMARIZER MODULE

The purpose of the summarizer on the client is to aggregate all the downloaded personalized *Seg* into a single continuous video stream using FFmpeg [28]. The module is scalable; however, currently, it only supports continuous stream playback in the proposed approach. Note that there are no restrictions for fixing the length of the generated summary in the proposed architecture. However, as the client downloads all the personalized *Seg*, a module can be integrated into the system, which manages the summary length according to user the preference. The web-based HLS video player is explained in the following section.

4) WEB-BASED HLS VIDEO PLAYER

The HLS video player provides functionalities for the user to choose the video title and personalized event(s) according to their preference and then play the generated summary. The interface was designed using an open-source HTML5 HLS video player [40]. Figure 6 shows HLS IIS server containing *Seg*s and *ThuCon* (left side), web-based HLS video player interface displaying generated summary (right side).

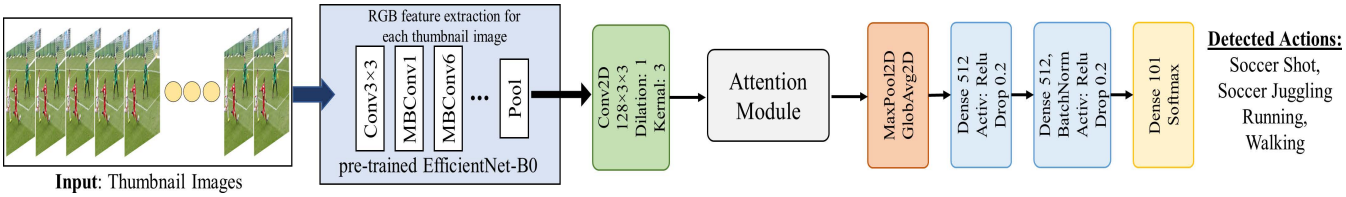


FIGURE 4. Proposed lightweight action recognition model used to analyze personalize events from thumbnails in the video summarization process. RGB thumbnails are forward propagated through a 2D CNN model to extract features from the fully connected layer.

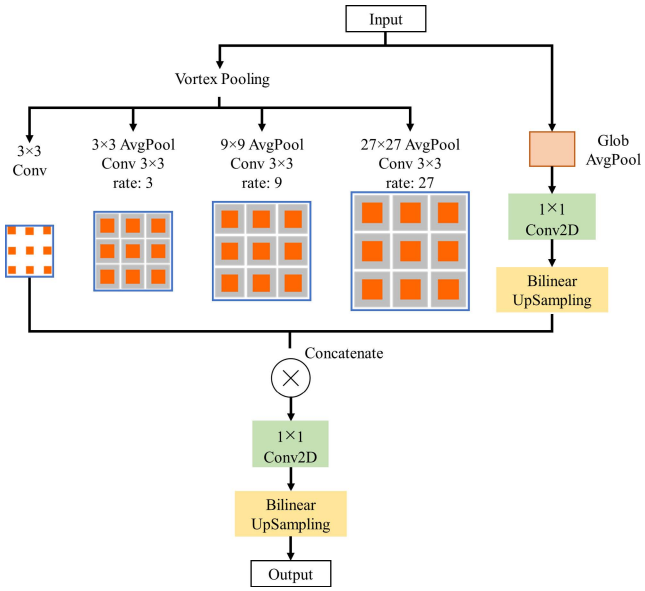


FIGURE 5. Architecture of attention module in the proposed 2D CNN model.

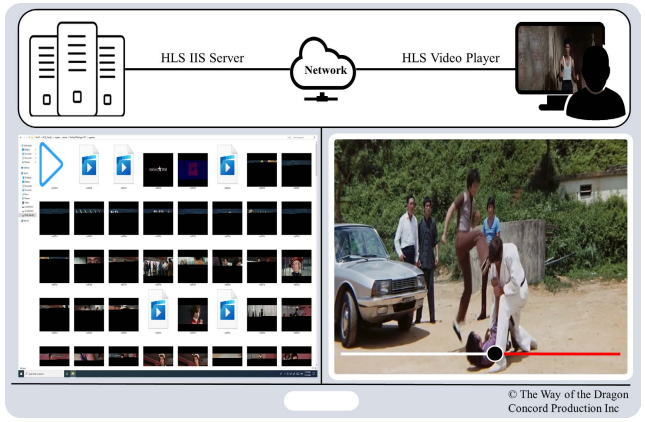


FIGURE 6. HLS IIS server containing segments and thumbnail containers (left side), web-based HLS video player interface displaying generated summary (right side).

It supports VoD sessions, and media content (e.g., segments and playlists) can be assessed in the VoD session on the client side. The list of segments in their playback order is stored in a text-based M3U8 playlist file. The player can use the M3U8 playlist to determine the available bitrates and locations of the *Segs*. The data delivery is entirely client-driven, which means that the video player can determine when to request each segment from the playlist file in the playback order or with a specific timestamp. In addition, it can shift between different video bitrates during playback.

IV. EXPERIMENTAL RESULTS AND DISCUSSION

In this section, we present an extensive experimental investigation of the proposed approach. First, the experimental setup of hardware specifications used in experiments are described. Then, the complete flow of the proposed video summary generation is explained from the user perspective. Then, the accuracy of the proposed action recognition 2D CNN model is presented and compared with those of other well-known approaches. Finally, the performance of the proposed LTC-SUM method is compared with baseline video summarization methods along with discussion.

A. EXPERIMENTAL SETUP

1) HARDWARE CONFIGURATION

In the experimental evaluation, the HLS clients and the HLS server were locally configured. Two different types of hardware configurations were used for the HLS client. The high computational resources (HCR) device was run on an open-source Ubuntu 18.04 LTS operating system with dual quad-core 2.10 GHz Xeon processors, GeForce RTX 2080 Ti, and 62 GB RAM. The low computational resources (LCR) device was a Nvidia Jetson TX2. Three distinct experimental setups were configured for the HLS client: (i) the proposed LTC-SUM approach was configured on the LCR and (ii) on HCR devices, and (iii) all the baseline approaches were configured only on the HCR device. The HLS IIS server was configured on Windows 10 in all experiments. All hardware devices were locally connected to the Sungkyunkwan University network. Table 2 lists the specifications of each hardware device used in the experiments. The entire video

TABLE 2. Specifications of hardware devices.

Device	CPU	GPU	RAM
HCR Client	Quad-core 2.10 GHz Xeon	GeForce RTX 2080 Ti	62 GB
LCR Client	Quad ARM A57/2MB L2	Nvidia Pascal 256 CUDA cores	8 GB
HLS IIS Server	Intel Core i7-8700K	GeForce GTX 1080	32 GB

summarization process using the proposed approach is explained in the following subsection.

2) PROPOSED THUMBNAIL-BASED SUMMARIZATION PROCESS

This section provides the complete flow of the proposed video summary generation process from a user perspective. This flow is described based on a set of 18 video titles used for the experiments. A complete description of the set of videos is provided in Table 3. The genres of the cinematographic movies and documentaries analyzed were Western, sport, and action.[4] Since a movie/documentary may consist of more than one genre, the most dominant genre is considered (i.e., western, sport, or action). Initially, the user selects a video title from the list of available video titles using the web interface. In the experiments, the user could select a video title from among 18 video titles with different playtimes and each with the a frame size of 640 × 480 pixels.

Depending on the video genre selected by the user, they were asked to choose the recommended event(s) from the list of events corresponding to the selected video genre. In the experiment, based on the set of videos described in Table 3, ten distinct event(s) could be selected from the UCF101 action categories list. These events are archery, cricket-bowling, cricket-shot, horse-race, horse-riding, nunchucks, punch, soccer-juggling, soccer-shot, and tai-chi. These events were selected and categorized based on the genre of the video title. Figure 7 shows sample images of the selected events for analyzing the video.

Once the user selects the preferred event(s), the HLS client downloads all the *ThuCon*s of the corresponding video from the HLS IIS server. Note that the downloaded *ThuCon*s cover the entire length of the video, and a very low bitrate is required to transmit all the *ThuCon*s from the server to the client. This is because *ThuCon* tends to be lightweight in terms of size and small in number compared with the frames of the same video (refer to Table 3 for quick comparisons). After obtaining all the *ThuCon*s, the system extracts *Thum*s from *ThuCon*s, and the pre-trained 2D CNN model proceeds by analyzing all of them based on the preferred event(s) of the user. All *Thum*s relevant to the preferred event(s) are listed. Based on the shortlisted *Thum*s, the system generates a text-based list of detected *Thum*s in chronological order according to the *Thum* number. The list provides temporal information about the personalized *Seg*s that need to be used to generate a personalized summary of the requested video. The text-based list of detected *Thum*s was prepared separately whenever a new process started for each video title.

The system determines the *Seg* number from the text-based list based on the detected personalized *Thum*s, and requests to download *Seg*s with different timestamps from the HLS IIS server. If a *Seg* takes too long to download, an alternate bitrate can be selected. Once all *Seg*s are received, the system aggregates them into one continuous video stream using FFmpeg [28], in which a user can watch using the web-based HLS video player interface. The described flow of the proposed thumbnail-based summarization process is illustrated in Figure 8.

[4]The eighteen video titles arbitrarily chosen consisted of three genres (Western, sport, and action) for the experimental evaluation. However, the proposed approach is not limited to these titles and can be used for arbitrary video titles and genres.

FIGURE 7. First six images display sample events for action and western genre videos: archery, nunchuck, punch, horse-race, and horse-riding. The last four images display selected events for sports genre movies: soccer-juggling, soccer-shot, cricket-bowling, and cricket-shot.

3) BASELINE APPROACHES

In this subsection, the baseline approaches are described for comparison with the proposed thumbnail-based method. As explained in Section II-A, well-known video summarization approaches process every frame of the corresponding video to generate a summary. Thus, some of the prominent SoA techniques were adopted as the baseline approaches in this study. To generate the summary using baseline methods, all videos in Table 3 were stored locally. However, the proposed LTC-SUM approach does not require the videos to be stored locally which brings an additional gain in storage efficiency. The baseline approaches were as follows:

- **HECATE [7].** It analyzes the aesthetic features from all temporarily extracted frames of the corresponding video. This method only supports fixed summary lengths, and a five-minute subset is generated for each video.
- **DR-DSN [8].** It is trained based on the SumMe dataset [41] using the default parameters. Initially, this method extracts frames from the corresponding video and then analyzes the extracted frames to generate a video summary. Using DR-DSN with default parameters, the summarization duration generated for all corresponding videos was 22 seconds.
- **VASNet [9].** Similar to DR-DSN [8], it is trained based on the SumMe dataset [41] using default parameters. First, it extracts frames from the corresponding video and then analyzes extracted frames to generate a subset. For every corresponding video, a 24-second subset is generated using the default parameters of VASNet.
- **AC-SUM-GAN [12].** Similar to the aforementioned baseline methods [8], [9], it first extracts frames from the corresponding video and then analyzes the extracted frames to generate a subset. It is trained based on the SumMe dataset [41], and it generates at 19-second summary for every corresponding video with the default configuration.

TABLE 3. List of video titles and their details used for analysis in the proposed approach.

S/N	Title & Year	Genre	IMDB	Duration	FPS	#Frames	$\#ThuCon$	$\#Thum$
1	89-2017	Sport	7.8	1h31min	25	135300	217	5412
2	Bobby-2016	Sport	7.1	1h37min	25	1420204	225	5608
3	Bruce Lee the Man & the Legend-1973	Action	6.7	1h25min	24	123658	207	5152
4	Django-1966	Western	7.3	1h22min	24	131749	220	5495
5	Django Unchained-2012	Western	8.4	2h45min	24	237909	397	9922
6	Goal-2005	Sport	6.7	1h58min	23	169932	284	7087
7	Little Big Man-1970	Western	7.6	2h19min	24	2005509	335	8362
8	M.S. Dhoni-2016	Sport	7.7	3h4min	24	265933	444	11080
9	Oklahoma!-1955	Western	7	2h25min	24	201422	337	8401
10	Shanghai Noon-2000	Western	6.5	1h50min	23	158648	265	6617
11	Snake In The Eagle's Shadow-1978	Action	7.4	1h30min	24	140473	235	5858
12	Take The Ball, Pass The Ball-2018	Sport	8.2	1h49min	25	163972	263	6558
13	The Indian Fighter-1955	Western	6.4	1h28min	23	127126	213	5302
14	The Legend of Drunken Master-1994	Action	7.6	1h42min	24	147454	247	6150
15	The Rider-2017	Western	7.4	1h44min	24	148404	236	5886
16	The Train Robbers-1973	Western	6.5	1h32min	23	132192	221	5514
17	The Way Of The Dragon-1972	Action	7.3	1h30min	24	142712	239	5953
18	Vengeance Valley-1951	Western	5.9	1h23min	30	147448	197	4919

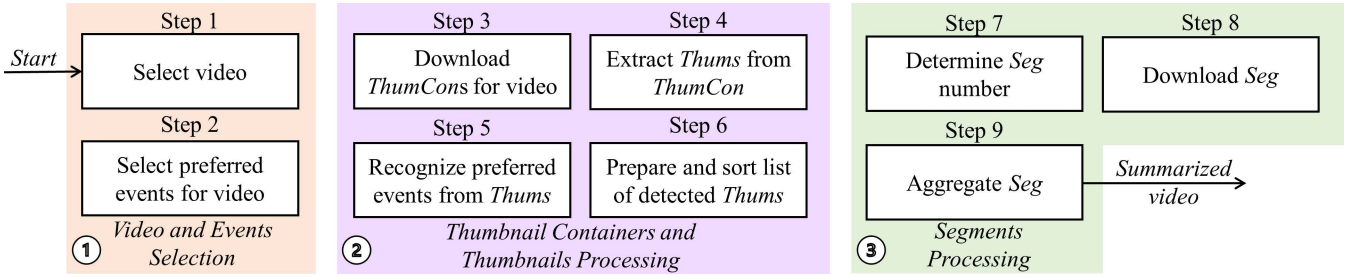


FIGURE 8. Schematic overview of the proposed LTC-SUM summarization process.

- **FB-SUM.** This is the frame-based summarization (FB-SUM) baseline method that analyzes every frame of the corresponding video during the process. Initially, the video frames were extracted using FFmpeg [28] for each video. The rest of the summation process followed the same steps as the proposed LTC-SUM method.

As highlighted before, there is a redundancy in frames while processing the entire video; thus, processing all frames of a video is not computationally efficient as it increases the processing time and wastes a significant portion of the limited computational resources. To validate the effectiveness, the computation time of the baseline approaches were compared with the proposed LTC-SUM approach in the following experiments.

B. EXPERIMENTAL EVALUATION OF ACTION RECOGNITION DATASETS

In this subsection, the accuracy of the proposed action recognition 2D CNN model is evaluated using the benchmark action recognition dataset UCF101 [37]. To the best of our knowledge, the best thumbnail-based approach was proposed in [42], therefore, we compared our results with [42]. The proposed model reported the highest validation accuracy of 77.81% in 36 epochs with 55.74 million flops. The proposed method achieved an increase of 4.06% in the validation accuracy, increasing from 73.75% [42] to 77.81%, and the training accuracy increased from 91.41% [42] to 96.06%. The work in [42] used InceptionV3, which has 24M parameters; however, EfficientNet-B0 was used as the backbone network in the proposed LTC-SUM method, which has 6.9M parameters. Comparisons with the other methods are summarized in Table 4.

TABLE 4. Comparison of average recognition score of the action recognition proposed method with other methods.

Methods	UCF101
Karpathy, Andrej, et al. 2014. [43]	65.4%
Murthy, OV Ramana, et al. 2015 [44]	72.8%
Mujtaba, et al. 2020 [42]	73.75%
Shu, Yu, et al. 2018 [45]	76.07%
Mujtaba, et al. 2022 [46]	76.25%
Liu, An-An, et al. 2016 [47]	76.3%
Proposed Method	**77.81%**

C. PERFORMANCE EVALUATION OF THE PROPOSED LTC-SUM METHOD

This subsection compares the performance of the proposed LTC-SUM video summarization method with the baseline schemes described in Section IV-A3. The performance evaluation experiments used ten different events to analyze the

TABLE 5. The total computation time required in minutes to generate a summary using baseline methods on an HCR device, and the proposed LTC-SUM method on HCR and LCR devices.

S/N	HECATE [7]	DR-DSN [8]	VASNet [9]	AC-SUM- GAN [12]	FB-SUM	Proposed LTC-SUM	
	HCR					LCR	HCR
1	20.85	4.92	5.09	4.81	70.41	7.64	**1.98**
2	22.02	5.15	4.77	4.87	67.23	8.16	**1.91**
3	28.51	4.10	4.31	4.43	60.91	5.56	**1.98**
4	21.05	5.13	5.32	4.74	63.67	8.25	**2.10**
5	54.41	8.58	8.36	8.69	132.32	14.64	**3.58**
6	28.56	6.32	6.18	5.92	80.60	10.03	**2.43**
7	51.91	7.32	7.03	7.04	102.16	12.07	**3.02**
8	85.23	9.53	9.93	9.52	123.29	15.87	**4.22**
9	53.49	7.74	7.91	7.98	118.60	12.23	**2.93**
10	32.03	5.51	5.57	5.64	82.65	9.85	**2.36**
11	25.50	5.24	4.91	4.86	73.14	6.49	**2.23**
12	28.29	6.01	5.88	6.16	131.83	9.87	**2.31**
13	18.71	5.21	4.75	4.64	65.36	7.90	**1.95**
14	31.33	5.23	5.23	4.98	77.21	6.78	**2.27**
15	17.20	5.57	5.09	4.97	73.45	9.08	**2.32**
16	23.11	4.78	4.77	4.81	60.82	7.93	**2.06**
17	29.00	5.39	5.29	5.20	78.08	6.67	**2.25**
18	24.59	5.55	5.57	5.08	74.36	7.64	**1.91**

proposed LTC-SUM and baseline approaches. The list of events is described in Section IV-A2. All the detected *Thum*s for the proposed LTC-SUM method and frames for the FB-SUM baseline method are included in the summarization process for which the detection accuracy was higher than 95% for western, 65% for action, 80% for cricket sports, and 90% for soccer sports videos.[5] The threshold of each video genre was selected to maintain the length of the summaries. The default parameters were used for the remaining baseline methods.

The computation time (in minutes) required to generate a video summary using the baseline and the proposed approaches were compared in the first experiment, where all approaches were configured on the HCR device (refer to Table 2 for detailed specifications of the device). The steps involved in calculating the computation time are (i) frame extraction from the video (FB-SUM baseline) and *Thum*s extraction from *ThuCon*s (proposed); (ii) event(s) recognition using the lightweight trained 2D CNN model from frames (FB-SUM baseline) and *Thum*s (proposed); (iii) determining and downloading *Seg*; (iv) and finally, aggregate *Seg* into a single continuous video stream. Meanwhile, default configurations and steps are used for HECATE [7], DR-DSN [8], VASNet [9], and AC-SUM- GAN [12] baseline approaches to generate summaries. Compared with the number of frames, the number of lightweight thumbnail images was significantly smaller (Table 3). Thus, the overall computation time of the proposed LTC-SUM method is significantly lower than that of the FB-SUM baseline approach. Table 5 shows the computation time in minutes required to generate a summary using baseline methods, on the HCR device. Extracting frames from the video and using all the extracted frames to generate a summary are the key factors in increasing the overall computation time while using the baseline methods.

Because this study focused on generating summaries resource-constrained on client end devices, in the next experiment, the proposed LTC-SUM method is configured on the LCR device (i.e., the Nvidia Jetson TX2). Table 6 lists the computation time required in minutes on every step to generate a summary using the FB-SUM baseline method on the HCR device and proposed LTC-SUM method on the HCR and LCR devices.

Table 7 depicts the duration of summaries generated automatically to detected frames/*Thum*s for the corresponding video using FB-SUM and LTC-SUM methods on the LCR device. From Table 6, it can be observed that the computation time for FB-SUM is significantly higher than that for LTC-SUM. In addition, it can also be observed from Table 5 that even when the proposed technique is implemented on the LCR device, the computation time is still significantly shorter than that of FB-SUM and HECATE [7] baseline approaches implemented on the HCR device.

Considering that the combined duration of all videos was 1,974 min, HECATE [7], DR-DSN [8], VASNet [9], AC-SUM-GAN [12], and FB-SUM baseline approaches took 595.789 min, 107.28 min, 105.96 min, 104.34 min and 1536.09 min using the computational resources of the HCR device to generate the 18 summaries for each video as shown in Table 5, respectively. Meanwhile, the proposed approach on HCR took 43.82 min to generate the 18 summaries for each video. Thus, based on the analysis of these 18 videos, computationally for the HCR device, on average, the proposed approach is 13.59, 2.45, 2.42, and 2.38 times faster than HECATE [7], DR-DSN [8], VASNet [9], and AC-SUM-GAN [12], respectively. The computational resources of the LCR device are very low compared to the HCR device, even when the proposed method is 3.57 HECATE [7]; and

[5]The threshold value directly impacts the duration of the generated summary. If a low threshold is selected, then a lengthy summary will be generated.

TABLE 6. Computation time required in every step to generate the summary using the FB-SUM method on the HCR device and the proposed LTC-SUM method on HCR and LCR devices.

S/N	$Thum$ Extraction			Events Recognition			Download $Segs$			Aggregate $Segs$			Total		
	FB-SUM	LTC-SUM		FB-SUM	LTC-SUM		FB-SUM	LTC-SUM		FB-SUM	LTC-SUM		FB-SUM	LTC-SUM	
		LCR	HCR		LCR	HCR		LCR	HCR		LCR	HCR		LCR	HCR
1	4.98	0.14	0.11	65.27	7.33	1.82	0.13	0.15	0.04	0.02	0.02	0.01	70.41	7.64	**1.98**
2	5.06	0.17	0.11	62.06	7.86	1.78	0.09	0.12	0.02	0.02	0.01	0.01	67.23	8.16	**1.91**
3	4.72	0.12	0.09	56.09	5.34	1.86	0.09	0.09	0.02	0.01	0.01	0.01	60.91	5.56	**1.98**
4	5.29	0.16	0.1	58.26	7.88	1.94	0.1	0.18	0.05	0.02	0.03	0.01	63.67	8.25	**2.1**
5	10.32	0.28	0.18	121.87	14.24	3.36	0.11	0.11	0.03	0.02	0.02	0.01	132.32	14.64	**3.58**
6	6.05	0.21	0.13	74.49	9.74	2.27	0.05	0.07	0.01	0.01	0.01	0.01	80.6	10.03	**2.43**
7	7.58	0.25	0.16	94.41	11.56	2.79	0.15	0.22	0.06	0.02	0.03	0.02	102.16	12.07	**3.02**
8	9.88	0.33	0.2	113.3	15.42	3.98	0.09	0.1	0.03	0.02	0.02	0.01	123.29	15.87	**4.22**
9	8.69	0.23	0.15	109.82	11.94	2.75	0.07	0.05	0.02	0.02	0.01	0.01	118.6	12.23	**2.93**
10	6.9	0.18	0.12	75.63	9.54	2.2	0.1	0.11	0.04	0.02	0.02	0.01	82.65	9.85	**2.36**
11	5.17	0.17	0.11	67.72	6.05	2.05	0.22	0.24	0.06	0.03	0.03	0.01	73.14	6.49	**2.23**
12	6.28	0.21	0.12	125.38	9.3	2.1	0.15	0.32	0.08	0.02	0.04	0.02	131.83	9.87	**2.31**
13	5.76	0.15	0.1	59.5	7.67	1.81	0.08	0.07	0.03	0.02	0.01	0.01	65.36	7.9	**1.95**
14	5.95	0.17	0.11	71.15	6.49	2.13	0.09	0.11	0.02	0.02	0.01	0.01	77.21	6.78	**2.27**
15	6.01	0.16	0.12	67.33	8.66	2.09	0.1	0.24	0.09	0.02	0.03	0.01	73.45	9.08	**2.32**
16	4.91	0.17	0.1	55.83	7.59	1.89	0.07	0.15	0.07	0.01	0.02	0.01	60.82	7.93	**2.06**
17	5.96	0.16	0.12	72.02	6.32	2.08	0.08	0.17	0.04	0.02	0.02	0.01	78.08	6.67	**2.25**
18	5.43	0.18	0.1	68.83	7.19	1.67	0.09	0.23	0.13	0.02	0.03	0.02	74.36	7.64	**1.91**

the 9.2 FB-SUM method is faster than the baseline approaches on the LCR device. In conclusion, these results show that the proposed method is computationally efficient even for an LCR device.

Note that the proposed approach is also efficient in terms of communication and storage compared to the baseline approaches. As in the baseline approaches, the complete video needs to be downloaded and stored. In the proposed approach, only the *ThuCon*s are downloaded and stored. Thus, compared with the complete video, the download time and storage requirements for *ThuCon*s are significantly less. For example, the size of the movie 89 (2017) is approximately 612 MB, while the size of the *ThuCon*s of the corresponding movie is just approximately 14 MB. In addition, DR-DSN [8], VASNet [9], AC-SUM-GAN [12], and FB-SUM baseline approaches need to store the original video along with the extracted frames during the summarization process. By comparing number of *Thum*s with number of frames in a video, the number of frames is very large. Thus, significant local storage is needed for the baseline approaches. Therefore, in addition to achiving the computational efficiency, the proposed LTC-SUM method is also efficient in terms of storage and communication requirements for the summarization process.

From Tables 5-7, it can be concluded that the proposed approach is significantly better than the baseline approaches in terms of low computational complexity and processing time for long-form videos. This superiority exists even when the proposed approach is configured on a significantly LCR device (Nvidia Jetson TX2). Interestingly, the duration of the summaries generated using the proposed approach was much smaller than the duration of the summaries generated using FB-SUM baseline approach (refer to Table 7). It is intuitive to ask what the impact of the significant reduction in computational time and the small duration of video summaries has on the quality of the summary. In the following section, the results of a comprehensive qualitative survey are presented to answer this question.

D. QUALITATIVE EVALUATION

This section presents an evaluation of the quality of the summaries generated using the proposed method by comparing it with the summaries generated using the FB-SUM baseline approach. Because this study focused primarily on personalized summaries, only the FB-SUM baseline approach was evaluated. The evaluation was based on a survey conducted with the help of 56 participants: 44 males and 12 females with an age range of 15–35 years– (i.e., most respondents were young). The participants were from nine different geographical locations and covered a wide range of professions; however, most respondents were researchers and faculty members.

The survey was based on 18 movies (refer to Table 3), depending on the genre of the video, and a list of options for event(s) was defined. The participants could choose to generate a personalized summary. The selected options of event(s) from the UCF101 dataset for Western genre videos were (i) horse-riding, horse-racing, (ii) archery, punch, and (iii) horse-riding, horse-racing, archery, and punch. For action genre videos: (i) archery, punch, (ii) tai-chi, nunchuck, and (iii) tai-chi, nunchuck, archery, and punch. The sports genre videos were divided into two categories:– soccer and cricket. The selected options of event(s) for soccer genre videos were (i) soccer-juggling, (ii) soccer-penalty, and (iii) soccer-juggling and soccer-penalty. For cricket genre videos: (i) cricket-bowling, (ii) cricket-shot, and (iii) cricket-bowling and cricket-shot.[6]

[6]Note that the proposed technique is not limited to the above-mentioned list of preference options for each video. For simplicity, we adopted the event from the UCF101 dataset and defined a list of preference options for each video. Proposing a sophisticated method that can generate a list of preference options is beyond the scope of this study.

TABLE 7. Duration of generated video summaries by analyzing images and requesting video segments in the second experiment for the FB-SUM and proposed LTC-SUM methods.

S/N	# Images Detected		# *Segs* Requested		Summary Duration	
	FB-SUM	LTC-SUM	FB-SUM	LTC-SUM	FB-SUM	LTC-SUM
1	4157	67	105	34	18m30s	**5m51s**
2	2522	22	85	15	15m37s	**2m34s**
3	1248	37	96	25	17m34s	**4m12s**
4	4940	106	107	51	18m11s	**8m55s**
5	4108	62	103	30	18m14s	**4m36s**
6	810	15	45	13	7m22s	**1m58s**
7	3278	110	129	54	22m39s	**9m49s**
8	1518	48	100	24	16m55s	**4m37s**
9	2197	45	99	16	16m56s	**2m52s**
10	1600	52	89	29	15m14s	**4m48s**
11	5487	120	204	72	34m36s	**13m1s**
12	7317	150	131	66	22m40s	**11m41s**
13	841	35	78	20	13m	**3m17s**
14	984	32	106	23	19m1s	**4m2s**
15	2858	130	75	43	13m12s	**7m25s**
16	1156	41	63	25	10m50s	**4m29s**
17	1986	80	103	47	18m40s	**8m15s**
18	5814	226	112	83	19m58s	**13m54s**

Each participant selected one of the movie titles and the corresponding option of event(s) from the list. For each movie title and the preferred option of event(s), two summaries were generated using the proposed LTC-SUM and FB-SUM baseline techniques. The performance of the generated video summaries was evaluated objectively using an exact rating scale. The participants were asked to rate the summary, which was considered better according to the three evaluation criteria: information coverage, visual pleasure, and general satisfaction. An anonymous questionnaire was created for the generated summaries so that the users could not determine which method (i.e., LTC-SUM or FB-SUM) was used. They were requested to watch both summaries and answer questions by ranking the results on a scale of 1–10 (1 being the worst and 10 being the best). Table 8 lists the questions and the average ratings given by the participant for each question for both approaches.

Despite the fact that the summary generated using the proposed approach was short and required less computation time as the entire analysis was based only on *Thum*s, the qualitative evaluation suggests that the proposed approach was almost the same (better in some aspects) compared with the FB-SUM baseline approach. From the results of Q1–Q2, we can say that the proposed method does not lose the personalized aspects in terms of the preferred events compared with the FB-SUM baseline. From Q3–Q4, it can be observed that the length of the summary is crucial, as most users prefer short summaries, thus leading to significantly higher average ratings for the proposed approach. In Q5, we specifically asked about the similarities among the summaries of both approaches, and the obtained results suggest that participants observed significant similarities with an average rating of 6.89. Based on this qualitative evaluation, it can be concluded that the proposed approach performs very well and receives higher average ratings compared with the FB-SUM baseline without losing significant important information

TABLE 8. Average rating (1∼10) of the FB-SUM baseline and proposed LTC-SUM approaches.

Questions	Baseline	Proposed
Q1: Did the generated summary give related actions (events) according to your preferences?	7.14	**7.59**
Q2: Rate generated summary.	7.16	**7.52**
Q3: Is the length appropriate for the generated summary?	6.45	**7.39**
Q4: Compare to both generated summaries which one is good rate.	6.89	**7.32**
Q5: Correlations (similarities) of the generated summaries.	6.89	
Q6: Would you like to watch the movie after watching the generated summary?	7.09	**7.14**

(e.g., preferred events). Figure 9 depicts the sample frames obtained from the video summary generated using the proposed LTC-SUM method.

E. DISCUSSION AND LIMITATIONS

In the previous sections, we evaluated the overall effectiveness by comparing the proposed LTC-SUM method with the baseline approaches. The proposed framework exhibited a better performance by lowering the computational complexity and computation time in the summarization process. During quantitative experiments, it was observed that there were many redundant images (frames) to determine the segment numbers using the FB-SUM baseline approach. Thus, a significant portion of the computational resources are used to process the redundant frames and determine the segment number from the detected frames. Meanwhile, the proposed LTC-SUM method needs to process fewer images (thumbnails) to determine the segment number, as shown in Table 7. This significantly reduces the computational time, the demand for computational resources, communications, and storage required to generate summaries. In addition, the proposed method can solve the computational and privacy bottlenecks on the resource-constrained end-user devices during the personalized summarization process.

During the qualitative evaluation in Section IV-D, the average ratings of the summaries generated using the proposed approach were higher than the FB-SUM baseline. One of the reasons that summaries generated using the proposed approach have higher ratings is that they have a short duration. The summary generated using the proposed LTC-SUM and FB-SUM baseline approaches have similar events. Table 8 lists the similarity ratings provided by the participants for the generated summaries. The proposed LTC-SUM approach can generate summaries according to user interests with a highly computationally efficient mechanism.

Previously, full long-form videos were segmented into small clips duration in the summarization process [2]. This is because extensive computational resources are required to store temporal information while analyzing complete long-form videos. However, the overall computational complexity is increased by adding more processing steps to generate

FIGURE 9. **Illustrations of the frame samples from generated video summaries.**

a summary. The overall computation, communication, and storage efficiencies are improved significantly by analyzing thumbnail containers using the proposed method. It can extract and generate summaries based on user preferences from relevant content (such as events and objects). However, it might not be effective for short-form videos that have multiple and faster scene transitions. It was observed that sometimes some frames in the generated summary are not relevant according to the preferred event(s) – which can be mitigated by adopting the solution suggested in previous research [42].

Currently, this paper focuses on the full long-form videos of three genres – Western, sports, and action. The simplicity and scalability for implementing different configuration devices made it easy to adapt the proposed framework to other genres of video. In addition, it can support privacy-preserving solutions [48] effectively by adapting efficient encryption techniques [49]; it can be adapted to three screen TV solutions [50], [51], [52], [53]. The proposed method can also be adopted in ATSC 3.0 and can use over-the-top (OTT) services to provide a personalized interactive application.

V. CONCLUDING REMARKS

This paper presents a personalized lightweight client-driven LTC-SUM keyshot video summarization framework. The framework is designed for resource-constrained end-user devices to generate personalized summaries using their computational resources while resolving computational and privacy bottlenecks. Instead of using entire video data, which are computationally intensive, the lightweight thumbnail containers are used in the proposed method to generate subsets of the corresponding video. This significantly improves the computational, communication, and storage efficiencies as compared to state-of-the-art summarization approaches. For this purpose, a lightweight 2D CNN model was designed to detect personalized events from thumbnails. Extensive quantitative experiments were conducted on full 18 feature-length videos that demonstrated the superior performance of LTC-SUM compared to several state-of-the-art video summarization approaches using the same computational resources end-user devices. Qualitative results showed that the proposed method outperformed the baseline approach and received higher average ratings without losing significantly important information. It is planned to integrate the proposed method with other streaming protocols as future work.

REFERENCES

[1] U. Cisco. (2018). *Cisco Annual Internet Report (2018–2023) White Paper*. [Online]. Available: https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[2] J. Lei, Q. Luan, X. Song, X. Liu, D. Tao, and M. Song, "Action parsing-driven video summarization based on reinforcement learning," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 7, pp. 2126–2137, Jul. 2019.

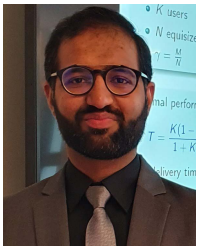
[3] S. S. Thomas, S. Gupta, and V. K. Subramanian, "Context driven optimized perceptual video summarization and retrieval," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 10, pp. 3132–3145, Oct. 2019.

[4] C. Huang and H. Wang, "A novel key-frames selection framework for comprehensive video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 2, pp. 577–589, Feb. 2020.

[5] M. Ma, S. Mei, S. Wan, Z. Wang, D. D. Feng, and M. Bennamoun, "Similarity based block sparse subset selection for video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 10, pp. 3967–3980, Oct. 2021.

[6] M. Wang, R. Hong, G. Li, Z.-J. Zha, S. Yan, and T.-S. Chua, "Event driven web video summarization by tag localization and key-shot identification," *IEEE Trans. Multimedia*, vol. 14, no. 4, pp. 975–985, Aug. 2012.
[7] Y. Song, M. Redi, J. Vallmitjana, and A. Jaimes, "To click or not to click: Automatic selection of beautiful thumbnails from videos," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, New York, NY, USA, 2016, pp. 659–668.
[8] K. Zhou, Y. Qiao, and T. Xiang, "Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2018, pp. 1–8.
[9] J. Fajtl, H. S. Sokeh, V. Argyriou, D. Monekosso, and P. Remagnino, "Summarizing videos with attention," in *Proc. Asian Conf. Comput. Vis.*, 2018, pp. 39–54.
[10] Y. Yuan, T. Mei, P. Cui, and W. Zhu, "Video summarization by learning deep side semantic embedding," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 29, no. 1, pp. 226–237, Nov. 2017.
[11] Z. Ji, K. Xiong, Y. Pang, and X. Li, "Video summarization with attention-based encoder–decoder networks," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 30, no. 6, pp. 1709–1717, Jun. 2020.
[12] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris, and I. Patras, "AC-SUM-GAN: Connecting actor-critic and generative adversarial networks for unsupervised video summarization," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 31, no. 8, pp. 3278–3292, Aug. 2021.
[13] B. Google. (2020). *Shortform and Longform Videos*. [Online]. Available: https://support.google.com/google-ads/answer/2382886
[14] N. Babaguchi, Y. Kawai, T. Ogura, and T. Kitahashi, "Personalized abstraction of broadcasted American football video by highlight selection," *IEEE Trans. Multimedia*, vol. 6, no. 4, pp. 575–586, Aug. 2004.
[15] Y. Wei, S. M. Bhandarkar, and K. Li, "Video personalization in resource-constrained multimedia environments," in *Proc. 15th Int. Conf. Multimedia (MULTIMEDIA)*, New York, NY, USA, 2007, pp. 902–911.
[16] P. Varini, G. Serra, and R. Cucchiara, "Personalized egocentric video summarization of cultural tour on user preferences input," *IEEE Trans. Multimedia*, vol. 19, no. 12, pp. 2832–2845, Dec. 2017.
[17] Y. Li, S. Lee, C.-H. Yeh, and C.-C. J. Kuo, "Techniques for movie content analysis and skimming: Tutorial and overview on video abstraction techniques," *IEEE Signal Process. Mag.*, vol. 23, no. 2, pp. 79–89, Mar. 2006.
[18] M. Ellouze, N. Boujemaa, and A. M. Alimi, "IM(S)2: Interactive movie summarization system," *J. Vis. Commun. Image Represent.*, vol. 21, no. 4, pp. 283–294, May 2010.
[19] W.-T. Peng, W. Chu, C. Chang, C. Chou, W. Huang, W. Chang, and Y. Hung, "Editing by viewing: Automatic home video summarization by viewing behavior analysis," *IEEE Trans. Multimedia*, vol. 13, no. 3, pp. 539–550, Jun. 2011.
[20] X. Chen, Y. Zhang, Q. Ai, H. Xu, J. Yan, and Z. Qin, "Personalized key frame recommendation," in *Proc. 40th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, New York, NY, USA, Aug. 2017, pp. 315–324.
[21] Y. Pan, Y. Chen, Q. Bao, N. Zhang, T. Yao, J. Liu, and T. Mei, "Smart director: An event-driven directing system for live broadcasting," *ACM Trans. Multimedia Comput., Commun., Appl.*, vol. 17, no. 4, pp. 1–18, Nov. 2021.
[22] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning spatiotemporal features with 3D convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vis. (ICCV)*, Dec. 2015, pp. 4489–4497.
[23] Q. V. Le, W. Y. Zou, S. Y. Yeung, and A. Y. Ng, "Learning hierarchical invariant spatio-temporal features for action recognition with independent subspace analysis," in *Proc. CVPR*, Jun. 2011, pp. 3361–3368.
[24] K. Simonyan and A. Zisserman, "Two-stream convolutional networks for action recognition in videos," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, Cambridge, MA, USA, vol. 1, 2014, pp. 568–576.
[25] C. Feichtenhofer, A. Pinz, and R. P. Wildes, "Spatiotemporal multiplier networks for video action recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 7445–7454.
[26] B. Pan, W. Lin, X. Fang, C. Huang, B. Zhou, and C. Lu, "Recurrent residual module for fast inference in videos," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 1536–1545.
[27] M. O'Leary, "IIS IIS IIS and modsecurity," in *Cyber Operations*. Springer, 2019, pp. 789–819.
[28] FFmpeg. (2020). *FFmpeg Github Page*. [Online]. Available: https://github.com/FFmpeg/FFmpeg
[29] R. Hopkins, "Digital terrestrial HDTV for North America: The grand alliance HDTV system," *IEEE Trans. Consum. Electron.*, vol. 40, no. 3, pp. 185–198, Aug. 1994.
[30] T. Amert, N. Otterness, M. Yang, J. H. Anderson, and F. D. Smith, "GPU scheduling on the NVIDIA TX2: Hidden details revealed," in *Proc. IEEE Real-Time Syst. Symp. (RTSS)*, Dec. 2017, pp. 104–115.
[31] C. Mueller, S. Lederer, C. Timmerer, and H. Hellwagner, "Dynamic adaptive streaming over HTTP/2.0," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2013, pp. 1–6.
[32] R. Zurawski, "The hypertext transfer protocol and uniform resource identifier," in *The Industrial Information Technology Handbook*. CRC Press, 2004, pp. 456–478.
[33] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, Jun. 2014, pp. 512–519.
[34] M. Tan and Q. Le, "EfficientNet: Rethinking model scaling for convolutional neural networks," in *Proc. 36th Int. Conf. Mach. Learn.*, 2019, pp. 6105–6114.
[35] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "ImageNet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2009, pp. 248–255.
[36] C.-W. Xie, H.-Y. Zhou, and J. Wu, "Vortex pooling: Improving context representation in semantic segmentation," 2018, *arXiv:1804.06242*.
[37] K. Soomro, A. Roshan Zamir, and M. Shah, "UCF101: A dataset of 101 human actions classes from videos in the wild," 2012, *arXiv:1212.0402*.
[38] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Commun. ACM*, vol. 60, no. 6, pp. 84–90, May 2017.
[39] I. Loshchilov and F. Hutter, "Decoupled weight decay regularization," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–19.
[40] Video-Dev. (2020). *HLS.JS Github Page*. [Online]. Available: https://github.com/video-dev/hls.js/
[41] M. Gygli, H. Grabner, H. Riemenschneider, and L. Van Gool, "Creating summaries from user videos," in *Proc. Eur. Conf. Comput. Vis.* Cham, Switzerland: Springer, 2014, pp. 505–520.
[42] G. Mujtaba and E.-S. Ryu, "Client-driven personalized trailer framework using thumbnail containers," *IEEE Access*, vol. 8, pp. 60417–60427, 2020.
[43] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei, "Large-scale video classification with convolutional neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 1725–1732.
[44] O. V. R. Murthy and R. Goecke, "Ordered trajectories for human action recognition with large number of classes," *Image Vis. Comput.*, vol. 42, pp. 22–34, Oct. 2015.
[45] Y. Shu, Y. Shi, Y. Wang, Y. Zou, Q. Yuan, and Y. Tian, "ODN: Opening the deep network for open-set action recognition," in *Proc. IEEE Int. Conf. Multimedia Expo (ICME)*, Jul. 2018, pp. 1–6.
[46] G. Mujtaba, J. Choi, and E.-S. Ryu, "Client-driven lightweight method to generate artistic media for feature-length sports videos," in *Proc. 19th Int. Conf. Signal Process. Multimedia Appl.*, Lisbon, Portugal, 2022, pp. 102–111.
[47] A.-A. Liu, Y.-T. Su, W.-Z. Nie, and M. Kankanhalli, "Hierarchical clustering multi-task learning for joint human action grouping and recognition," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 1, pp. 102–114, Jan. 2017.
[48] C. Newell and L. Miller, "Design and evaluation of a client-side recommender system," in *Proc. 7th ACM Conf. Recommender Syst.*, New York, NY, USA, Oct. 2013, pp. 473–474.
[49] G. Mujtaba, M. Tahir, and M. H. Soomro, "Energy efficient data encryption techniques in smartphones," *Wireless Pers. Commun.*, vol. 106, no. 4, pp. 2023–2035, Jun. 2019.
[50] G. Mujtaba and E.-S. Ryu, "Human character-oriented animated GIF generation framework," in *Proc. Mohammad Ali Jinnah Univ. Int. Conf. Comput. (MAJICC)*, Jul. 2021, pp. 1–6.
[51] G. Mujtaba, S. Lee, J. Kim, and E.-S. Ryu, "Client-driven animated gif generation framework using an acoustic feature," *Multimedia Tools Appl.*, vol. 80, pp. 35923–35940, Feb. 2021.
[52] E.-S. Ryu and N. Jayant, "Home gateway for three-screen TV using H.264 SVC and raptor FEC," *IEEE Trans. Consum. Electron.*, vol. 57, no. 4, pp. 1652–1660, Nov. 2011.
[53] E.-S. Ryu and C. Yoo, "Towards building large scale live media streaming framework for a U-city," *Multimedia Tools Appl.*, vol. 37, no. 3, pp. 319–338, May 2008.

GHULAM MUJTABA received the B.S. degree in computer science from the COMSATS Institute of Information and Technology, Pakistan, in 2013, the M.S. degree in computer science from Indus University, Pakistan, in 2016, and the Ph.D. degree in computer engineering from Gachon University, South Korea, in 2021. During his Ph.D. degree, he also worked as a Researcher at Sungkyunkwan University (SKKU), Seoul, South Korea. His B.S. degree was funded by the ICT Research and Development Scholarship by the Ministry of IT, Pakistan. He has gained vast academic and professional experience in numerous organizations. His research interests include computer vision, multimedia communications, and mobile computing.



ADEEL MALIK received the B.S. degree in electrical (telecommunication) engineering from the COMSATS Institute of Information and Technology, Pakistan, in 2013, the M.Sc. degree in computer science and engineering from Dankook University, South Korea, in 2018, and the Ph.D. degree under the supervision of Prof. Petros Elia. He received his Ph.D. degree while working at EURECOM's Duality Project. From 2014 to 2016, he worked as a Research Assistant with Dr. Jalaluddin Qureshi on Namal College-funded research projects focusing on the construction of wireless transmission protocols. His research interests include content-centric wireless networks, computer vision, and multimedia communication.

EUN-SEOK RYU (Senior Member, IEEE) received the B.S., M.S., and Ph.D. degrees in computer science from Korea University, Seoul, South Korea, in 1999, 2001, and 2008, respectively. He is currently an Associate Professor at the Department of Computer Science Education, Sungkyunkwan University (SKKU), Seoul. Prior to joining the Sungkyunkwan University, in 2019, he was an Assistant Professor at the Department of Computer Engineering, Gachon University, Seongnam, South Korea, from March 2015 to August 2019. He was also a Principal Engineer at Samsung Electronics, Suwon, South Korea, where he led a Multimedia Team. He was a Staff Engineer at InterDigital Labs, San Diego, CA, USA, from January 2011 to February 2014, where he researched and contributed to next generation video coding standards, such as HEVC and SHVC. From September 2008 to December 2010, he was a Postdoctoral Research Fellow at the Georgia Centers for Advanced Telecommunications Technology (GCATT), School of Electrical and Computer Engineering, Georgia Institute of Technology, Atlanta, GA, USA. In 2008, he was a Research Professor at the Research Institute for Information and Communication Technology, Korea University. His research interests include multimedia communications, including video source coding and wireless mobile systems.