Lightweight Client-driven Personalized Multimedia Framework for Next Generation Streaming Platforms

Ghulam Mujtaba

Advisor: Prof. Jaehyuk Choi

Co.Advisor: Prof. Eun-Seok Ryu

Department of IT Convergence Engineering
Gachon University

A dissertation submitted to the department of Computer Engineering and the committee on graduate studies of Gachon University in partial fulfillment of the requirements for the degree of Doctor of Philosophy

IT Convergence Engineering June 2021

Lightweight Client-driven Personalized Multimedia Framework for Next Generation Streaming Platforms

Ghulam Mujtaba

Streaming platforms employ centralized server-based personalized multimedia content generation technologies to overcome information overloads and browse an ever-growing video collection on the internet. The advantage of these techniques is that all information is co-located, such as user data, video content, and functionalities. However, the lack of transparency in managing and processing users' data has increased the demand for conversational privacy protection technologies. Additionally, there are still some incurable issues that need to be handled by server-side techniques such as (i) handling of secondary/companion devices, (ii) real-time user emotion collection, and (iii) huge computational load to process all the information in real-time. These issues inspire us to overcome and discover the feasibility of a client-driven approach that can create numerous personalized multimedia content while reducing privacy and computational bottlenecks on resource-constrained end-user devices.

This dissertation presents a client-driven personalized multimedia content generation framework for streaming platforms. The proposed framework can generate several personalized multimedia contents for feature-length videos on the end-user devices, such as movie trailers, animated GIFs, and video summaries simultaneously. The state-of-the-art methods that acquire and process entire video data to generate personalized multimedia are highly computationally intensive. In this regard, the proposed framework uses lightweight thumbnail containers to handle the complex process of detecting events parallelly resolving computational and privacy bottlenecks on the resource-constrained end-user devices. This significantly reduces computational complexity and improves communication (between server and client) and storage efficiency. These improvements are achieved by extracting features from thumbnails, which helped select and retrieve just a handful of specific segments. In this context, the 2D CNN model is designed that can extract features from thumbnails. The framework is designed to manage a wide range of end-user hardware platforms with heterogeneous computing, networks, and storage capabilities.

To validate the proposed framework, this dissertation designs and implements three different client-driven techniques for streaming platforms. The first proposed method is designed to facilitate and expedite personalized trailers generation in the film-making process. The second proposed method is designed to generate personalized animated GIFs for full-length sports videos. Finally, the third proposed method is designed to produce personalized keyshot-based video summaries of different genres and diverse full feature-length videos such as documentaries, movies, and sports matches. Extensive quantitative experiments show that the proposed methods are significant computationally efficient than the state-of-the-art methods on similar client device specifications. To the best of our knowledge, all three proposed methods are the first-ever client-driven approaches for streaming

platforms that analyze thumbnail containers to create personalized multimedia content such as a trailer, animated GIF, and video summary.

Keywords: personalize media, multimedia content, movie trailer, animated GIF, video summary, thumbnail containers, client-driven, feature-length videos, streaming platforms, OTT media service, ATSC 3.0, 2D CNN.

To my parents,
my siblings,
and friends
for their lifelong love,
encouragement and sacrifices.

~

To my lovely wife,
for her endless patience,
and support

~

To my son Muhammad Raza
for bringing joy
to our lives.

DECLARATION

I hereby declare that except where specific reference is made to the work of others, the contents of this dissertation are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This dissertation is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This dissertation contains fewer than 25,000 words including appendices, bibliography, footnotes, tables and equations and has fewer than 30 figures.

Ghulam Mujtaba
June 2021

ACKNOWLEDGEMENTS

All praises are due to Allah, the only worthy of worship, the most gracious, and the most merciful. I would desire to express my sincere gratitude to my supervisors, Prof. Jaehyuk Choi and Prof. Eun-Seok Ryu, for allowing me to obtain my Ph.D. in their research group. I am thankful for their continuous support throughout my Ph.D. study for their patience, motivation, and encouraging attitude. I feel very fortunate and blessed to research under the supervision of marvelous human beings on earth with beautiful souls. Additionally, I would like to thank my dissertation evaluation committee members Prof. Yong Ju Jung, Prof. Jungchan Cho, and Prof. Jaekwang Kim, for providing insightful comments and valuable suggestions for improving the quality of this dissertation. I appreciate the support and encouragement given by current and former MCSL lab members during this endeavor. Their presence certainly made the journey much more comfortable and exciting.

I am fortunate to have friends in Gachon, SKKU, COMSATS Lahore, ICTIAN (BNA), and college, for always being around in good and tough times. I am deeply grateful to my siblings, my parents, and my in-laws, all of whom provided endless support throughout different stages of my life. Finally, I convey my heartiest gratitude to my wife and son Muhammad Raza for their unconditional love, patience, understanding in my tough times, and always being my absolute strength. Without their support and encouragement, I would not have achieved my goals.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

List of Abbreviations

ATSC	Advanced Television Systems Committee
CMAF	Common Media Application Format
CNN	Convolutional Neural Network
CTR	Click-through Rate
DASH	Dynamic Adaptive Streaming over HTTP
FPS	Frame Per Second
GIF	Graphics Interchange Format
HCR	High Computational Resources
HLS	HTTP Live Streaming
LCR	Low Computational Resources
LSTM	Long Short-Term Memory
OTT	Over-the-top
RNN	Recurrent Neural Network
SGD	Stochastic Gradient Descent
VoD	Video on Demand
VSP	Video Service Provider

CHAPTER 1

INTRODUCTION

This Chapter provides the background of this dissertation, urgency for new methods to generate personalized multimedia content, addressing challenges, dissertation research goals, dissertation contributions, and the outline.

1.1 Background

Every day, new information is appearing on the internet in various applications and domains. Due to innumerable data and time constraints, it is often challenging for users to find relevant content. Before the internet era, people had to ask their friends and relatives for suggestions on various items such as shopping, games, and movies. However, this method of consultation carries the risk of making poor decisions. Personalized recommended algorithms are adopted in various domains and applications to alleviate this problem and reduce overall information overload.

In today's online world scenarios, personalized recommendation algorithms are an influential part of the user experience and decision-making. Several video service providers (VSPs) use various server-driven recommendation algorithms to suggest the most relevant content according to the user's preferences, as depicted in Figure 1.1. However, it is sometimes very challenging for many users to find the most relevant media content from recommended data. To overcome this problem in recent years, VSPs are focusing on server-based personalized multimedia content generation methods based on the user's browsing history and personal data [1]. The trade-off for this rapid development comes in the cost of processing user data, which is being collected on an unprecedented scale by streaming platforms.

The centralized server-driven personalized multimedia generation techniques are the most prominent in VSPs. The advantage of this is that all the information is co-located, such as users' data, video content, and functionalities. However, the lack of transparency in collecting and processing users' data has increased the demand for conversational privacy protection technologies. Besides, server-driven algorithms can be used to explicitly control and influence users' behaviors and opinions, which may create divisions between nations based on their opinion [2]. These are some of the factors

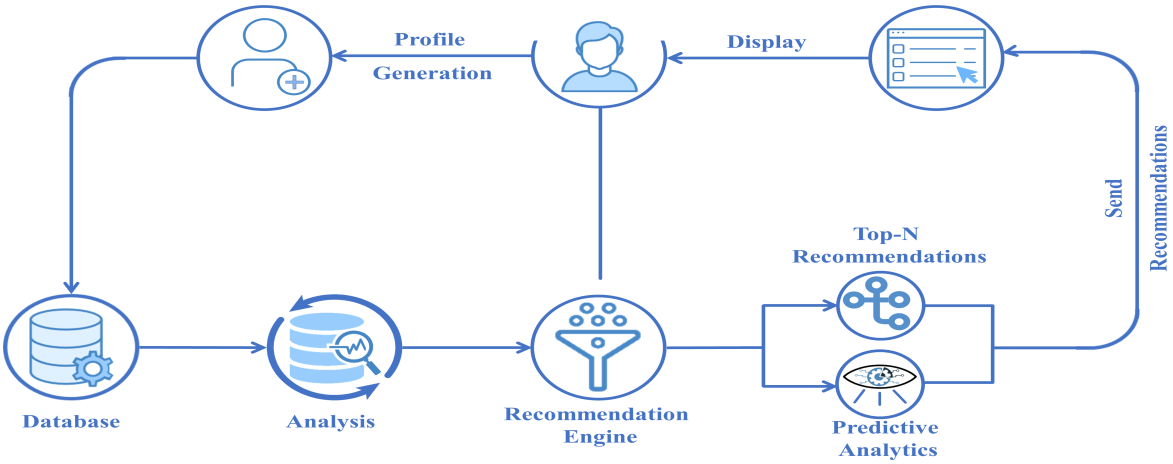

Fig. 1.1 Conceptual architecture of server-driven personalized recommendation algorithm system.

we considered while designing a conversational personalized multimedia generation framework. This will allow users to control and extract data according to their current mood and interests.

1.2 Motivation

The field of recommendation algorithm applications has exploded over the last decade, making personalized recommendations ubiquitous. During this time, remarkable developments have been made in the industry and research community to design various media-recommended applications. Most enterprise companies like Netflix, Google, Amazon; use several recommendation algorithms within their services to grow their business in response to user demands. The application domain is the first factor to scrutinize when designing recommendation algorithms. It is because the application domain has a significant impact on the algorithmic approach. Based on a particular application domain, some of the popular recommendation system applications fall into five categories, as shown in Figure 1.2. The core business of several companies now entirely depends on their unique recommendation algorithms. Each has different challenges in its diverse applications and domains.

According to a recent survey, adults in the United States watch movies for nearly six hours each day [3]. It makes entertainment applications with recommended algorithms one of the most promising domains for research. The new advanced methods in the industry are under consideration to support rapid innovation and provide the best user-personalized experience. We analyzed patent applications between 1997 and 2018. The leading target technologies used by most well-known companies are deep learning-based personalized recommendations, scene understanding & annotation, viewer emotion recognition, media information, video clip extraction, 5G/ATSC 3.0/TV, and client-driven technology. Figure 1.3 depicts the leading companies currently considering these technologies in the entertainment application domain to design personalized multimedia recommendation algorithms.

In a real-time application scenario, the viewer may have different interests even for the same video. Therefore, personalized multimedia generation techniques have become significantly important.

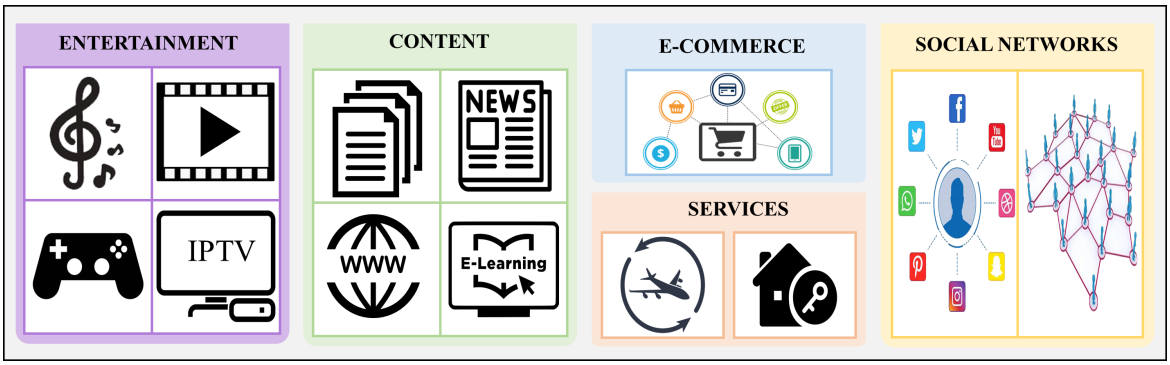

Fig. 1.2 The most common personalized recommendation applications in five domains.

Real-time user emotions, preferences, and companion device processing are a few influential elements of future internet-based video streaming services to generate media content. Meanwhile, server-driven methods are the most common solutions in the entertainment industry. It can be demanding to keep a server-side solution responsive in real-time with limited computational resources while serving numerous concurrent users. There is great urgency to design new lightweight personalized multimedia generation technologies to overcome all these challenges and support rapid innovation and demand. The following section summarizes the objectives and challenges based on reviewed target technologies for streaming platforms.

1.3 Challenges and Objectives

Click-through rate (CTR) is an influential metric for newly broadcast videos on streaming platforms. There is a strong correlation between customized multimedia content (i.e., trailers, artwork, summaries, and animated GIFs) and personalization, which results in higher CTR [4]. However, multimedia content (trailer) is currently being generated in the universal structure on streaming platforms [1]. Users may not like a particular trailer if it does not match their interests. This can cause users to

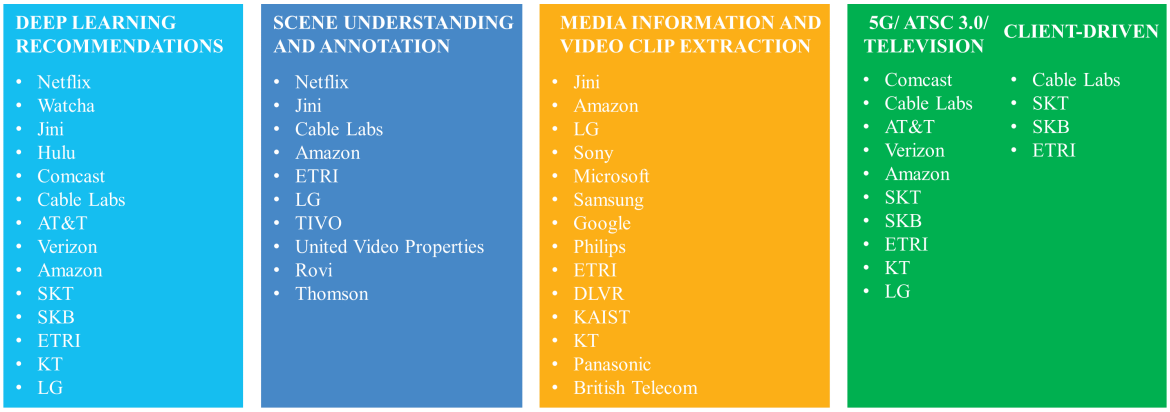

Fig. 1.3 Leading companies that are developing targeted technologies.

skip videos, which can significantly reduce the CTR of the corresponding video, leading to financial damage. Due to the recent popularity of personalized multimedia content on streaming platforms, there is enormous demand for new methods that can generate multimedia content based on user preference while using minimal computing resources.

The centralized server-driven techniques are under consideration to automate personalized multimedia content generation to increase the CTR for the corresponding video. The advantage of a centralized server-driven solution is that all information is in the same place including, users' data, video content, and functionalities. However, numerous challenges persist regarding the practical implementation of server-driven solutions such as (i) user privacy, (ii) handling of secondary/companion devices, (iii) real-time user emotion collection, and (iv) huge computational load to process all the information in real-time. There are still some incurable issues that need to be handled by server-driven solutions.

- Generally, a video contains miscellaneous information such as character appearance, motion, interactions between objects, events, and scenes. Considering that the Frames Per Second (FPS) of an hour video is 25, the video has thousands of frames. To properly process the entire video data, extensive computational resources will require using existing approaches [5–7]. As the length of the video increases, the demand for computational resources will also increase. With limited computational resources constrained, this is not a viable approach to process the entire video data, which will increase the overall computational time.

- Previously, the full feature-length videos are segmented into small clips in the personalized media generation process [8]. This is because extensive computational resources are needed to store temporal information while analyzing complete feature-length videos. However, the overall computational complexity increased by adding more processing steps to generate a summary. Most of the client devices have limited computational resources.

- Video summary is one of the popular multimedia content in streaming platforms. A video summary provides an instant preview of full feature-length videos such as movies and documentaries. However, a personalized video summary with the optimal length maybe not immediately available to all users. The generation of a real-time personalized video summary with current approaches will require enormous computational resources [9, 10], even in server-driven techniques to process user data and the entire video. Besides, user data is collected on an unprecedented scale during the model training process, and there are serious concerns related to user privacy in server-driven solutions.

- The flow of user data between end-user devices and servers can grow enormously as the number of users and their interactions grow. Server-driven solutions can be burdensome to provide real-time responses with limited computational capabilities to an overwhelming number of concurrent users while processing all data.

These are the main aspects that motivated us to design an innovative lightweight framework that can support a wide range of computational end-user devices. Besides, it can overcome privacy and computational bottlenecks for resource-constrained devices during the personalized multimedia content generation process.

1.4 Research Goals

New methods, services, and features are constantly transforming to facilitate the personalized recommendation process. Meanwhile, lightweight personalized multimedia content generation techniques are nascent. More effective approaches need to be designed to bridge the semantic gap between video understanding and personalization. This dissertation explores the feasibility of a lightweight framework that can support and generate several personalized multimedia contents simultaneously.

Most well-known companies use server-driven techniques to provide personalized recommendations and generate customized media according to users' interests. However, user privacy is a leading concern in those techniques because they store and process personal data on servers. This dissertation discovers the feasibility of a client-driven approach that can enable users to manage their sensitive personal data while generating personalized multimedia content.

Generally, personalized keyshot-based video summarization methods are used to produce subsets for short-form videos (i.e., user-generated TikTok and news) or long-form videos (i.e., feature-length films and soccer matches). Generally, the length of short-form and long-form videos are under 10 minutes and over 10 minutes, respectively [11]. The playback duration of short-form videos is already concise, so it is impractical and may be ineffective to generate keyshot-based subsets for such videos. Meanwhile, the playback duration for long-form videos can exceed 90 minutes, particularly for movies and sports genre videos. Keyshot-based summarization is more practical and effective for such video categories to give users a quick glimpse. This dissertation explores a new technique to personalize media for long-form videos.

Traditionally, the video is a combination of continuously moving frames that operate at 25 FPS. Intuitively, there is a significant amount of redundancy in the frame for any unit second. Considering that computational resources are limited and expensive. So processing highly correlated frames will waste a considerable portion of computational resources which is not feasible in real-time scenarios. This dissertation explores a new, innovative, lightweight framework that can utilize minimal computational resources of end-user devices during the personalized multimedia content generation process.

Generating personalized multimedia content requires extensive computing resources. Meanwhile, end-user devices have limited computational resources. This dissertation explores a new method to generate personalized multimedia content for resource-constrained devices while reducing computational bottlenecks.

Most streaming platforms manage video, audio, and other associated content individually for every corresponding video. Videos are divided and saved in small continuous segments. This dissertation

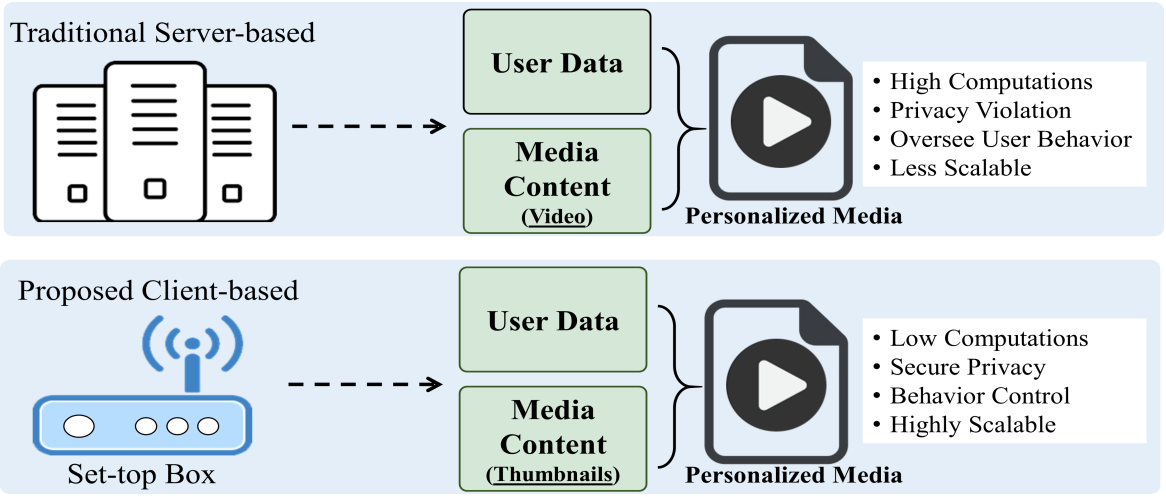

Fig. 1.4 Traditionally, personalized multimedia content is generated while analyzing video data using server-based techniques. This dissertation proposes an innovative client-driven framework that uses lightweight thumbnail containers in the personalized multimedia content generation process while reducing privacy and computational bottlenecks for resource-constrained devices.

explores an efficient method that requires less network bandwidth and local storage during the personalized multimedia content generation process comparing with previous approaches. Figure 1.4 shows a general overview of the traditional and the proposed personalized multimedia content generation methods.

1.5 Dissertation Contributions

Generating personalized multimedia content while analyzing entire video content requires enormous computational resources. Therefore, video content (trailers) is produced using the one-size-fits-all framework. There is extreme urgency for effective techniques to reduce the computational complexity and bridge the semantic gap between personalization and video understanding. This dissertation contributes to the personalized multimedia generation area to reduce that gap. The essential points are as follows:

- This dissertation proposes a new lightweight client-driven framework that can simultaneously generate numerous personalized multimedia content such as video summaries, animated GIFs, and movie trailers. It handles the complex process of detecting personalized events (such as penalty shoot-outs of soccer videos) from lightweight thumbnails. The proposed framework uses the computing resources of the client device in the entire process. The framework is invented to manage a large number of client devices having heterogeneous computational resources. Figure 1.5 shows a conceptual diagram of the proposed framework. To the best of our knowledge, this is the first client-driven approach to create personalized multimedia content using thumbnail containers for streaming platforms.

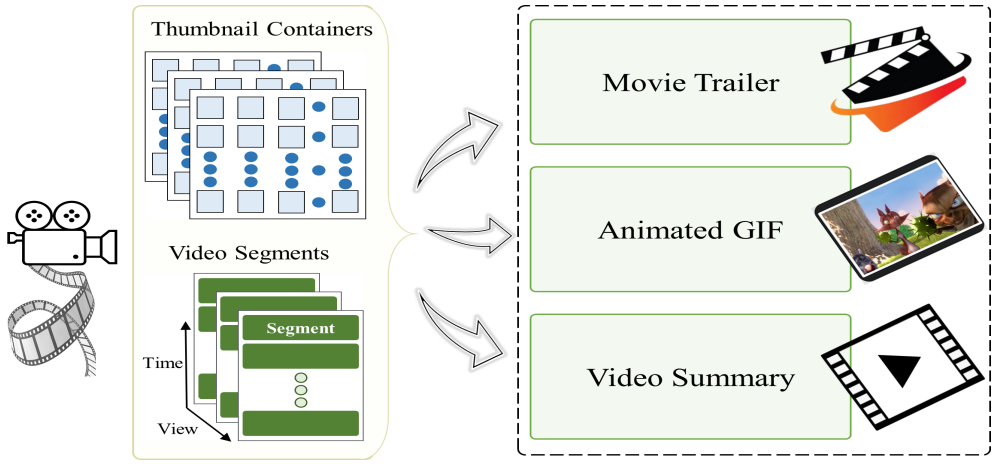

Fig. 1.5 This dissertation presents a client-driven personalized multimedia content generation framework for streaming platforms. The proposed method can simultaneously generate diverse personalized multimedia content, such as movie trailers, animated GIFs, and video summaries for concurrent users. Instead of processing the entire video data, it analyzes lightweight thumbnail containers throughout the process that reduces privacy and computational bottlenecks for resource-constrained end-user devices. Additionally, it reduces the overall computational complexity and makes the proposed approach highly efficient in terms of computation, communication, and storage.

- In this dissertation, we designed new 2D CNN models that can analyze lightweight thumbnails and detect personalized events. Quantitative results show that designed models outperformed previous 2D CNN models.

- Normally, a handful of trailers are produced in the film-making process because they require a high level of cognitive effort and cost. This dissertation proposes a new framework to alleviates this problem by facilitating a personalized trailer creation procedure [12]. It handles the sophisticated operation of recognizing personalized events from lightweight thumbnails in real-time. Twenty-five western and sports genre films are analyzed to test the efficiency of the proposed framework. The official trailers are then analyzed in comparison to the ones created by the proposed framework.

- Animated GIFs are ubiquitous on streaming platforms due to their small size, short length, and storytelling nature (no audio) [13]. They play a vital and decisive role in the video selection process. Its ubiquitous adoption and use have increased the demand for the design of lightweight, personalized GIF generation methods. The innovative client-driven framework is proposed in this dissertation to advance the research of GIF generation. The proposed framework uses thumbnail containers and video segments instead of all frames and videos to analyze personalized events and generate animated GIF. It allows the creation of animated GIFs within suitable computation time for resource-constrained devices such as the embedded AI computing device Nvidia Jetson TX2.

- Personalized video summarization techniques produce a short version of the full feature-length video that conveys the meaningful segments according to the users' preferences. Based on this, viewers can quickly get an overview of the entire story without watching the full video. However, this feature is not applicable to newly broadcast videos due to the enormous demand for computational resources and user data. This dissertation proposes a new lightweight client-driven framework that is computationally efficient and requires minimal computational resources to create personalized video summaries in real-time. Instead of obtaining and processing the entire video, it uses lightweight thumbnail containers and segments to generate a personalized summary for concurrent users. The proposed method is efficient because the whole video is not processed, stored, and nor transmitted over the network during the process.

1.6 Notations and Definitions

The following definitions are used throughout this dissertation:

- **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.
- **Event/Action:** Event/action corresponds to certain types of activity, such as penalty shoot-out in a football match or horse-riding in western movies.
- **Thumbnail Container:** $ThmCon$ A thumbnail container $ThmCon$ is a collection of thumbnails extracted from the video. The sequence of all $ThmCon$ covers the entire video length.
- **Thumbnail:** $Thum$ A thumbnail $Thum$ is obtained from the $ThmCon$. A single $ThmCon$ has 25 $Thum$, which is used in the video player to instantaneously preview the video. The number of $Thum$ in a $ThmCon$ can be varied. However, the number of $Thum$s is fixed to 25 in this study, which is based on our study on web-based YouTube player.

1.7 Outline

This dissertation is organized as follows: **Chapter 2** reviews the most relevant existing techniques that are relevant to this research. **Chapter 3** introduces universal and key modules of the proposed framework for a personalized multimedia content generation along with the hardware configuration of the devices used for quantitative experimental evaluations. **Chapter 4** presents an innovative framework to facilitate personalize movie trailer generation process. **Chapter 5** introduces the lightweight client-driven framework for the personalized GIF generation. **Chapter 6** introduces personalized lightweight client-driven keyshot video summarization framework for long-form videos such as movies and documentaries. Finally, **Chapter 7** concludes this dissertation by providing an overview of the results achieved from the three proposed frameworks and insights into research challenges that remain to be addressed.

CHAPTER 2

RELATED WORK

Over the last decade, numerous personalized multimedia content generation approaches have been introduced to mitigate information overload issues and increase CTR. Server-based personalization multimedia solutions are adopted that can generate media according to the users' requirements and their previous activities with the system effectively and efficiently. However, breaches of user privacy and the enormous demand for computational resources are one of the biggest concerns in server-driven personalization approaches.

This Chapter first reviews video understanding methods using deep learning approaches. Next, existing event and action recognition methods from videos and images are investigated. Video summarization methods are then reviewed. The following describes the keyframe and keyshot summarization approaches. Later, animated GIF generation approaches are analyzed. Finally, we reviewed related work to summary and trailer generation methods on movies and documentaries, respectively.

2.1 Video Understanding Methods

Understanding videos is one of the most prominent areas of computer vision research. In the context of the personalized media generation process, detecting shots or events from videos according to user preferences is a crucial and challenging task. Therefore, analyzing and identifing personalized events from thumbnails is a critical phase of the proposed multimedia generation method. Here, some prominent SoA CNN techniques are reviewed. The CNN models have surpassed traditional approaches in recent works [14–17]. This is because they are very capable and generalized in extracting the overall features compared to the handmade features. For this, different variants of 2D CNNs and three-dimensional CNNs (3D CNNs) have been used to analyze pictures. Only spatial operations can use used on a single picture using 2D CNNs. However, temporal operations along with maintaining temporal dependencies between the input video frames can be performed using 3D CNNs [14]. In [15], the 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, a CNN network called C3D was used to extract later-fed video features to SVM to identify the action [14]. Unlike previous methods, another CNN-based SoA event detection approach proposed that used two variants

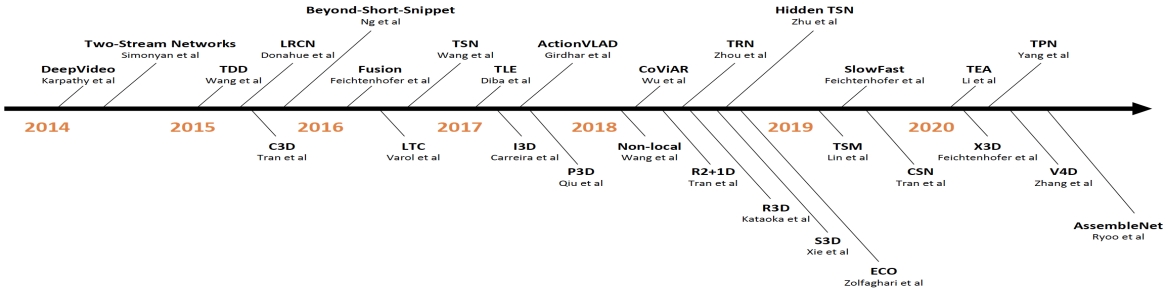

Fig. 2.1 A chronological overview of recent SoA representative work in video action/event recognition.

of the stream: the spatial stream and the temporal stream [17]. The video is decomposed into spatial (RGB representation) and temporal (optical flow representation) components. Later, the frames are fed to two different 3D CNNs.

2.2 Events Recognition Methods

Event recognition is a common problem in recognizing and categorizing video segments as per the predefined set of actions or activity classes utilized to understand videos. Figure 2.1 depicts a chronological overview of recent representative work in video understanding [18]. Most methods adopt temporal segments to prune and classify videos [19]. Recent researches have focused on leveraging context to enhance and improve event recognition approaches. Context represents and utilizes both spatiotemporal information and attention; this helps learn adaptive confidence scores to utilize surrounding information [20]. The more advanced methods for temporal integration are frequently using recurrent neural networks (RNNs) and long short-term memory (LSTM) neural networks to solve motion-aware sequence in learning and activity recognition problems [21, 22]. In this context, a convolutional LSTM network with attention-based mechanisms is proposed to support multiple convolutional kernels and layers [21]. Attention models have also been used to improve integrated spatiotemporal information. Recent studies have used two model-based attention mechanisms within this category of methods [23]. The first is a spatial-level attention model that determines important areas within a frame, and the second addresses the time-level attention to identify frames in a video.

2.3 Video Summarization Methods

In recent years, researchers have introduced several techniques for summarizing movies. These can be broadly classified into two techniques: *generalize* summaries [24–30] and *personalized* summaries [31–34]. The generalized techniques are referred to as approaches that generate the same summary for all users; in contrast, the personalized techniques generate distinct summaries for each user corresponding to their preference. Generalized summarization techniques do not use

user preferences. Instead, they rely on several clues such as subtitles, scripts, and movie formats with visual and audio features. For example, in [24], documentary videos were summarized using concept-expansion trees to create a relational graph to describe semantic concepts. Another study [25] used a liner combination to create summaries; four perceptive models were combined according to various cues, including contrast, statistical rhythm, motion, and key scenes. In contrast, the perspective of relationships between the characters in the movie was analyzed rather than the audio-visual features in [26]. Similarly, in [27], researchers used acoustic segmentation, including the maximum posteriori probability approach, to identify key characters in the movie. This method can be used to index and retrieve specific character shots as well as create summaries. In [28], researchers created a movie summary that only contained lead characters. They used face clustering to obtain these characters based on their appearance in the movie. The multimodal saliency-based movie summarization scheme was proposed in [29]. The researchers extracted features from three distinct approaches and combined them into a multimodal bend curve structure. Six different textual summarization algorithms were applied to scripts and subtitles of different genres to create summaries of a movie or documentary [30]. An important contribution of their work is identifying the best algorithm for the specific genre of a movie or documentary.

The personalized summarization techniques utilize users' preference-based events, shots, and features to create summaries. The approach in [31] used short and long-term audio-visual temporal features to detect sub-stories from movies and to generate a summary whose length could be adjusted according to user preference. The technique proposed in [32] allows users to select content, type of different shots, and summary length, and then it generates a personalized movie summary. However, movie content and user preferences are equated at the feature stage instead of the semantic level. The emotion of the viewer and their attention was used to create a summary [33]. The mood of the viewer was identified by their different facial expressions, blinking, and head and eye movements while watching a video. A summarization technique that relies on user-generated comments was proposed in [34]. They used real-time comments created by the audience on the timestamps of the movie. The number of comments showed the excitement of the audience, and the content of the comments gave the idea of the current scene.

2.4 Keyframe and Keyshot Summarization Methods

Contrary to summarization techniques discussed in Section 2.3 for the movies and documentaries, several proposed techniques mainly focus on the short and lengthy videos. The methods are categorized as keyframe [8, 35, 6, 36] and keyshot [5, 7, 9, 37, 38, 10]. In [8] scripts and subtitles of the videos were mined and later used as semantic information to generate a video summary. The Sequential Multi-Instance Learning (SMIL) model was trained using synchronized subtitles and the script was used to find actions in a video. The model uses an action analysis model to cut the video into clips containing a single action, and for each clip, the model selects frames that are distinguishable compared to other actions. Reinforcement learning is used to sequentially extract a fixed number

of keyframes using a time-dependent location, integrate information about the keyframes using a recurrent neural network, and categorize the video. Perceptual video summary, video browsing, and video search methods proposed in [35] by extracting low-level features of videos. Gestalt principal cost function is used to obtain and identify perceptual features such as motion, color, shape, and size while selecting a frame to summarize the video. Instead of video, summarized keyframes are used to retrieve and index the video. The automatic video summarization method is proposed in [6] by extracting spatio-temporal information and inter-frame motion curves using a capsule network. The self-attention model is used to select keyframes in a video sequence and aggregate all those keyframes to generate a video summary. Attention mechanisms are used to focus on the information in the sequence and find important points. All shots are automatically taken from the video and the highest point of the attention curve in the keyframe as a summary image of the shot is selected. The transition detection method is applied in frames using cut, crop cut, dissolve, and fade effects. The kernel blocks sparse subset selection (KBS3) and similarity-based block sparse subset selection (SB2S3) models used by [36] in the summarization process. The KBS3 model analyzes the local content similarity of adjacent frames while SB2S3 analyzes the global similarity relationships between frames. The proposed method can find similarities between frames and can consider one global relationship between all frames. The block sparsity further considers the local relationship of one of the adjacent frames.

The keyframe summarization method gives a quick glimpse of the video as a set of images, but too much valuable information is dropped. The keyshot summarization methods try to overcome this challenge and provide more informative summaries in the form of videos. In [5] researchers use aesthetic scoring to change point detection for segmentation from the video. The authors have used the K-nearest neighbor algorithm for clustering and removing the redundant frames. The video summarization task is formulated as a sequential decision-making process in [7], researchers developed a deep summarization network (DSN) to predict a probability for each video frame. The final summary was generated based on the probability, which indicated how likely a frame was selected. The sequence-to-sequence network is made of a soft self-attention with a 2-layer fully connected network proposed in [9], to process the CNN features of the video frames and compute frame-level importance scores that are used for important fragment selection. A deep side semantic embedding (DSSE) model by leveraging queries as side information method is proposed in [37]. The DSSE architecture consists of two subnetworks, each with a unimodal autoencoder. One DSSE autoencoder encodes the video frame input and the other encodes the side information of the text feature associated with the video. The keyshot summary is generated by minimizing the distance between the selected video frame and the side semantic information in the latent subspace. The smaller the distance, the more relevant it is. A supervised-based encoder-decoder network method is proposed in [38]. It measures the importance of the sequence of video frames and produces a series of keyshots containing humans as output. The encoder uses bidirectional long short-term memory to encode contextual information between input video frames. Meanwhile, the decoder is used for two attention-based LSTM networks to get additive and multiplicative objective functions. The keyshot

selection model aims to convert a frame-level importance score into a shot-level score and generate a summary. In [10], researchers embed an Actor-Critic model into a Generative Adversarial Network (GAN) and select the most important frames from the video. The selected frames are later used to form the video summary.

2.5 Animated GIF Generation Methods

Animated GIFs, first created in 1987, have become widespread in recent years. Specifically, in [13], animated GIFs were reported to be more attractive than other forms of media, including photos and videos, on social media platforms such as Tumblr. They identified some important factors that contribute to the fascination users have with GIFs, such as animations, storytelling capabilities, and emotional expression. In addition, several studies [39, 40] have trained models for predicting viewers' perceptual sentiments toward animated GIFs. Despite the engagement, in [41], it was discovered that viewers may have diverse interpretations of animated GIFs used in communication. They predicted facial expressions, histograms, and aesthetic features and compared them to [40] in order to determine the most appropriate video features for expressing useful emotions in GIFs. Another recent study [42] used sentiment analysis to estimate text and visual emotion scores for annotated GIFs. From an aesthetics perspective, in [5], frames were selected by analyzing various objective and subjective metrics (e.g., visual quality and aesthetics) of video frames to generate GIFs. In a recent study [43], the authors proposed a client-driven method to mitigate privacy issues while designing a lightweight method for streaming platforms to create GIFs. Instead of adopting the entire video in the process, the authors used an acoustic feature to reduce the overall computation time, while using limited computational resources.

2.6 Video Trailer Generation Methods

The trailer can be very cinematic with post-production features such as sophisticated shot transitions, background music, and overlaid text [44]. Because of the trailer's artistic and creative aspects, specified areas that facilitate the process of producing video (i.e., movie) trailers have generally not received the same attention as movie summaries and highlights. The study by [45] is the first to introduce a semi-automated approach to create movie trailers within the human-AI collaboration. In this study, researchers specifically interpreted the several kinds of emotions from horror films and selected the best ten moments from feature films to generate movie trailers. In another movie trailer method [44], authors focus on action genre movies. They specifically investigated the level of visual movement throughout the film by defining and detecting specific voice cues (voice, music, silence, etc.) and selected individual sequences to create trailers. In the context of trailer generation for television programs, researchers focused on identifying the textual correspondence between the text of the show summary provided by the Electronic Program Guide and the closed captions of the original video in [46]. Researchers first created a textual summary to create a trailer.

CHAPTER 3

UNIVERSAL MODULES OF PROPOSED MULTIMEDIA GENERATION FRAMEWORK

The central idea of this dissertation is to propose a client-driven personalized multimedia content generation framework while reducing computational and privacy bottlenecks. It is invented to withstand a different set of end-user resource-constrained devices with distinct computational, storage, and network capabilities. The proposed framework has two main modules: the HLS client and the HLS server. The overall primary focus of the proposed framework is on client-driven applications and implementations. Meanwhile, a cross-platform HLS server is configured locally from an implementation perspective. Figure 3.1 depicts the schematic illustration of the proposed personalized multimedia content creation framework.

This Chapter describes the background of HLS, DASH, and CMAF along with the universal modules of the proposed framework. The universal components are the HLS server, HLS clients, HTPP persistent connection, and web-based user interface. This Chapter also presents the configuration of server and client devices used for all quantitative experimental evaluations.

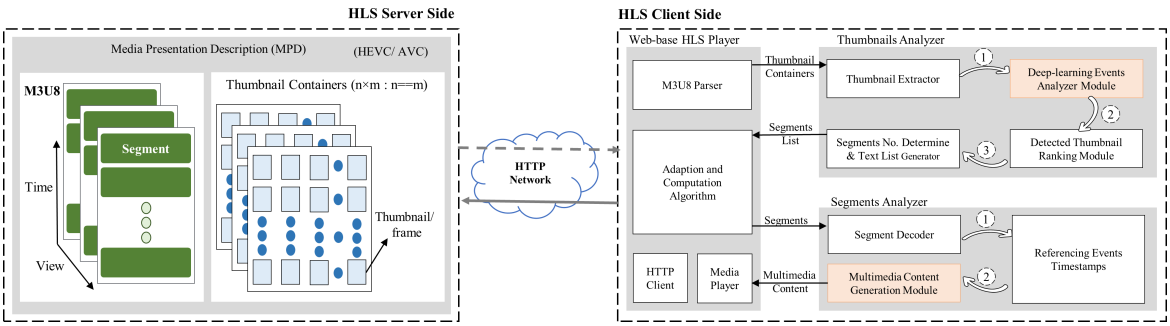

Fig. 3.1 Schematic illustration of the proposed personalized multimedia content creation framework.

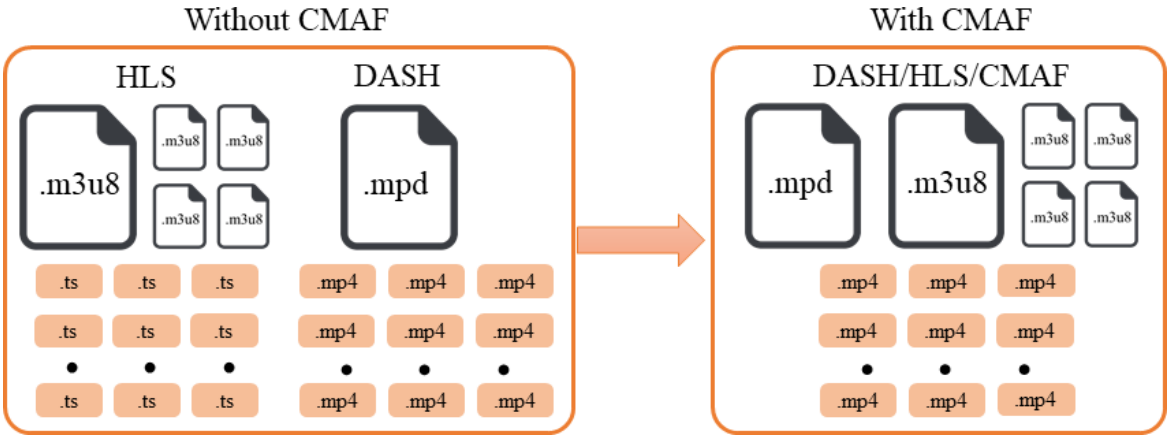

Fig. 3.2 Streaming platforms are employing CMAF to reduce storage, cost, complexity, and latency on servers. CMAF uses only fragmented '.mp4' containers. This cuts the storage on the server in half.

3.1 Background: HLS, DASH and CMAF

The streaming platform services split the video into smaller video segments listed in the playlist file. The video player is responsible for loading the playlist its associated segments, and repeat until the entire video is played. The Master Playlist lists can list different playlists, allowing the player to make dynamic video quality selections and avoid buffering. This allows the video player to identify the ideal bit rate for the current network and switch from one playlist to another.

Today, almost all streaming videos are compressed using standardized encoding and compress technologies. Generally, H.264 or H.265 (HEVC) is used for video and AAC for audio. The data is stored in containers (sometimes called formats) to provide synchronization and general metadata. HTTP Live Streaming (HLS) and Dynamic Adaptive Streaming over HTTP (DASH) are the most commonly used for container transferring. HLS wraps or encapsulates data in MPEG-2 (.ts), and DASH uses MPEG-4 (ISOBMFF) containers. To play each container, the video player needs the same encapsulation format.

OTT services need to store and encode every video according to HLS and DASH protocols. Additionally, there is an additional cost to provide HLS and DASH services to users, increasing the complexity of building and operating these systems. Common Media Application Format (CMAF) aims to solve these problems by converging on an existing single container format for OTT media distribution, rather than creating the already existing container format. Adopting CMAF for video streams significantly reduces cost, complexity, and latency as depicted in Figure 3.2. It also eliminates the investment associated with encoding and storing multiple copies of the same content.

In this dissertation, we use only the HLS streaming protocol. The main reason is each segment of the HLS has its header and can be read/modify separately. Meanwhile, the entire DASH video has only one header, so to read/modify a particular segment, we need to decode/retrieve the video header. By doing so, the complexity of the system will increase.

3.2 HLS Server

In the proposed framework, the first main universal module is the HLS server. This module is invented to support multiple heterogeneous client devices that can download *ThmCon*s and *Seg*s concurrently. For this purpose, Internet Information Services (IIS) is selected and configured locally in Microsoft Windows 10. The predominant purpose of choosing IIS is that, it maintains most network protocols [47]. The videos in IIS are encoded using FFmpeg as H.264/AAC Moving Picture Experts Group 2 (MPEG-2) Transport Stream (.ts) *Seg*s [48]. This reduces potential packet damage and loss in transmission. Every *Seg* simulates approximately 10 seconds of playback time of the original video with consecutive timestamps. Similarly, a list of all corresponding video *Seg*s is saved in a text-based playlist file (M3U8) according to the *Seg* playback arrangement.

The HLS server also contains *ThmCon*s along with the *Seg*s of the corresponding video. These *ThmCon*s are collected individually from the source video using FFmpeg [48]. Every *ThmCon* has 25 *Thum*s, and the first frame of the video per second is selected as the thumbnail. The sequence of 25 *Thum*s produces a single *ThmCon*. *Thum*s are merged into 5×5 according to playtime. A single thumbnail and *ThmCon* depicts the playback time of the corresponding video for 1-second, and 25-seconds, respectively. The entire duration of the source video is covered in sequences of *ThmCon*s. The size of each thumbnail and *ThmCon* is fixed at 160×90 ($width \times height$) pixels and 800×450 ($width \times height$) pixels, respectively. The size of *Thum*s and *ThmCon*s is fixed after analyzing YouTube's web-based player. Figure 3.3 depicts an example of a *ThmCon* of a video obtained in the YouTube web-based video player on an end-user device on the left side and a *Thum* presenting a specific period on the right side. The following section describes the configuration of HLS client components.

Fig. 3.3 The sample of Thumbnail container orientation in the left side, and usage of thumbnails for instant preview in a web-based video player shown in the right side.

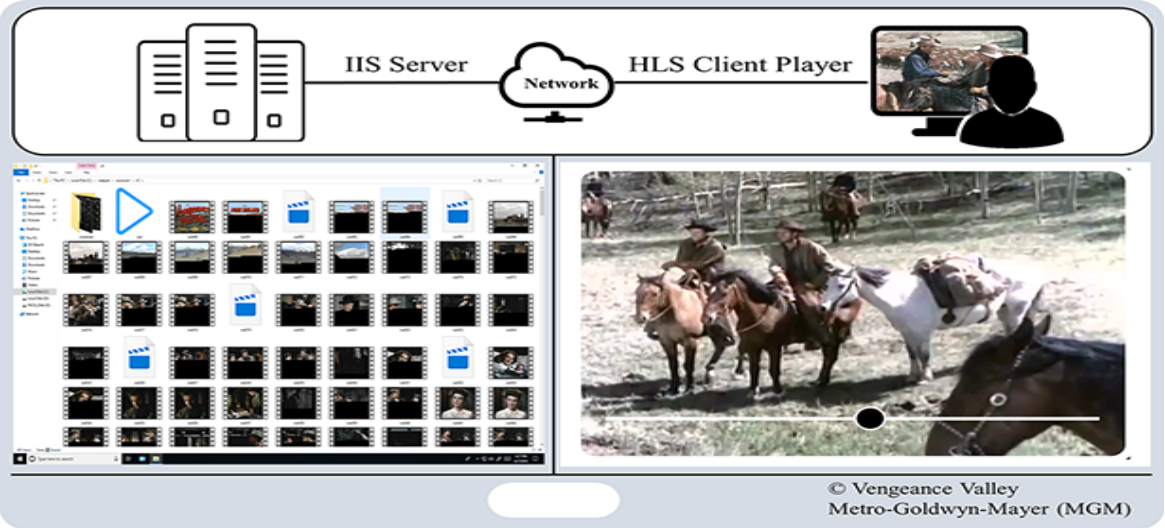

Fig. 3.4 HLS server with segments and thumbnail containers are on the left and source video on the right that preview a specific time in the user interface.

3.3 HLS Clients

The aim of the HLS client is to generate personalized multimedia content using the computational resources of client device. In the subsequent section, the universal components of the HLS client are described.

3.3.1 HTTP Persistent Connection

The personalized multimedia generation process initiates several requests from the user-end device to the server to obtain *ThmCon*s and *Seg*s. A lucrative HTTP 2.0 persistent connection is adopted in the process. This allows multiple requests exchange and data returned simultaneously over a single TCP connection [49]. Open connections are fast for frequent data exchanges because they remain accessible for HTTP requests and responses instead of terminating after a sole transaction. There are several benefits of using persistent connections. For example, fewer new connections and TLS handshakes reduce overall CPU utilization and round trips [50]. The following section describes the HLS client's user interface.

3.3.2 User Interface

The player interface is designed using the HLS JavaScript open-source library [51]. It depends on HTML5 and media source extensions for playback by transferring the *Seg*s. The data transmission is entirely client-driven. It means that the player can manage playback order or timestamps to analyze and determine a particular *Seg* from playlists. The player can also change and select different bitrates of the video during the playback. For this purpose, it uses M3U8 playlists of the corresponding video

to decide the available bitrate and position of the *Seg* in the server. Video content that represents the entire period (i.e., *Seg*s, playlixsts, etc.) can be calculated in a client-side VoD session. Figure 3.4 shows an HLS server with *Seg*s, *ThmCon*s are on the left and source video is on the right that previews a specific time in the user interface.

3.4 Hardware Configurations

The proposed framework is designed to support different hardware device configurations. Two distinct configuration devices are used as HLS clients to evaluate the proposed approach. The first end-user device is designed as the High Computational Resource (HCR) which consists of Ubuntu 18.04 LTS as the operating system. Meanwhile, the second end-user device is designed as the Low Computation Resource (LCR) configured on Nvidia Jetson TX2. The Jetson device is primarily selected to focus on resource-constrained devices. The Jetpack 4.3 SDK is adopted to automate the primary installation of the Nvidia Jetson TX2. It supports several energy profiles and max-n profiles used in all quantitative evaluations.

Throughout the personalized multimedia content generation process, the baseline and proposed approaches use the same device: the HLS server, HCR, and the LCR clients. All devices are locally configured and used in experimental evaluations. Meanwhile, the HLS server was assembled with Windows 10 and is utilized in every experiment. Each hardware device is locally connected to the Sungkyunkwan University (SKKU) network. Table 3.1 shows the specs for all hardware devices utilized in the experiments.

Table 3.1 Detailed hardware specifications for server and clients devices.

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

CHAPTER 4

PERSONALIZED MOVIE TRAILER GENERATION FRAMEWORK

Generally, a handful of movie trailers are generated as the one-size-fits-all framework in the film-making process. Streaming platforms are trying to overcome this problem by providing personalized trailers with centralized server-side solutions. The personal data of the users is needed and examined to complete the process. This leads to two main problems: privacy breaches and huge demands for computational resources.

This Chapter introduces a new client-driven framework that can facilitates the personalized trailer creation mechanism. It can generate trailers according to users' preferences as depicted in Figure 4.1. It tackles the sophisticated step of recognizing personalized events in real-time from lightweight thumbnails. Twenty-five western and sports genre full feature-length films are analyzed to test the efficiency of the proposed framework. The official trailers are then analyzed in comparison to the ones created by the proposed framework.

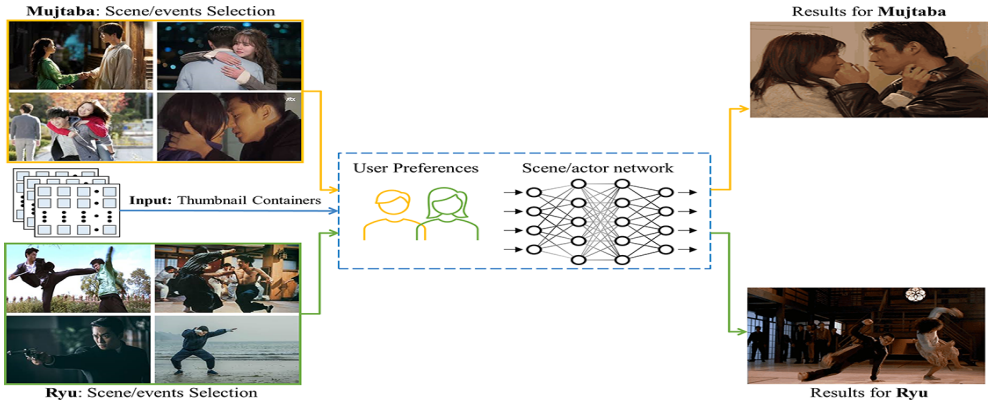

Fig. 4.1 Conceptual design of proposed framework that facilitates the personalized trailer generation process.

4.1 Background

According to a recent survey, everyday adults in the United States watch movies for nearly six hours [52]. This significant time allotment is insufficient for many users to explore relevant content [53]. Streaming platforms use different recommendation algorithms to suggest video content and quick exploration. The algorithm provides suggestions that are based on the viewer's history and sensitive data [54]. Despite the personalized recommendations, viewers may need to spend more time and interaction recognizing the importance of the content. Trailers generated according to user preference can facilitate quick media exploration and help users find relevant media content instantly.

Commonly, trailers are considered a subset of the source movie (video). Trailers are generated to highlight key segments and attract users to the movie [55]. Besides, trailer production demands a high level of cognitive exertion because of its diversity. Hence, a handful of movie trailers are generated in the film-making process. In some situations, a single trailer is generated for the corresponding movie. Since every user has a different set of preferences, that may reduce the popularity of the movie, which can cause enormous economic damage [56].

The centralized server-side techniques are under research and development to automate personalized movie trailers generation process [4]. The advantage of these techniques is that all information is co-located, such as user data, media content, and functionalities. The downside is that the client must send data about all user operations to the server. Therefore, it can compromise the privacy of users. Besides, the amount of information transmitted between the server and the client can rapidly increase as the number of users grows. Therefore, the server requires an enormous amount of computational resources to process all user information. These factors have led us to research conversational methods that allow users to control sensitive data and generate personalized trailers according to their interests. The following section presents the proposed framework.

4.2 Proposed Framework

The overall objective of the proposed framework is to ease the personalized trailer generation process. Instead of analyzing the entire video, it tackles the complicated procedure of identifying personalized events from lightweight thumbnails. It makes the proposed framework computationally efficient. Quantitative and qualitative assessments are conducted in 25 full feature-length films and documentaries in the sports and western genres. Later, the number of related events are analyzed from the official trailer using the generated ones.

The proposed framework design supports different end-user hardware devices with distinct configurations. Besides, the scalable design of the proposed framework can dynamically tune under a variety of network conditions. Figure 4.2 depicts the overall system design of the proposed framework. As described in Section 2.6, previous trailer generation methods ignore computational resource cost, processing time, and user interest. Table 4.1 depicts the comparison of the proposed framework with previous trailers approaches.

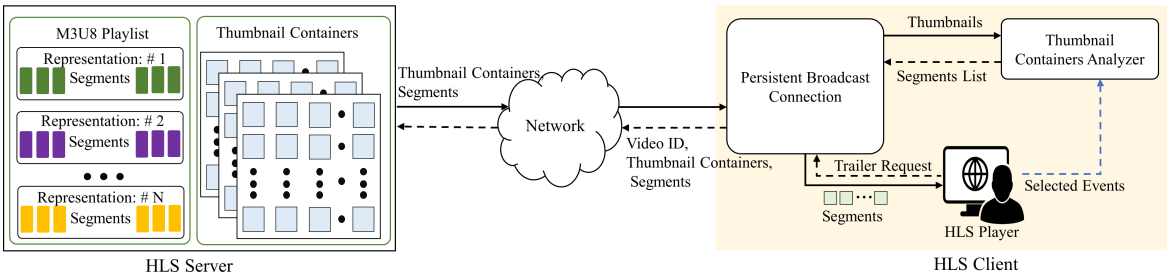

Fig. 4.2 Overall system design of the proposed trailer generation framework.

The main contributions of the proposed framework are: (i) proposed the first-ever client-driven thumbnail-based framework to facilitate the trailer generation process, (ii) introduce and employs HTTP persistent broadcast connections to decrease the corresponding response and network bandwidth, and (iii) server and client are configured locally to verify the capability of the proposed framework. The proposed framework uses the same modules and hardware device configurations that are described in Section 3 and Section 3.4, respectively. The following section describes the event recognition and trailer creation modules.

Table 4.1 The main comparison of the proposed framework with current approaches; V: *Video*, A: *Audio*, Tx: *Text*.

Approach	Genre	Video category	Personalize	Data type			
				A	V	Tx	TC
Smeaton,et al [44]	Action	Movie	×	✓	✓	×	×
Smith,et al [45]	Horror	Movie	×	✓	✓	×	×
Kawai,et al [46]	Adventure	Documentary	×	×	×	✓	×
Proposed	Western, Sports	Movie, Documentary	✓	×	×	×	✓

4.2.1 Events Recognition Module

This module is intended to recognize personalized events from *Thumss*. For this purpose, the 2D CNN network is trained on the UCF-101 dataset [57]. It has 101 action classes under 13,320 videos obtained from YouTube. Every video in the dataset is subsampled up to 40 frames to train the network. The SoA pre-trained ImageNet Inception-V3 image annotation network is used to acquire frame-level features [58]. The model is trained by freezing the top layers and updating the weights only in the final stages. The SGD optimization algorithm is used in the training process with a learning rate of 0.01. The first train and test use in the training process as suggested in [57]. Section 4.4.1 provide the details of performance evaluation of the proposed model.

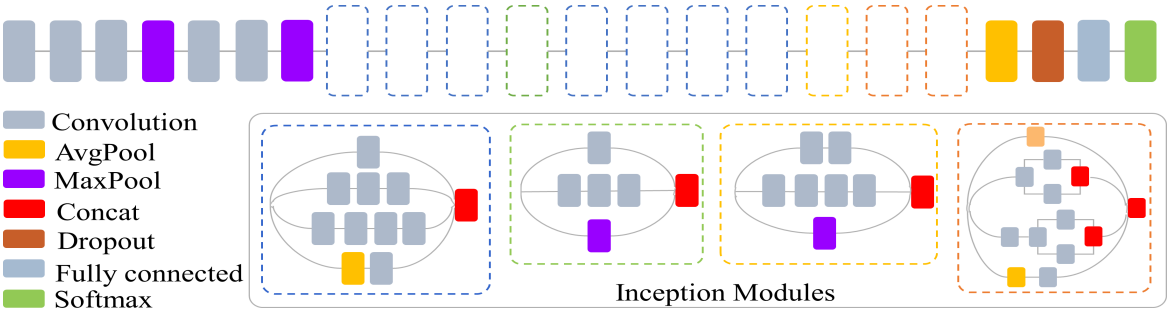

Fig. 4.3 Architecture of InceptionV3 2D CNN model used for events recognition from thumbnails.

4.2.2 Trailer Creation Module

This module is intended to select download *Seg*s and aggregate them into a single continuous video stream using FFmpeg [48]. The trailer generated is less than 150 seconds long. This length is managed by dynamically controlling the threshold while analyzing the events. For this purpose, the computational resources of the client device is used. The module is scalable but, the current version of the proposed approach only supports continuous stream playback.

4.3 Trailer Procedure from User Perspective

This section details on the proposed framework from the user perspective. Full 25 feature-length videos are examined to validate the efficiency of the proposed framework. The videos are selected from western and sports genre movies and documentaries categories. Table 4.6 depicts the title, release year, genre, IMDB rating, and duration of all videos. Video can consists of multiple genres, so here the most dominant video genre provided, i.e., western or sports. The content of all movies may vary. However, we have found that events are universal in the same genre of movies. For example, horse riding events are usually similar in all western films. It supported the creation of trailers by generalizing the events categories according to the video genres. All videos are analyzed by six different events picked from the UCF-101 action category list (refer to Figure 4.4).

Initially, the end-user device requires *ThmCon*s for the corresponding video to start the trainer generation process. HTTP persistent connection is used to transmit pre-generated *ThmCon*s from the server. The duration of the full video is covered by the sequence of transmitted *ThmCon*s. The number of frames is correlated with the FPS of the video. Meantime, the collection of *ThmCon*s is correlated with the length of the video. The number and size of *ThmCon*s are very negligible associated with the number of frames. Thus, a very low bit rate is required to transfer *ThmCon*s.

The proposed framework utilizes a canvas to obtain *Thum*ss from *ThmCon*s using FFmpeg [48]. The proposed 2D CNN network is used to identify an event from the extracted *Thum*ss. It needs two inputs every iteration : (i) preferred event and (ii) *Thum* images. The user employs the web interface to select an event. The event recognition process also allows the user to choose multiple events according to his/her preference. Figure 4.5 depicts event recognition process from *Thum*ss.

Fig. 4.4 The first four pictures depict an example of the events picked for the sports genre video: cricket shot, cricket bowling, soccer juggling, and soccer shot. The last two pictures depict an example of the events chosen for the western genre video: horse racing and horseback riding.

The 2D CNN network analyzes all *Thumss* individually according to the event elected by the user. The system chronically ranks the detected *Thumss* once all of them are analyzed. Only high-precision *Thumss* are elected, and later the information is saved in a text-based list.

The proposed system analyzes the detected *Thum* list to create another text-based list for *Seg*s. The second text-based file is used to request particular *Seg*s with varying timestamps for download from the server. All the specific *Seg*s are transmitted from server to client accordingly. If the video *Seg* is downloaded prolonged, an alternative bitrate can be picked by referencing the M3U8 playlist. Once all the *Seg*s are downloaded, the system can arrange and aggregate them according to user preferences.

4.4 Experiments

This section demonstrates comprehensive experimental assessments of the proposed framework. First, the efficiency of the proposed action recognition model is compared with the SoA 2D CNN methods. Then, the quantitative evaluation of the proposed framework is performed. Finally, the qualitative assessment of generated and the original trailers of the movies are presented.

4.4.1 Action Recognition Model

This subsection describes the evaluation of existing 2D CNN methods for the UCF-101 dataset. To the best of our knowledge, [59] reported the best performance on the UCF-101 dataset when utilizing

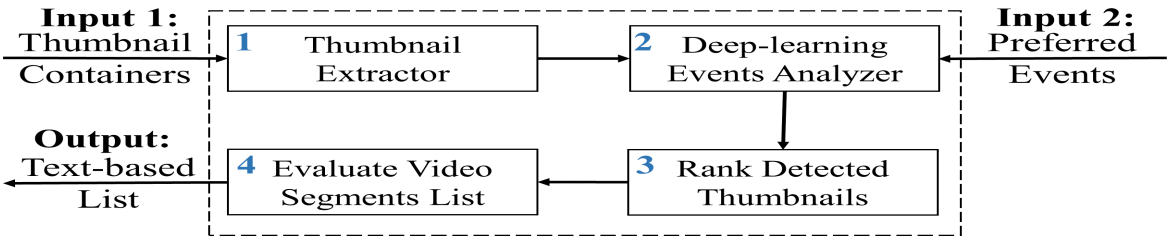

Fig. 4.5 Thumbnail containers events analyzer process.

2D CNN. The proposed CNN model is 0.95% better in the form of validation accuracy. The total sum of parameters in the proposed CNN model was 24 million. The experimental evaluation of the baseline and the proposed methods on the UCF-101 dataset are depicted in Table 4.2. The proposed 2D CNN model is adopted in each experiment to recognize personalized events from *Thum*ss.

4.4.2 Quantitative Evaluation

This section describes the quantitative evaluation of the proposed framework. For this purpose, full feature-length 25 videos with 13 western and 12 sports genres are selected. Besides, official trailers are collected for each corresponding video. The frame size for each video and its trailers is 640×480 ($width \times height$) pixels. The count of official trailers, YouTube ID, and total views are depicted in Table 4.6. If the video contained multiple official trailers, simply the one with the most views is noted and examined. One of the most famous streaming platforms (i.e., YouTube) is utilized to explore official trailers for all videos. "Trailer" and "promotion" are the important keywords in the searching cause. From the 25 list videos in the dataset, we are able to identify the official trailers for the two movies. Therefore, it is presumed that the official trailers for the two videos are not publicly available online.

The number and size of *ThmCon*s are extremely small correlated with the number and size of video frames. This reduces the computational resources and processing time required to examine the *ThmCon*s. The percentage comparison of the number of frames and *ThmCon*s in the video is depicted in Table 4.3.

Because this is the first client-driven method to expedite the trailer creation procedure. The entire video is analyzed is analyzed as the baseline scheme to compare the computational efficiency of the proposed framework. The overall processing time, in minutes required for the baseline and proposed approaches is shown in Table 4.4. To analyze the baseline method two phases are used: (i) frames obtain from the video using FFmpeg [48], and (ii) events recognition from frames using the proposed action recognition model. Meanwhile, all steps are performed using the proposed approach described in Section 4.3 to calculate the processing time. All of these seven steps are (i) requesting and downloading the corresponding movie *ThmCon*s, (ii) extracting *Thum*ss from the *ThmCon*, (iii) recognizing events from *Thum*ss, and (iv) detecting text-based Includes ranking and preparation of the list. Prepare a *Seg* list from *Thum*ss, (v) a text-based list of detected *Thum*ss, (vi) request and download specific *Seg*s, and (vii) aggregate the downloaded *Seg*s. The proposed thumbnail-based

Table 4.2 Comparison of proposed CNN action recognition model.

CNN Methods	Overall validation accuracy (%)
Karpathy, Andrej, et al. 2014 [60]	65.40%
Murthy, OV Ramana, et al. 2015 [59]	72.8%
Proposed	**73.75%**

Table 4.3 The overall performance efficiency of thumbnails compared to frames in the video.

S/N	FPS	# Frames	# *Thum*ss	%
4	23	189983	7923	**4.17**
18	25	163972	6558	**3.99**
25	30	147448	4919	**3.33**

approach used more steps, but the overall time is much shorter than when processing the baseline method. In this comparison, the *HCR* device is used as the end-user machine.

4.4.3 Qualitative Evaluation

The qualitative assessment is performed to compare the official trailer with the trailer generated using the proposed framework to gain user awareness. For this purpose, a set of 46 attendants participated. Those are divided into two groups according to their attractiveness in the film genre (western or sports). From a demographic point of view, attendants in the survey targeted a vast range of age groups (18–40 years) and 12 distinct geological places.

The trailer may include its background music, post-production characteristics, and overlay text, as explained in Section 4.1. Moreover, the most appealing and appropriate scenes from the movie are picked to explain the story in a concise way. The current version of the generated trailer using the proposed framework has the absence of post-production features, background music, and the selection of the most suitable scenes to captivate and inspire users. Due to these restrictions, it was

Table 4.4 The processing time in minutes is needed to generate the trailer using the baseline and proposed methods.

S/N	Baseline		Total (Baseline)	Proposed
	Frame extraction	Events recognition		
4	14.66	94.24	108.9	**2.85**
18	16.39	98.39	114.78	**2.55**
25	4.07	52.2	56.27	**2.04**

Table 4.5 Duration and number of events recognized from the official and generated trailers of the corresponding video.

S/N	Official trailers		Generated trailers	
	Duration	Events	Duration	Events
2	2 minutes 13 seconds	3	1 minutes 53 seconds	**6**
8	2 minutes 20 seconds	5	1 minutes 33 seconds	**7**
12	4 minutes 20 seconds	2	1 minutes 49 seconds	**6**
13	3 minutes 17 seconds	6	1 minutes 43 seconds	**8**
23	3 minutes 32 seconds	3	2 minutes 11 seconds	**8**
25	2 minutes 34 seconds	3	1 minutes 50 seconds	**6**

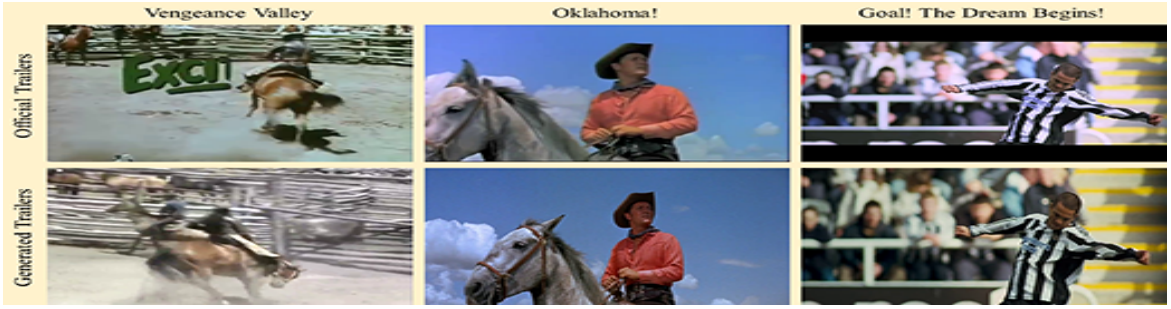

Fig. 4.6 Samples of similar events in official and generated trailers.

not possible to perform a user rating to rank the official trailers and the generated trailers. Therefore, a purely analytical investigation is performed to discover total events according to the interests of the participants. Participants further participated focused on similar events in official and generated trailers.

The official and generated trailers are presented to the participants at the same time. Table 4.5 show the qualitative results obtained from the participants. Attendants calculate the number of events that correspond to the genre of the movie. The obtained results show that the generated trailer has more events compared to the official trailer. The participants have identified similar events of the official and generated trailers throughout the analysis. Figure 4.6 depicts the example of the common events of the official and generated trailers from three movies. Attendees observe that the style of the generated trailer is similar to the corresponding movie genre.

4.5 Discussion

This Chapter introduces an innovative thumbnail-based client-driven framework that facilitates the trailer generation process. Instead of processing the entire video, the proposed framework examines the lightweight thumbnails to recognize personalized events. Twenty-five broadcast feature-length videos are evaluated to find the effectiveness of the proposed framework. Quantitative results show that the proposed framework require less processing time compared to the baseline approach. The qualitative evaluation is conducted with the help of 46 participants. The official trailers of two movies in the dataset could not be obtained. Therefore, it is assumed that there is no official trailer available online. In this case, the proposed framework can assist in the trailer generation process. This study focused on the sports and western genres of the trailer generation process. However, it can easily adapt to other video genres. It can support privacy protection solutions that allow users to choose to retain personal data on the client-side [61]. The proposed client-driven approach is highly responsive and scalable to support a wide range of end-user hardware devices with a variety of computational capabilities.

Table 4.6 The minutiae of video titles utilized in the trailer generation process.

S/N	Title	Genre	IMDB	Length	FPS	*Fra*	*ThmCon*	*Thum*	Trailer	YouTube
1	89, 2017	Sport	7.8	1h31min	25	135,300	217	5412	1	UmUiHof01l4
2	Bobby, 2016	Sport	7.1	1h37min	25	1,420,204	225	5608	1	f1dJSOU-CUk
3	Bodyline, 1984	Sport	8.5	5h30min	25	73,820	119	2952	0	-
4	Bone Tomahawk, 2015	Western	7.1	2h12min	23	189,983	317	7923	1	QuGmtoQBPEM
5	Dakota, 1945	Western	6.1	1h22min	24	117,895	197	4917	1	CVeqwq-ZvVI
6	Django, 1966	Western	7.3	1h31min	24	131,749	220	5495	2	w8Ge2hmSTbo
7	Django Unchained	Western	8.4	2h45min	24	237,909	397	9922	3	eUdM9vrCbow
8	Goal! 2005	Sport	6.7	1h58min	23	169,932	284	7087	1	67LM5X9-MHA
9	Iqbal, 2005	Sport	8.1	2h12min	25	189,945	304	7597	1	5OIco9k7KcE
10	Kenny, 2017	Sport	7.3	1h26min	25	129,649	208	5186	1	6mA6uA2-Rcw
11	Lagaan, 2001	Sport	8.2	3h44min	23	322,944	539	13469	1	oSIGQ0YkFxs
12	Little Big Man, 1970	Western	7.6	2h19min	24	200,509	335	8362	1	7K4l5ZZe4-k
13	Dhoni, 2016	Sport	7.7	3h4min	24	265,933	444	11080	1	6L6XqWoS8tw
14	Oklahoma, 1955	Western	7	2h25min	24	201,422	337	8401	1	V6uD9-aLCps
15	Pelé, 2016	Sport	7.2	1h47min	23	147,545	257	6415	1	XBrfxHOXsDE
16	Playing Away, 1987	Sport	6.6	1h40min	24	146,237	244	6093	0	-
17	Shanghai Noon, 2000	Western	6.5	1h50min	23	158,648	265	6617	1	FqHg5fc_0_U
18	Take The Ball, 2018	Sport	8.2	1h49min	25	163,972	263	6558	1	VfKls9Eo1ZI
19	The Game of Their Lives, 2005	Sport	6.1	1h41min	24	145,699	244	6076	1	1H2fRn8PStw
20	Indian Fighter, 1955	Western	6.4	1h28min	23	127,126	213	5302	1	hWP2Un2Dr5I
21	Vengeance Valley	Western	6.4	2h30min	24	148,404	248	6189	2	AlrWRttLTkg
22	The Rider, 2017	Western	7.4	1h44min	24	141,143	236	5886	1	2IV3LvS_M6M
23	The Tracker, 2002	Western	7.4	1h30min	23	132,192	221	5514	1	P0rK5Q-TX-k
24	Train Robbers, 1973	Western	6.5	1h32min	24	158,733	265	6620	2	CUiCu-zuAgM
25	True Grit, 2010	Western	7.6	1h50min	30	147,448	197	4919	1	j2srpV2RN-4

CHAPTER 5

PERSONALIZED ANIMATED GIF GENERATION FRAMEWORK

Animated GIFs play a unique role in the streaming platforms to increase the CTR of the corresponding video. There is a strong correlation between personalization and GIFs, which results in higher CTR. They become ubiquitous in the streaming platforms to instantly highlight recommended videos as shown in Figure 5.1. However, GIFs are being generated and provided on the streaming platforms without user preferences.

This Chapter proposes a lightweight framework for generating personalized GIFs employing the computing capabilities of client devices. The proposed approach analyzes lightweight thumbnail containers to recognize personalized events from long-form videos i.e., sports matches. This minimizes the overall computational complexity compared to baseline approaches. To the best of our knowledge, it is the first method that adopts thumbnail containers in the GIF creation process for streaming platforms.

5.1 Background

The popularity of GIF has grown rapidly in recent years, as the development of new social media applications and usage of instant messengers have increased. GIF images are popular among netizens due to their small size, short length, and storytelling nature (no audio) [13]. Because of these traits, the GIFs have been viewed over 2 billion times on the GIPHY website alone [62]. Their usage and importance are ubiquitous in streaming platforms and help the users to preview recommended videos instantly. Despite this extensive use of streaming platforms, academic research on GIF generation is limited.

Currently, GIFs are more popular than ever on streaming platforms because of their simple format and portability. This enables their broad usage on any web browser without requiring any extra plugins. Its ubiquitous adoption and prevalence have increased the demand for designing lightweight personalized GIF generation methods. Some related researches have focused on the lightweight approach to generating GIFs for streaming platforms [63, 43]. However, these previous efforts have overlooked during the personalization phase. This is mainly due to the need for private user data and the significant demand for computational resources required for processing. Dedicated server-side

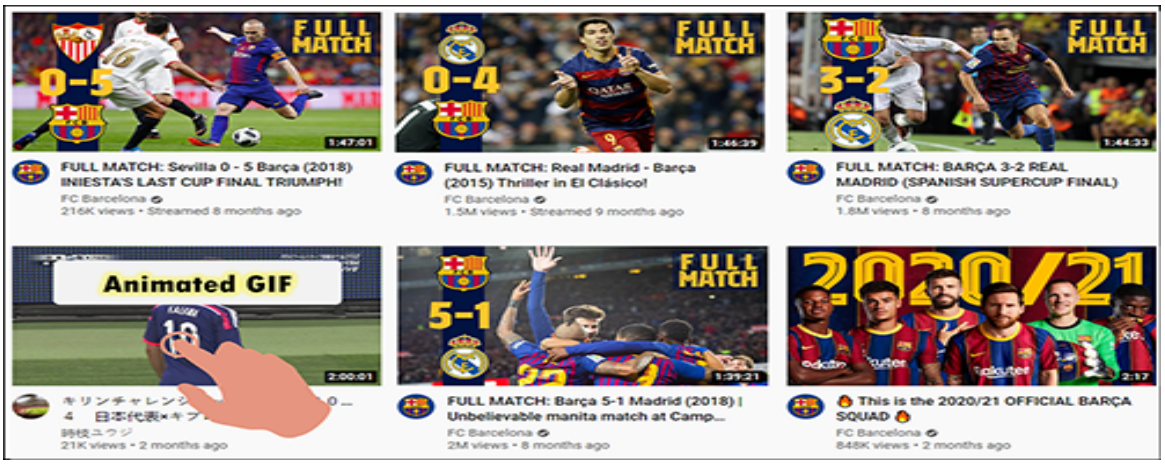

Fig. 5.1 Sample screenshot of the YouTube homepage, where animated GIFs are adopted to highlight recommended videos.

solutions can solve this problem; however, user privacy violation is the primary concern in such solutions.

The CTR is a vital metric for streaming platforms, and GIFs play a decisive role. There is a strong correlation between the GIF and personalization, which increases the overall CTR of the corresponding video if GIF is more relevant. As a result, GIFs are becoming more and more important in the process of video selection. However, presently they are generated through the universal framework without user consent. Users may not like a particular GIF because the particular GIF does not match their interests. This can cause the user to skip the video, which will significantly reduce the CTR of the video.

Lightweight client-driven techniques for generating GIFs are still in their infancy and need more effective ways to bridge the semantic gap between video understanding and personalization. Most modern client devices have limited computing resources, and analyzing the entire video will take significant time to generate animated GIFs. That is not feasible for real-time solutions. Since personalization is one of the essential elements of early media content adoption, here, we focused on the personalization and lightweight aspects of the GIF generation process. This Chapter proposes an effective GIF generation scheme that considering the preferences of the user and resource-constrained devices. The following section describes the proposed GIF generation framework.

5.2 Proposed Framework

The use of streaming platforms is more popular than ever compared to the traditional systems [3]. The importance of streaming platforms is the spotlight by the statistic that about 500 hours of video per minute are only uploaded on YouTube [64]. Streaming platforms use *ThmCon*s and animated images to instantly provide users with an overview of this vast collection of video content. GIFs are used to get a little glimpse of the recommended video content. Meanwhile, *ThmCon*s are adopted to

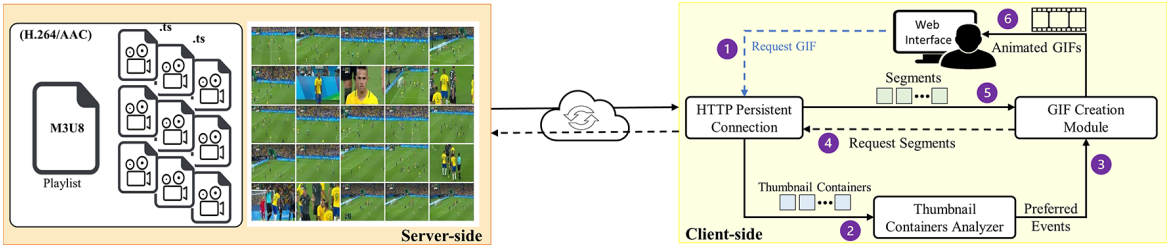

Fig. 5.2 Schematic illustration of the proposed GIF creation framework.

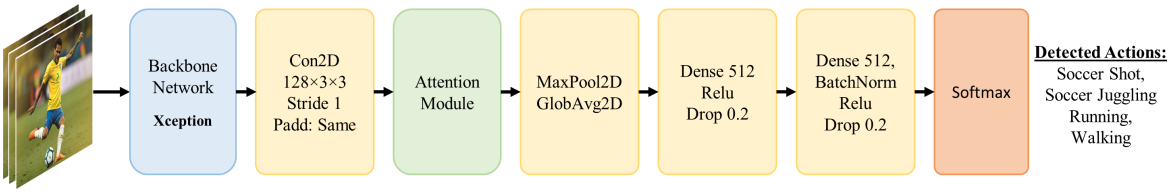

Fig. 5.3 The proposed 2D CNN model used to detect personalized events from thumbnails.

manipulate the video timeline. Owing to the usage of *ThmCon*s and the popularity of animated GIFs on streaming platforms, we have proposed an innovative client-driven method.

This Chapter proposes an innovative client-driven approach to generate personalized animated GIFs that is simultaneously computationally efficient. To make the proposed method lighter, *ThmCon*s are used instead of the entire video to analyze personalized events. This makes the process more effective and reduces the overall processing time compared to previous methods. Besides, it uses small segments to generate GIFs, reducing valuable network bandwidth and storage needs. Full feature-length six broadcast soccer videos are analyzed to estimate the effectiveness of the proposed method.

The main contributions of this research can be summarized as follows. (1) A new lightweight client-driven method is proposed to create animated GIFs according to user preference. (2) The 2D CNN model is designed to recognize personalized events according to user interests. (3) Extensive quantitative and qualitative analysis is performed using full feature-length six sports videos. Quantitative results demonstrate the proposed approach is 3.57 times more computationally efficient than the SoA approach. We also conducted qualitative evaluations in collaboration with nine participants.

Figure 5.2 shows schematic illustration of the proposed GIF creation framework. The proposed framework consists of two main parts: *HLS server* and *HLS client*. Chapter 3 provides the details of the HLS server and general modules of the HLS client. The configuration and role of the event recognition and the GIF generation modules in the proposed framework are described in the following section.

5.2.1 Events Recognition Module

The purpose of this module is to analyze the *ThmCon*s according to the user event preferences. For this purpose, the 2D CNN model is designed to analyze *Thum*s. The 2D network is trained on

the UCF-101 dataset [57]. The dataset contains 13,320 videos consisting of 101 different action categories. Frame-level features are extracted using the SoA Xception image annotation model [65] that is pre-trained with ImageNet [66]. Figure 5.3 shows the proposed 2D CNN architecture. The performance of the proposed network is enhanced by using an attention module derived from the vortex pooling [67]. This module aggregates contextual information using multi-branch convolution with scaling factor to make it more effective.

Data enhancements are applied to reduce overfitting with the proposed approach. The dataset is split into two parts, and the first list is used in the training process as suggested in [57]. Each video is subsampled up to 40 frames to train the model using the UCF-101 dataset. Before being served as input to the network, all images are pre-processed by first cropping the central area and then resizing them to 244×244 pixels. Shear transformations are also performed at 20° angles, 10° random rotations, 0.2 horizontal and vertical shifts, and random horizontal flips of the image. The model i trained with a variant of the SGD algorithm with momentum 0.9 and learning rate 0.01, using the default weight decay values (SGDW) [68]. In the experiment, an early stop mechanism is applied during the training process with ten patience. The training data is provided in mini-batch with a size of 32 and a learning rate of 0.001 to minimize costs. The Keras toolbox is used for deep feature extraction and the GeForce RTX 2080 Ti GPU for implementation. Section 5.5.1 provides the validation accuracy of the proposed events recognition model.

5.2.2 GIF Generation Module

The purpose of this module is to determine the *Seg* number from the detected $Thum$s, download specific *Seg*s, and finally generate an animated GIF from the corresponding *Seg*. The first three seconds of the *Seg* are used to generate a GIF using this module. The length of each GIF is fixed in the proposed approach, but the module can be extended to a specific length according to user preference. Besides, the proposed module can manage to generate animated GIFs from the climax part of the video. A structured story generally consists of explanations, ascending actions, climaxes, and end. The most exciting part of the plot is the climax section where all the major events occur, which represents the most exciting part [43]. Figure 5.4 depicts the classic story plot structure of the Big Buck Bunny (2008) video. The next section details the GIF creation process.

5.3 Animated GIF Generation Process

This section describes the entire flow of the proposed GIF generation process from the perspective of a user. The flow is explained based on the full six soccer 2018 World Cup videos obtained from YouTube. Table 5.5 shows the complete description of the selected videos. The number of views of the corresponding video is counted in December 2020. It also shows the number of frames per second (FPS) of the video, the total number of frames, $ThmCon$s, and $Thum$s. The resolution of all videos used in the experiment is 640×480 pixels. All videos are investigated using two different events

selected from the UCF-101 action category list. The two events chosen are soccer shots and soccer juggling. These events are selected based on the genre/type of video. Note that the proposed approach is not limited to these events. It can adopt additional events depending on the content of the video and the dataset.

To generate an animated GIF for a particular video, the user first selects the video from the web interface. The client device requests and downloads *ThmCon*s for the video. The transferred *ThmCon*s covers the entire length of the video. The number of *ThmCon*s is much less than the number of frames in the video (refer to Table 5.5). Therefore, the bit rate required during transmission is significantly lower. Each *Thum* is extracted individually from *ThmCon*s using the canvas. Each *ThmCon*s comprises twenty-five *Thum*s. After selecting the video title, the corresponding video event is selected later using the web interface. The user can select multiple events during the GIF generation process. The proposed 2D CNN model requires two inputs during the recognition process: 1) *Thum* and 2) preferred event.

All steps that are involved in the GIF generation process are shown in Figure 5.5. The deep learning model analyzes each *Thum* individually based on the event selected by the user. Once all *Thum*s are analyzed, they are ranked in chronological order and stored in a text-based file. *Thum*s are selected based on the threshold to maintain the quality of the generated GIF. A text-based file is parsed and the *Seg*s are downloaded to get a specific *Seg* of the selected *Thum*. The client device then requests a specific *Seg* with a different timestamp. The HTTP server responds to this request by sending the corresponding *Seg*. The *Seg* is then used to generate the animated GIF. The GIF is created from the *Seg* using FFmpeg [48].

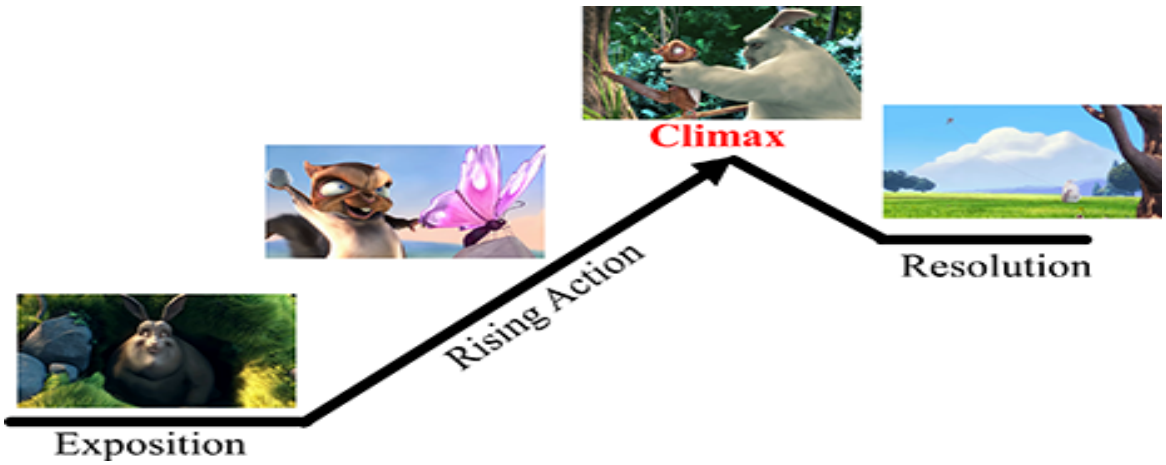

Fig. 5.4 The classical plot architecture of storyline videos.

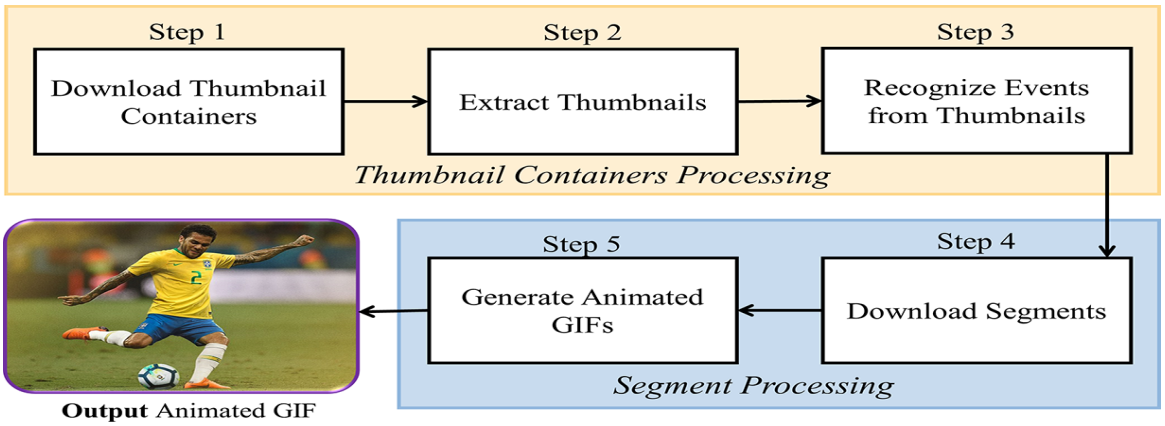

Fig. 5.5 All steps are required to generate an animated GIF using the proposed method.

5.4 Baseline Methods

This subsection describes some of the well-known baseline approaches used to compare the proposed GIF generation methods. The baseline approach is as follows:

- **HECATE** [5]: This baseline approach analyzes aesthetic features from the entire video frame. It extracts and temporarily saves the corresponding video frame on the end-user device to analyze. This method only supports fixed periods and the number of GIFs. Here, three GIFs are generated for each video.
- **AV-GIF** [43]: It analyzes the entire audio and video files of the corresponding video to generate a GIF. This is the baseline approach used in [43]. The default parameters are used to generate the animated GIF. Here, one GIF is generated for each video.
- **Climax-GIF** [43]: This baseline approach uses the climax part of the audio and corresponding *Seg*s to generate a GIF. This is the current state-of-the-art client-driven method for animated GIF generation. The default parameters are used to generate the animated GIF. Here, one GIF is generated for each video.

5.5 Experiments

This section shows an extensive experimental evaluation of the proposed approach. First, the experimental setup is explained with the baseline approaches. The accuracy of the proposed behavioral recognition model is then presented, and its validation accuracy is compared to previous well-known approaches. Finally, the performance of the proposed method is compared with the baseline methods.

5.5.1 Action Recognition Model

This section describes the evaluation of existing 2D CNN methods for the UCF-101 dataset. As far as we know, [12] is the only method and reported the best performance on the UCF-101 dataset when using the *ThmCon*s to recognize events. The proposed CNN model is 2.5% better in terms of validation accuracy, within the range of 51.32 million floating-point operations per second. The total number of parameters in the proposed CNN model is 25.6 million. Table 5.1 shows the experimental results of the baseline and proposed methods on the UCF-101 dataset. All 2D CNN models [65, 69–71, 58] are trained on the UCF-101 dataset with a similar configuration as described in Section 5.2.1, however without the attention module.

Table 5.1 Comparison of proposed CNN action recognition method with other approaches.

CNN Methods	Overall validation accuracy (%)
MobileNetV2 [69]	59.06%
MobileNetV3Small [70]	68.75%
MobileNetV3Large [70]	71.88%
DenseNet121 [71]	65.31%
InceptionV3 [58]	61.25%
Karpathy, Andrej, et al. 2014 [60]	65.40%
Shu, Yu, et al. 2018 [72]	76.07%
Mujtaba, et al. 2020 [12]	73.75%
Xception [65]	68.44%
Proposed	**76.25%**

5.5.2 Quantitative Evaluation

This section provides the performance evaluations of the proposed and the baseline methods which are described in Section 5.4. This performance evaluations is performed using six soccer videos (refer to Table 5.5 for to obtain details of the videos). The computation time of the proposed method is calculated: (i) time required to download the *ThmCon*s corresponding to the video, (ii) obtain *Thum* extraction from the *ThmCon*s, (iii) recognize the personalized events from the *Thum*, (iv) estimate the *Seg* number and download *Seg* for the text-based list, and (v) generate the GIF from the *Seg*. All *Thum*s are selected with an accuracy above the threshold of 80.0%. This threshold is set to maintain the quality of the GIF. In all experiments, the model load time is not added when determining the computation time.

The computational time required to generate an animated GIF using the proposed and the baseline approaches is evaluated in the first experiment. All approaches consisted of HCR devices (refer Table 3.1 for detailed specifications). Table 5.2 shows the computational time required in seconds for each step using the proposed method on the HCR device to generate the GIF. Table 5.3 shows the calculation time required in minutes for the baseline and the proposed methods on the HCR device

Table 5.2 Computation time required in seconds to generate GIFs using the proposed techniques on the HCR device.

S/N	Download *ThmCon*	Extract *Thum*s	Recognize events	Download *Seg*s	Generate GIFs	Total (seconds)
1	4.27	7.86	96.95	0.56	21.53	131.17
2	4.48	7.93	95.29	1.56	23.27	132.53
3	4.29	7.95	94.28	1.31	26.88	134.71
4	4.89	8.17	98.89	2.05	19.68	133.68
5	4.5	8.34	92.92	1.21	18.47	125.44
6	4.17	7.94	94.24	2.52	15.88	124.75

Table 5.3 Computation time required in minutes for baseline and proposed methods to generate GIFs on the LCR and HCR devices.

S/N	HACATE [5]	AV-GIF [43]	Climax-GIF [43]		Proposed	
	HCR		LCR	HCR	LCR	HCR
1	85.34	16.69	38.71	6.83	10.08	**2.19**
2	85.64	15.78	36.17	6.48	9.85	**2.21**
3	81.86	15.56	35.40	6.42	9.32	**2.25**
4	59.61	16.18	40.06	6.85	10.45	**2.23**
5	57.81	15.35	37.96	6.57	13.96	**2.09**
6	87.99	19.87	35.60	6.57	8.92	**2.08**

to generate the GIFs. HECATE [5] method analyzes every frame of the video and determines the aesthetic features for generating a GIF. The AV-GIF baseline method [43] uses the entire audio and video file for the GIFs generation process. The Climax-GIF [43] is the current client-driven SoA method for GIF generation, and it uses the climax part of the audio and *Seg* for the GIF generation process. The proposed method uses a very small image (thumbnail) to analyze the personalized event. This significantly reduces the computation time during the GIF generation process.

Since this study focused on generating GIFs using resource-constrained client devices, the proposed and baseline approaches use LCR devices (i.e., Nvidia Jetson TX2) in subsequent experiments. Table 5.3 shows the computation time required in minutes using the baseline and the proposed method to generate GIFs using the baseline and proposed approaches. In this experiment, HECATE [5] and AV-GIF [43] could not be examined because they require significant computational resources. The Climax-GIF [43] method is the only baseline approach which is examined on the LCR device. The overall computational time of the proposed method is significantly lesser than Climax-GIF [43].

From the communication and storage perspective, the proposed method is also more efficient than the baseline methods. The HECATE method requires a locally stored video file to start processing [5]. Similarly, the Climax-GIF method is used to generate a GIF [43], the entire corresponding audio file and video *Seg* need to be download. However, the proposed method requires the same process to download a lightweight *ThmCon*s. For example, in the Germany-Mexico match, the file size of video

and audio is 551 MB and 149 MB, respectively. However, the size of *ThmCon*s is 22.2 MB for the same video. Therefore, the proposed method has significantly reduced the download time and storage requirements compared to the baseline methods.

The total time for the six videos is 666.87 minutes. HECATE [5] required 458.25 minutes, AV-GIF [43] required 99.43 minutes, Climax-GIF [43] required 39.72 minutes, and the proposed method requires 17.06 minutes to generate a GIF for the videos on the HCR device. Meanwhile, Climax-GIF [43] requires 223.92 minutes, and the proposed method required 62.59 minutes on the LRC device to generates GIFs. Therefore, based on the analysis of these six videos, on average, the proposed method is 26.86 times faster than the HECATE [5] method, 5.83 times faster than the AV-GIF [43] method, and 2.33 times faster than the Climax-GIF [43] method when using the HCR device. Similarly, for the LCR device, the proposed approach is 3.57 times faster than the Climax-GIF [43] method. The proposed method also generates more GIFs than the baseline method. For every video, HECATE [5] three GIFs, AV-GIF [43] and Climax-GIF [43] one-GIF, and the proposed method ten GIFs are generated. In summary, the results validate that the proposed method is more computationally efficient than the baseline methods: HCR and LCR.

5.5.3 Qualitative Evaluation

This section evaluates the quality of generated GIFs of the proposed method by comparing them with GIFs taken from YouTube and generated by using the baseline approaches. The evaluation is based on a survey of 9 participants. A group of students was selected according to their interest in football. The survey is based on 6 videos (table 5.5). The quality of the generated GIF was evaluated using an accurate rating scale. Participants were asked to evaluate the GIF based on their perceived arousal. An anonymous questionnaire was created for the generated GIFs, preventing users from determining the method used to create the GIF. Participants were asked to view all GIFs and rank them on a scale of 1–10 (1 is the lowest and 10 is the highest). Table 5.4 shows the ratings given by the participants for all approaches. For the six videos, YouTube, HECATE [5], Climax-GIF [43], the average rating of the proposed method is 5.0, 6.46, 5.65, and 7.48, respectively. Figure 5.6 shows a sample frame from a GIF generated by each of the three approaches.

Table 5.4 Average ratings (1∼10) assigned by participants for the proposed and baseline methods

S/N	YouTube	HECATE [5]	Climax-GIF [43]	Proposed
1	4.67	6.78	5.67	**8.11**
2	4.67	6.22	7.00	**8.56**
3	4.78	7.56	5.33	**8.44**
4	5.56	5.44	5.22	**5.78**
5	4.22	6.33	5.00	**7.44**
6	6.11	6.44	5.67	**6.56**

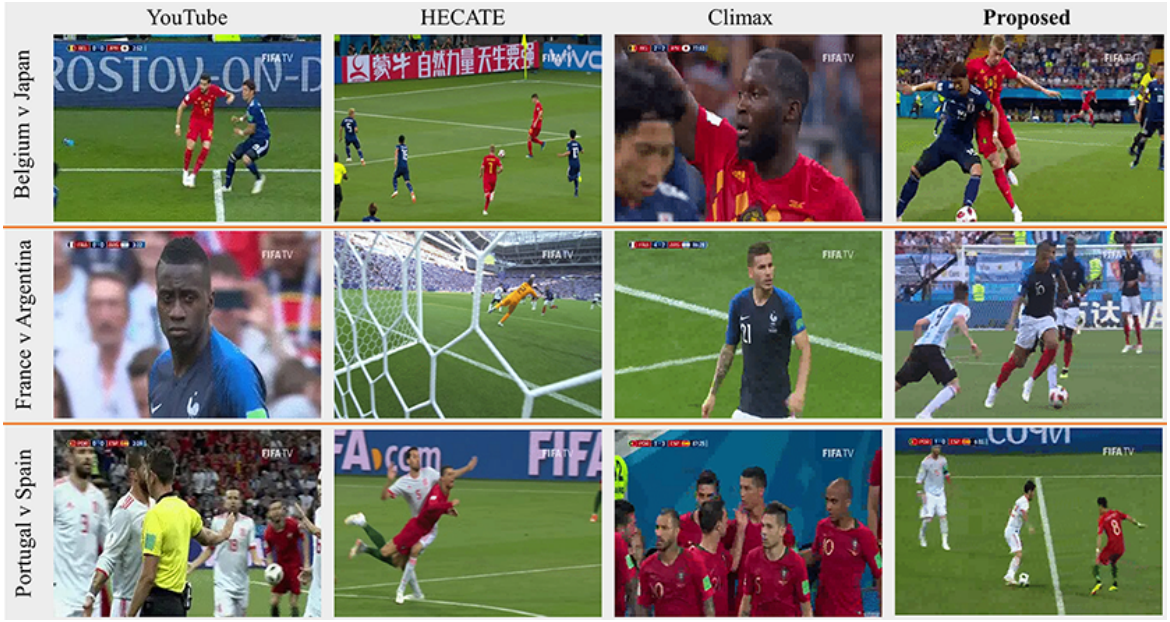

Fig. 5.6 Sample frames from GIFs generated using the baseline and proposed approaches.

5.6 Discussion

This chapter introduces an innovative and lightweight technique to generate animated GIFs. The proposed method uses the computational resources of client devices during the entire process. It analyzes the thumbnail containers to recognize personalized events and uses the corresponding video segments to generate animated GIFs. This improves computational efficiency and reduces the demand for communication and storage resources. Extensive experimental results obtained based on a set of full feature-length six videos show that the proposed approach is 2.33 and 3.57 times faster than the current SoA method on HCR and LCR devices, respectively. Qualitative results indicate that the proposed method has outperformed the existing methods and has received a higher ratings. In the future, the proposed method can be adapted to other sports categories by considering different events and datasets.

Table 5.5 List of videos used for analysis in the proposed GIF generation method.

S/N	Title	FPS	Length	# Frames	# *ThmCon*	# *Thums*	Views	YouTube ID
1	Belgium vs Japan	30	1h 52min	202,036	270	6734	1.1m	ervkVzoFJ5w
2	Brazil vs Belgium	30	1h 50min	199,506	267	6650	0.9m	5OJfbYQtKtk
3	France vs Argentina	25	1h 50min	165,653	266	6626	2.6m	J41d0cHAfSM
4	France vs Croatia	30	1 h 54min	205,243	274	6841	1.3m	7Fau-IwbuJc
5	Germany vs Mexico	30	1 h 48min	196,106	262	6536	1.1m	3fYpcapas0k
6	Portugal vs Spain	30	1 h 50min	198,556	266	6625	1.7m	Xhu5Bz1xDf0

CHAPTER 6

PERSONALIZED VIDEO SUMMARIZATION FRAMEWORK

Today, we are witnessing the tremendous growth of online video data. This is driven by two main factors: (i) constant increase in user engagement with smart and powerful video recording devices which have content sharing capabilities online, and (ii) widespread use of social media networks and video sharing platforms as a means of communication by billions of people [3]. This enormous growth has increased the need for technologies that enable users to browse through vast and growing collections and help them quickly retrieve video content according to their interests. The development of automatic video summarization techniques is part of the response to this demand. These techniques produce a short version of the full-length video that convey the meaningful segments. Based on this, viewers can quickly get an overview of the entire story without watching the full-length video. For instance, a typical full feature-length romantic video can be summarized into few minutes, highlighting meaningful events such as cuddling.

This Chapter proposes a novel lightweight framework to create a personalized video summary on the end-user device. It deals complex process of detecting personalized events from lightweight thumbnail containers. This makes it computationally efficient and unlike state-of-the-art approaches, the entire video is not processed to generate a summary. Simultaneously, bandwidth and storage are lessened, as the summary generation device only retrieves the lightweight thumbnail containers of the corresponding video instead of the entire video from the source. To the best of our knowledge, this is the first personalized client-driven summarization framework that analyzes thumbnail containers to generate a subset.

6.1 Background

Over past decades, several approaches are proposed to automate video summarization. In general, these techniques fall into two categories: keyframes [8, 35, 6, 36] and keyshots [5, 7, 9, 37, 38, 10]. Keyframes are also known as static storyboards, representative frames, or static image summaries. Meanwhile, keyshots can also 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 approximate visual representation. Meanwhile, keyshots consist of typical

continuous video segments of the full-length video which are shorter than the original video. All summarization techniques have their importance based on the application to which they are applied. Generally, keyframe-based summaries are lighter in size than keyshot summaries. However, sufficient valuable information is omitted during the process. For example, it is challenging to get the context of the previous frame in a keyframe-based summary. Also, it lacks original sound. Consequently, keyshot-based video summarization methods are predominantly selected to handle and to overcome these challenges.

Keyshot-based video summarization methods are used to produce subsets for short-form videos (i.e., user-generated TikTok and news) or long-form videos (i.e., movies and soccer matches). Generally, the length of short-form and long-form videos are under 10 minutes and over 10 minutes, respectively [11]. The playback duration of short-form videos is already concise, so it is impractical and may be ineffective to generate keyshot-based subsets for such videos. Meanwhile, the playback duration for long-form videos can exceed 90 minutes, specifically in movie and sports videos categories. The keyshot-based summary methods are more practical and effective for such video categories to provide a quick overview to users.

The sample video contains a variety of information such as character appearance, motion, interactions between objects, events, scenes, and more. Considering a one-hour long-form video at 25 Frames Per Second (FPS), there will be thousands of frames in it. Existing approaches require extensive computational resources to process the entire video data (i.e., frames) [5, 7, 9, 10]. If the video is of high definition or over high definition, the demand for computing resources will increase further. Deep learning-based methods also require segmented processing of long-form videos, further increasing the number of processing steps and the demand for computational resources [8]. This is not a viable approach for resource-constrained devices to process all frames because it increases the overall computational time. Keeping in mind that computational resources are limited, there is a lack of a lightweight keyshot-based video summary method for long-form videos.

Video summarization is a daunting task due to its subjectiveness. This is because every user has different preferences, even for similar video content. The personalized video summarization method provides precise solutions to this problem [73]. The algorithms aim to generate customized content for every user according to their interests. However, personalized video summaries with optimal lengths for new long-form broadcasted videos (e.g., sports matches) are not immediately available to users. With current approaches [35, 8], generating personalized summaries in real-time will require enormous computational resources to process user data and video content. The centralized dedicated servers can provide real-time solutions to this problem as all the information is in the same location, including video content and user data. However, user privacy is a serious concern for server-based solutions, as the server has extensive sensitive data about user interests. These issues encourage us to overcome and discover the feasibility of a client-driven approach that can create personalized keyshot video summaries on resource-constrained devices.

The automated video summarization process in real-time on the client device is still an unsolved problem despite the extensive work. This is mainly because the current summarization techniques

uses segments [8] or entire frames/video [36–38, 35, 6, 10, 7, 9] data in the process that require enormous computational resources. 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 at the client device. This is not feasible for real-time requirements. In addition, the video summarization methods that required semantic information in the process may require mining and pre-processing steps [37] to get useful information, and information may not be publicly available for all videos such as scripts and subtitles [8]. 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 that makes the process computation, communication, and storage efficient and fast to create a summary in real-time.

6.2 Proposed Framework

This Chapter proposes a novel client-driven framework called LTC-SUM that uses Lightweight Thumbnail Container in the SUMmarization process. It handles the complex process of detecting personalized events (such as penalty shoot-outs of soccer videos) from lightweight thumbnails. This makes the proposed approach computationally efficient because the entire video is not processed. In addition, the proposed technique is efficient in terms of communication (between server and client) and storage requirement, as the entire video does not need to be transmitted over the network and stored. Contrary to previous keyshot-based methods [5, 7, 9, 10], this paper aims to generate subsets for long-form videos such as movies and documentaries. The proposed approach is a fully client-driven which can generate distinct video summaries automatically for concurrent users according to their interests (see Figure 6.1 for an example). The main technical contributions to this study are summarized as follows:

The main technical contributions of this study are summarized as follows: (1) A novel thumbnail-based client-driven framework is proposed to generate keyshot video summaries according to user preference. (2) A lightweight two-dimensional conventional neural network (2D CNN) model is designed that can identify personalized events from thumbnails. (3) Quantitative and qualitative evaluations are conducted on eighteen movies and documentaries (approximately 32.9 h of duration).

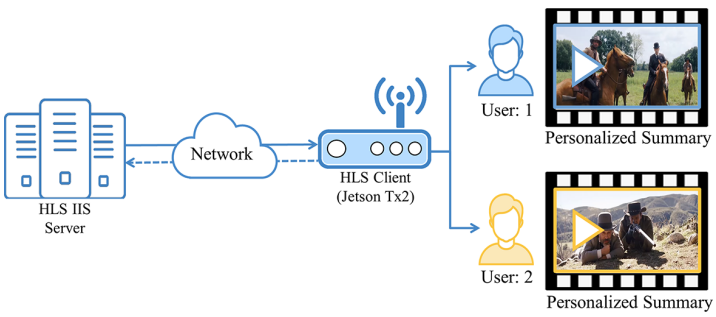

Fig. 6.1 Conceptual diagram of the proposed video summarization framework.

Extensive quantitative experiments show that the proposed method is more computationally efficient than the SoA baseline methods on the same client devices configurations. The qualitative evaluations are conducted with the collaboration of 56 participants.

Figure 6.2 illustrates the high-level system architecture of the proposed lightweight video summarization framework. The system comprises two main parts: the *HLS server* and the *HLS client*. Chapter 3 provides the details of the HLS server and common modules of the HLS client. The configuration and role of the event recognition and the *Seg*s summarizer modules in the proposed framework are described in the following section.

6.2.1 Events Recognition Module

The main task of the thumbnail container analyzer is to detect the preferred events from *Thum*s. Then, based on the selected thumbnails, generate a list of personalized *Seg*s and use it to produce a personalized summary. For this purpose, a lightweight 2D CNN model is designed that can detect personalized events from each thumbnail. To detect the personalized thumbnail based on preferred events of the user from each *Thum* with high accuracy, the CNN model has to be trained using thousands of images, which requires high processing GPU power. In this context, transform learning [74] is useful, where a pre-trained model is used for other purposes. This method is applied to train the EfficientNet-B0 [75], which is trained on a large-scale ImageNet [66] dataset to extract the frame-level features of each thumbnail. Compared to ConvNets, EfficientNet outperforms state-of-the-art architectures on ImageNet and has fewer parameters and FLOPS [75]. This makes EfficientNet [75] a suitable candidate for detecting personalized events from lightweight *Thum*s. The backbone of the proposed network is based on the EfficientNet-B0 [75]. Figure 6.3 shows the proposed action recognition model used to process all *Thum*s and detect personalized events. To improve the performance of the proposed network, the attention module is used, which is derived from vortex pooling [67]. The attention module is more effective by using multi-branch 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 proposed 2D CNN model is trained on the UCF101 dataset, which is a well-known action recognition dataset [57]. It consists of 13320 videos taken from YouTube, which are divided into 101 action categories [57]. In the proposed approach, data augmentation is applied [76] to reduce

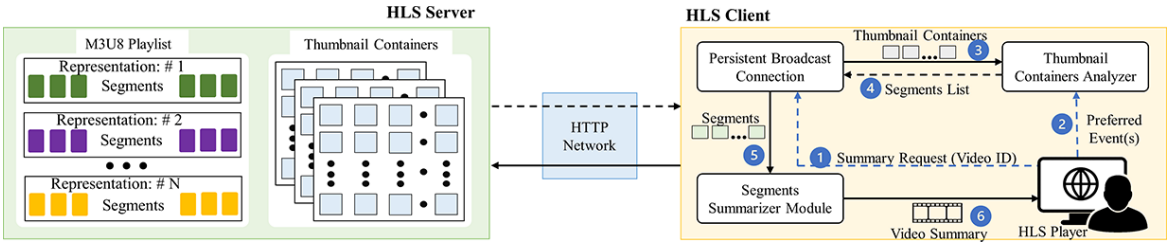

Fig. 6.2 High-level system architecture of the proposed framework.

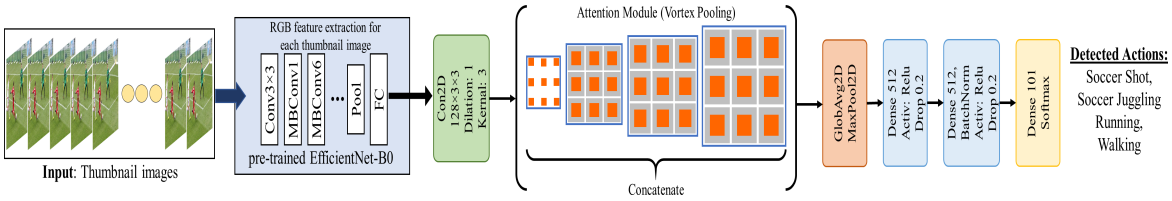

Fig. 6.3 Proposed 2D CNN action recognition model.

overfitting; this method has been proven to be very effective. To train the model using the UCF101 dataset, each video is subsampled down to 40 frames. Before being provided as input to the network, all images are preprocessed by first cropping the center region, and then they have been resized to 244×244 pixels. A shear transformation is also performed with 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 train/test, as suggested in [57]. The model is trained using a variant of SGD optimizer with momentum of 0.9 and learning rate of 0.01 using the default weight decay value (SGDW) [77]. In the experiments, the early stopping mechanism is applied in the training process with a patience of ten epochs. The Keras toolbox is used for deep feature extraction, and a GeForce RTX 2080 Ti GPU is used for implementation. The training data are fed in mini-batches with a size of 32 and a learning rate of 0.001 for cost minimization, and there are one thousand iterations for learning the sequence patterns in the data. The action recognition accuracy analysis of the model is presented in Section 6.5.1. In the following, the third component of the HLS client is described, which is the *Seg*s summarizer module.

6.2.2 Segments Summarizer Module

The purpose of the summarizer on the client is to aggregate all the downloaded personalized *Seg*s into a single continuous video stream using FFmpeg [48]. 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*s, a module can be integrated into the system, which manages the summary length according to user preference. The web-based HLS video player is explained in the following section.

6.3 Video Summarization Process

This section provides the complete flow of the proposed video summary generation process from the user perspective. In the following, this flow is described based on a set of eighteen video titles used for the experiments. A complete description of the set of videos is provided in Table 6.5. The genre of the cinematographic movies and documentaries which are analyzed are Western, sport, and

Fig. 6.4 The selected events used in video summarization process.

action[1]. 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 eighteen video titles with different playtimes and each with the frame size of 640×480 pixels.

Depending on the video genre selected by the user, they are 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 6.5, 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 are selected and categorized based on the genre of the video title. Figure 6.4 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 *ThmCon*s of the corresponding video from the HLS IIS server. Note that the downloaded *ThmCon*s cover the entire length of the video, and a very low bitrate is required to transmit all the *ThmCon*s from the server to the client. This is because *ThmCon* tends to be lightweight in terms of size as well as small in numbers compared with the frames of the same video (refer to Table 6.5 for quick comparisons).

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

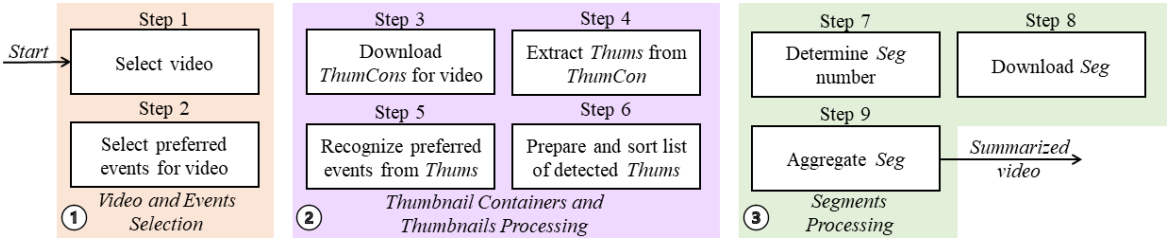

Fig. 6.5 Steps required in video summarization process.

After obtaining all the *ThmCon*s, the system extracts *Thum*s from *ThmCon*s, and the pre-trained 2D CNN model proceeds by analyzing all of them based on preferred event(s) of the user. All the *Thum*s relevant to the preferred event(s) are shortlisted. Based on the shortlisted *Thum*s, the system generates a text-based list of detected thumbnails in a 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 e personalized summary of the requested video. The text-based list of detected *Thum*s is 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 it 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 [48], in which a user can watch using the web-based HLS video player interface. The above-described flow of the proposed thumbnail-based summarization process is illustrated in Figure 6.5.

6.4 Baseline Methods

In this subsection, the baseline approaches are described for comparison with the proposed thumbnail-based method. Here, some of well-known video summarization approaches process every frame of the corresponding video to generate a summary. Thus, some of the prominent SoA techniques are adopted as baseline approaches in this study. All videos in Table 6.5 are stored locally, to compare computational efficiency and generate the summary using baseline methods. The point to note here is that storing the entire video or its data (frames) locally is not as storage efficient as the proposed LTC-SUM method. The baseline approaches as follows:

- **HECATE [5].** It analyzes aesthetic features from all temporarily extracted frames of the corresponding video. This method only supports fixed summary lengths, here the five minutes subset is generated for each video.
- **DR-DSN [7].** It is trained in the SumMe dataset [78] using the default parameters. Initially, this method extracts frames from the corresponding video and then analyzes those extracted frames to generate a video summary. Using DR-DSN with the default parameters, the summarization duration generated for all corresponding videos is 22 seconds.
- **VASNet [9].** Similar to DR-DSN [7], it is also trained the SumMe dataset [78] using the 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 default parameters of VASNet.
- **AC-SUM-GAN [10].** This is a recently published video summarization method on TCSVT. Similar to mentioned baseline methods [7, 9], it first extracts frames from the corresponding video and then analyzes extracted frames to generate a subset. It is trained on SumMe dataset

[78], and it generates the 19-second summary for every corresponding video with the default configuration.

- **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 are extracted using FFmpeg [48] for each video. The rest of the summation process follows the same steps as the proposed LTC-SUM method.

As highlighted before, there is a redundancy in frames while processing 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 are compared with the proposed LTC-SUM approach in the following experiments.

6.5 Experiments

In this section, an extensive experimental evaluation of the proposed approach is presented. First, 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.

6.5.1 Action Recognition Model

In this subsection, the accuracy of the proposed action recognition 2D CNN model is evaluated using *UCF101* [57], a benchmark action recognition dataset. To the best of our knowledge, the best thumbnail-based approach is proposed in [12]. We compare our results compared with those of [12]. The proposed model reports the highest validation accuracy of 77.81% in 36 epochs within 55.74 million flops. The proposed method achieves an increase of 4.06% in the validation accuracy, increasing from 73.75% [12] to 77.81%, and the training accuracy increases from 91.41% [12] to 96.06%. The researchers [12] used InceptionV3, which has 24M parameters; however, EfficientNet-B0 is used as the backbone network in the proposed LTC-SUM method, which has 6.9M parameters. The comparisons with other methods are summarized in Table 6.1.

6.5.2 Quantitative Evaluation

This subsection compares the performance of the proposed LTC-SUM video summarization method with the baseline schemes. The performance evaluation experiments used ten different events to analyze the proposed LTC-SUM and baseline approaches. The list of events is described in Section 6.3. All the detected thumbnails for proposed LTC-SUM method and frames for FB-SUM baseline method are included in the summarization process for which the detection accuracy is higher than

Table 6.1 Comparison of average recognition score of the action recognition proposed method with other methods.

Methods	UCF101
Karpathy, Andrej, et al. 2014. [60]	65.4%
Murthy, OV Ramana, et al. 2015 [59]	72.8%
Liu, An-An, et al. 2016 [79]	76.3%
Shu, Yu, et al. 2018 [72]	76.07%
Mujtaba, et al. 2020 [12]	73.75%
Mujtaba, et al. 2021 [80]	76.25%
Proposed	**77.81%**

95% for Western, 65% for action, and 80% for cricket sports, 90% for soccer sports videos[2]. The threshold of each video genre is selected to maintain the length of summaries. Meanwhile, the default parameters are used for rest of baseline methods.

The computation time required (minutes) to generate a video summary using the baseline and the proposed approaches are compared in the first experiment, where all approaches are configured on the HCR device (refer to Table 3.1 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 *ThmCon*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 [5], DR-DSN [7], VASNet [9], and AC-SUM- GAN [10] baseline approaches to generate summaries. Compared with the number of frames, the number of lightweight thumbnail images is significantly smaller (Table 6.5). Thus, the overall computation time of the proposed LTC-SUM method is significantly low compared with that of the FB-SUM baseline approach.Table 6.2 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.

Since this study focuses on generating summaries at the client end, in the next experiment, the proposed LTC-SUM method is configured on the LCR device (i.e., Nvidia Jetson TX2). Table 6.2 lists the computation time required in minutes to generate a summary using the FB-SUM baseline and proposed LTC-SUM methods on the HCR and LCR devices.

Table 6.3 depicts the duration of summary generated according to detected frames/thumbnails for the corresponding video using FB-SUM and LTC-SUM methods on the LCR device. From Table 6.2 it can be observed that enormous computation time is needed to process every frame of the video. In addition, it can be also observed from Table 6.2 that even when the proposed technique is implemented

[2]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 6.2 The total computation time required to generate the summary using baseline and proposed LTC-SUM approaches on HCR and LCR devices.

S/N	HECATE [5]	DR-DSN [7]	VASNet [9]	AC-SUM-GAN [10]	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**

on the LCR device, the computation time is still significantly low compared with the FB-SUM and HECATE [5] baseline approaches implemented on the HCR device.

Considering that the combined duration of all videos was 1,974 min, HECATE [5], DR-DSN [7], VASNet [9], AC-SUM-GAN [10], and FB-SUM baseline approaches took 595.789 min, 107.28 min, 105.96 min, 104.34 min and 1536.09 min using computational resources of HCR device to generate the 18 summaries for each video as shown in Table 6.2, 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, on average, the proposed approach was 13.59 HECATE [5], 2.45 DR-DSN [7], 2.42 VASNet [9], 2.38 AC-SUM-GAN [10] times faster than all baseline approaches on HCR device. The computational resources of LCR device are very low compared to HCR device, even the proposed method is 3.57 HECATE [5], and 9.2 FB-SUM method faster than baseline approaches on LCR device. In conclusion, these results show that the proposed method is extremely computationally efficient even on the LCR device.

Note that the proposed approach is also efficient in terms of communication and storage compared with the baseline approaches. As in the baseline approaches, the complete video needs to be downloaded and stored while, in the case of the proposed approach, only the $ThmCon$s are downloaded and stored. Thus, compared with the complete video, the download time and storage requirement

Table 6.3 Duration of generated video summaries in the second experiment for proposed thumbnail-based method.

S/N	# Detected frames/*Thum*		# *Seg*s 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**

for *ThmCon*s are significantly less. For example, the size of the movie 89 (2017) is approximately 612 MB, while the size of the *ThmCon*s of the corresponding movie is just approximately 14 MB. In addition, DR-DSN [7], VASNet [9], AC-SUM-GAN [10], and FB-SUM baseline approaches need to store original video along with extracted frames during the summarization process. By comparing number of thumbnails with number of frames in the a video, the quantity of frames is very large. Thus, the proposed LTC-SUM method is extremely efficient during the summarization process in terms of communication and storage.

From Tables 6.2-6.3, it can be concluded that the proposed approach is better than the baseline approaches in terms of low computational complexity and computation time. This superiority in exists even when the proposed approach is configured on a significantly lower computational resources device (Nvidia Jetson TX2). Interestingly, the duration of the summaries generated using the proposed approach was much smaller than the FB-SUM baseline approach (refer to Table 6.2). It is intuitive to ask what the impact of the significant reduction in computational time and the small duration of video summaries on the quality is. In the following, the results of a comprehensive qualitative survey are presented to answer this question.

6.5.3 Qualitative Evaluation

This section presents the 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. Since this paper has focused primarily on personalized summaries, so only the FB-SUM baseline approach is evaluated. The evaluation is based on a survey conducted with the help of 56 participants; 44 males and 12 females, and the age range was 15–35 years, (i.e., most respondents are young). The participants are from nine different geographical locations and covered a wide range of professions; however, most respondents' are researchers and faculty.

The survey is based on 18 movies (refer to Table 6.5), depending on the genre of the video, a list of options of 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 are (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 are divided into two categories (soccer and cricket). The selected options of event(s) for soccer genre videos are (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[3].

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 are generated using the proposed LTC-SUM and FB-SUM baseline techniques. The performance of the generated video summaries was objectively evaluated using the exact rating scale. The participants are asked to is the summary, which is considered better according to the three evaluation criteria: information coverage, visual pleasure, and general satisfaction. We create an anonymization questionnaire for the generated summaries so that the users could not determine which method (i.e., LTC-SUM or FB-SUM) is used.

[3]Note that the proposed technique is not limited to the above-mentioned list of preference options for each video. For simplicity, we have 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 6.4 Average rating (1∼10) of the baseline and proposed 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, please.	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**

Fig. 6.6 Illustration of frame samples obtained from generated video summaries.

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 6.4 lists the questions and the average rating given by the participant for each question for both approaches.

Despite the fact that the summary generated using the proposed approach is short and required less computation time as the entire analysis is based only on thumbnails, the qualitative evaluation suggests that the proposed approach is 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. It can be observed from Q3–Q4 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 as the average rating is 6.89. Based on this qualitative evaluation, it can be concluded that the proposed approach performs well and receives higher average ratings compared with the FB-SUM baseline without losing significant important information (e.g., preferred events). Figure 6.6 depicts sample frames obtained from the video summary generated using the proposed LTC-SUM method.

6.6 Discussion

This Chapter presents a unique thumbnail-based video summarization framework that requires the client device computational resources for the entire process. It processes lightweight thumbnail containers to generate a subset of the corresponding video, which improves computational efficiency and reduces the demand for various resources. This capability can be significantly useful for resource-contained and LCR devices. Extensive experiments on eighteen videos showe that the proposed

method is 35 times more computationally efficient than the baseline for the same HCR device configuration. Meanwhile, it is 9.2 times more computationally efficient on Nvidia Jetson TX2, which has considerably fewer computational resources, than the baseline on the HCR device. Qualitative results demonstrate that the proposed approach outperforms the baseline approach, and it has also received higher average ratings without losing significant important information. In the future, we intend to integrate the proposed method into other streaming protocols. Memory networks such as LSTM can be used to extend the proposed in the future.

Table 6.5 Details of video titles used for analysis in the proposed video summarization approach.

S/N	Title	Genre	IMDB	Duration	FPS	# Frames	# *ThmCon*	# Thumbnails
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 And 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! The Dream Begins (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 The Untold Story (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

CHAPTER 7

DISCUSSION AND CONCLUSION

This dissertation addresses three main challenges in personalized media generation. The first is to reduce computational complexity and processing time during the multimedia content generation process. The second is to use the minimum computational resources of the user-end device appropriately in the process. Third, it alleviates privacy issues while designing client-driven solutions, such as generating personalized media concurrently for each user according to their preferences. This dissertation proposes three novel client-driven multimedia generation methods to solve these challenges. This Chapter summarizes all three methods with overall conclusions and discusses further research opportunities.

7.1 Discussions

This section presents the overall discussion and contributions of this dissertation. This dissertation have proposed three different client-driven personalized multimedia generation frameworks. The details are described in the subsequent sections.

7.1.1 Personalized Movie Trailer Discussion

A new lightweight way to facilitate the trailer generation process is presented in Chapter 4. Instead of the entire video data, it analyzes a lightweight thumbnail container to detect personalized events. This makes the overall trailer generation process computationally and communicationally efficient. Quantitative and qualitative evaluations are conducted on 25 broadcast movies to find the effectiveness of the proposed method. The evaluation results show that the proposed method requires less processing time compared to the baseline approach and has received higher ratings than official trailers. The official trailers for the two movies in the dataset cannot be identified, so it is presumed official public trailers are unavailable online. In this case, the proposed method can assist in the trailer generation process. This study focus on the western and sports genres of the trailer generation process. However, it can be easily adapted to other genres along with the privacy-preserving solution. The proposed

client-driven approach is responsive, scalable, and supports a wide range of end-user hardware devices with a variety of computing capabilities.

7.1.2 Personalized GIF Generation Discussion

An innovative lightweight GIF generation technique for full soccer videos is presented in Chapter 5. It analyzes thumbnail containers of the corresponding video to obtain events such as penalty shootout. Instead of full-length video, it uses small video segments to generate animated GIFs. This improves computational efficiency and reduces the demand for communication and storage resources. Extensive experimental results obtained based on a set of six videos show that the proposed approach is 2.33 and 3.57 times faster than the current state-of-the-art method on HCR and LCR devices, respectively. Qualitative results indicate that the proposed method outperformed the existing methods and received a higher overall rating. The proposed approach can be extended by adapting to other sports categories by considering different events and datasets.

7.1.3 Personalized Video Summarization Discussion

The keyshot video summarization method is presented in Chapter 6. It processes lightweight thumbnail containers to generate a subset of the corresponding video, which improves computation efficiency and reduces the demand for various resources. This capability can be significantly useful for resource-contained and LCR devices. Extensive experiments on eighteen videos show that the proposed method is 35 times more computationally efficient than the baseline for the same HCR device configuration. On the other hand, the Nvidia Jetson TX2, which has significantly fewer computational resources, is 9.2 times more computationally efficient than the baseline for HCR devices. Qualitative results indicate that the proposed approach is superior to the baseline approach and have received a higher average rating without losing important information. In the future, we intend to extend our method by integrating other streaming protocols such as DASH along with Memory networks such as LSTM.

7.2 Conclusion and Future Insight

This dissertation has proposed a lightweight client-driven personalized multimedia generation framework for streaming platforms while reducing computational and privacy bottlenecks. Computational resources of end-user devices are used to simultaneously generate diverse personalized multimedia content for concurrent users. It handles the complex process of detecting personalized events (such as penalty shoot-outs of soccer videos) from lightweight thumbnails. Analyzing the lightweight thumbnail containers instead of the entire video data has significantly reduced the overall computational complexity. The proposed framework is also efficient in terms of communication (between server and client) and storage requirements because the entire video does not have to be transmitted over the network and stored locally. The framework is designed to manage a wide range of end-

user resource-constrained hardware platforms with heterogeneous computing, networks, and storage capabilities.

Three different client-driven techniques are designed for streaming platforms to validate the proposed framework. The first proposed method is designed to facilitate and expedite personalized trailers generation in the film-making process. The second proposed method is designed to generate personalized animated GIFs for full-length sports videos. Finally, the third proposed method is designed to produce personalized keyshot-based video summaries for different video categories such as documentaries, movies, and sports matches. Extensive quantitative experiments showed that the proposed frameworks are significant computationally efficient than the SoA methods on similar client device specifications. To the best of our knowledge, all three proposed frameworks are the first-ever client-driven approaches for streaming platforms that analyzes thumbnail containers to create personalized multimedia content such as a trailer, animated GIF, and video summary.

During the generation of multimedia content using the proposed thumbnail-based method, it is observed that some images (frames) in the generated media (GIFs, summaries, and trailers) segments are irrelevant according to selected events. The video player needs to decode the entire video segment even if it contains some related images. As a result, segment decoding and encoding require enormous computational resources. This issue can be resolved in the proposed method using entry points and exit points (timestamps) to retrieve only the relevant images within the segment. The decoder can decode all the video frames in the segment for in-image decoding but displays the video frames according to the entry and exit point information. Extensive video processing will not be required by adapting to the timestamp information on the decoder side.

ATSC 3.0 sets the standard for new media services that include new technologies such as run-time environments, broadcaster applications, and companion screens. Application-based services are one of ATSC 3.0's main service categories, enabling them to offer many useful features beyond linear broadcast services. Based on these application-based extensions, over-the-top (OTT) services can be designed that provide a variety of interactive personalized services. In the real-world scenario, generating and managing personalized media with a time machine in a set-top box is an extraordinary challenge to investigate.

In real-time application scenarios, the proposed method can be adapted to generate personalized multimedia on the smartphone or the set-top box. For example: if the smartphone is in fully charged mode, it uses the device's computing resources from a specific date to analyze your photos/videos and generate the 'memories' video summary. Personalized media can be generated similarly using the proposed method on the smartphone. Additionally, it is compelling to focus on designing an innovative framework that can generate multimedia content using recommendation algorithms on end-user devices. The client-based personalized multimedia generation technology is in the early stage new methods for different scenarios are open for academic research to investigate.

REFERENCES

[1] Sarah Min. Coming soon to netflix: Movie trailers crafted by ai., 2019.

[2] Bart P Knijnenburg, Martijn C Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. Explaining the user experience of recommender systems. User Modeling and User-Adapted Interaction, 22(4-5):441–504, 2012.

[3] U Cisco. Cisco annual internet report (2018–2023) white paper, 2020.

[4] Fernando Amat, Ashok Chandrashekar, Tony Jebara, and Justin Basilico. Artwork personalization at netflix. In Proceedings of the 12th ACM conference on recommender systems, pages 487–488, 2018.

[5] Yale Song, Miriam Redi, Jordi Vallmitjana, and Alejandro Jaimes. To click or not to click: Automatic selection of beautiful thumbnails from videos. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, CIKM '16, page 659–668, New York, NY, USA, 2016. Association for Computing Machinery.

[6] C. Huang and H. Wang. A novel key-frames selection framework for comprehensive video summarization. IEEE Transactions on Circuits and Systems for Video Technology, 30(2):577–589, 2020.

[7] Kaiyang Zhou, Yu Qiao, and Tao Xiang. Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In AAAI, 2018.

[8] J. Lei, Q. Luan, X. Song, X. Liu, D. Tao, and M. Song. Action parsing-driven video summarization based on reinforcement learning. IEEE Transactions on Circuits and Systems for Video Technology, 29(7):2126–2137, 2019.

[9] Jiri Fajtl, Hajar Sadeghi Sokeh, Vasileios Argyriou, Dorothy Monekosso, and Paolo Remagnino. Summarizing videos with attention. In Asian Conference on Computer Vision, pages 39–54. Springer, 2018.

[10] 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 Transactions on Circuits and Systems for Video Technology, pages 1–1, 2020.

[11] Blog Google. Shortform and longform videos, 2020.

[12] Ghulam Mujtaba and Eun-Seok Ryu. Client-driven personalized trailer framework using thumbnail containers. IEEE Access, 8:60417–60427, 2020.

[13] Saeideh Bakhshi, David A Shamma, Lyndon Kennedy, Yale Song, Paloma De Juan, and Joseph'Jofish' Kaye. Fast, cheap, and good: Why animated gifs engage us. In Proceedings of the 2016 chi conference on human factors in computing systems, pages 575–586, 2016.

[14] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. Learning spatiotemporal features with 3d convolutional networks. In Proceedings of the IEEE international conference on computer vision, pages 4489–4497, 2015.

[15] 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 CVPR 2011, pages 3361–3368, 2011.

[16] Karen Simonyan and Andrew Zisserman. Two-stream convolutional networks for action recognition in videos. Advances in neural information processing systems, 27:568–576, 2014.

[17] C. Feichtenhofer, A. Pinz, and R. P. Wildes. Spatiotemporal multiplier networks for video action recognition. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 7445–7454, 2017.

[18] Yi Zhu, Xinyu Li, Chunhui Liu, Mohammadreza Zolfaghari, Yuanjun Xiong, Chongruo Wu, Zhi Zhang, Joseph Tighe, R Manmatha, and Mu Li. A comprehensive study of deep video action recognition. arXiv preprint arXiv:2012.06567, 2020.

[19] Ke Yang, Xiaolong Shen, Peng Qiao, Shijie Li, Dongsheng Li, and Yong Dou. Exploring frame segmentation networks for temporal action localization. Journal of Visual Communication and Image Representation, 61:296–302, 2019.

[20] Fabian Caba Heilbron, Wayner Barrios, Victor Escorcia, and Bernard Ghanem. Scc: Semantic context cascade for efficient action detection. In 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 3175–3184. IEEE, 2017.

[21] Wenjie Pei, Tadas Baltrusaitis, David MJ Tax, and Louis-Philippe Morency. Temporal attention-gated model for robust sequence classification. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 6730–6739, 2017.

[22] Sebastian Agethen and Winston H Hsu. Deep multi-kernel convolutional lstm networks and an attention-based mechanism for videos. IEEE Transactions on Multimedia, 22(3):819–829, 2019.

[23] Yuxin Peng, Yunzhen Zhao, and Junchao Zhang. Two-stream collaborative learning with spatial-temporal attention for video classification. IEEE Transactions on Circuits and Systems for Video Technology, 29(3):773–786, 2018.

[24] Chong-Wah Ngo, Yu-Fei Ma, and Hong-Jiang Zhang. Video summarization and scene detection by graph modeling. IEEE Transactions on Circuits and Systems for Video Technology, 15(2):296–305, 2005.

[25] J. You, G. Liu, L. Sun, and H. Li. A multiple visual models based perceptive analysis framework for multilevel video summarization. IEEE Transactions on Circuits and Systems for Video Technology, 17(3):273–285, 2007.

[26] C. Weng, W. Chu, and J. Wu. Rolenet: Movie analysis from the perspective of social networks. IEEE Transactions on Multimedia, 11(2):256–271, 2009.

[27] H. Salamin, S. Favre, and A. Vinciarelli. Automatic role recognition in multiparty recordings: Using social affiliation networks for feature extraction. IEEE Transactions on Multimedia, 11(7):1373–1380, 2009.

[28] Jitao Sang and Changsheng Xu. Character-based movie summarization. In Proceedings of the 18th ACM International Conference on Multimedia, MM '10, page 855–858, New York, NY, USA, 2010. Association for Computing Machinery.

[29] G. Evangelopoulos, A. Zlatintsi, A. Potamianos, P. Maragos, K. Rapantzikos, G. Skoumas, and Y. Avrithis. Multimodal saliency and fusion for movie summarization based on aural, visual, and textual attention. IEEE Transactions on Multimedia, 15(7):1553–1568, 2013.

[30] Marta Aparício, Paulo Figueiredo, Francisco Raposo, David Martins de Matos, Ricardo Ribeiro, and Luís Marujo. Summarization of films and documentaries based on subtitles and scripts. Pattern Recognition Letters, 73:7 – 12, 2016.

[31] Ying Li, Shih-Hung Lee, Chia-Hung Yeh, and C. . J. Kuo. Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques. IEEE Signal Processing Magazine, 23(2):79–89, 2006.

[32] Mehdi Ellouze, Nozha Boujemaa, and Adel M. Alimi. Im(s)2: Interactive movie summarization system. Journal of Visual Communication and Image Representation, 21(4):283 – 294, 2010.

[33] W. 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 Transactions on Multimedia, 13(3):539–550, 2011.

[34] Xu Chen, Yongfeng Zhang, Qingyao Ai, Hongteng Xu, Junchi Yan, and Zheng Qin. Personalized key frame recommendation. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '17, page 315–324, New York, NY, USA, 2017. Association for Computing Machinery.

[35] S. S. Thomas, S. Gupta, and V. K. Subramanian. Context driven optimized perceptual video summarization and retrieval. IEEE Transactions on Circuits and Systems for Video Technology, 29(10):3132–3145, 2019.

[36] M. Ma, S. Mei, S. Wan, Z. Wang, D. D. Feng, and M. Bennamoun. Similarity based block sparse subset selection for video summarization. IEEE Transactions on Circuits and Systems for Video Technology, pages 1–1, 2020.

[37] Y. Yuan, T. Mei, P. Cui, and W. Zhu. Video summarization by learning deep side semantic embedding. IEEE Transactions on Circuits and Systems for Video Technology, 29(1):226–237, 2019.

[38] Z. Ji, K. Xiong, Y. Pang, and X. Li. Video summarization with attention-based encoder–decoder networks. IEEE Transactions on Circuits and Systems for Video Technology, 30(6):1709–1717, 2020.

[39] Weixuan Chen, Ognjen Oggi Rudovic, and Rosalind W Picard. Gifgif+: Collecting emotional animated gifs with clustered multi-task learning. In 2017 Seventh International Conference on Affective Computing and Intelligent Interaction (ACII), pages 510–517. IEEE, 2017.

[40] Brendan Jou, Subhabrata Bhattacharya, and Shih-Fu Chang. Predicting viewer perceived emotions in animated gifs. In Proceedings of the 22nd ACM international conference on Multimedia, pages 213–216, 2014.

[41] Jialun" Aaron" Jiang, Casey Fiesler, and Jed R Brubaker. 'the perfect one' understanding communication practices and challenges with animated gifs. Proceedings of the ACM on human-computer interaction, 2(CSCW):1–20, 2018.

[42] Tianliang Liu, Junwei Wan, Xiubin Dai, Feng Liu, Quanzeng You, and Jiebo Luo. Sentiment recognition for short annotated gifs using visual-textual fusion. IEEE Transactions on Multimedia, 22(4):1098–1110, 2020.

[43] Ghulam Mujtaba, Jaehyoun Kim, and Eun-Seok Ryu. Client-driven animated gif generation framework using an acoustic feature. Multimedia Tools and Applications, 2021.

[44] Alan F Smeaton, Bart Lehane, Noel E O'Connor, Conor Brady, and Gary Craig. Automatically selecting shots for action movie trailers. In Proceedings of the 8th ACM international workshop on Multimedia information retrieval, pages 231–238. ACM, 2006.

[45] John R Smith, Dhiraj Joshi, Benoit Huet, Winston Hsu, and Jozef Cota. Harnessing ai for augmenting creativity: Application to movie trailer creation. In Proceedings of the 25th ACM international conference on Multimedia, pages 1799–1808. ACM, 2017.

[46] Yoshihiko Kawai, Hideki Sumiyoshi, and Nobuyuki Yagi. Automated production of tv program trailer using electronic program guide. In Proceedings of the 6th ACM international conference on Image and video retrieval, pages 49–56. ACM, 2007.

[47] Mike O Leary. Iis iis iis and modsecurity. In Cyber Operations, pages 789–819. Springer, 2019.

[48] FFmpeg. Ffmpeg github page, 2018.

[49] M Thomson. Hypertext transfer protocol version 2, 2013.

[50] Christopher Mueller, Stefan Lederer, Christian Timmerer, and Hermann Hellwagner. Dynamic adaptive streaming over http/2.0. In 2013 IEEE International Conference on Multimedia and Expo (ICME), pages 1–6. IEEE, 2013.

[51] Video Dev. Javascript hls client using media source extension, 2020.

[52] Amy He. Time flies: U.s. adults now spend nearly half a day interacting with media., 2018.

[53] Megan Mahoney. Just the stats: Why should you leverage video marketing., 2019.

[54] Shuchita Godbole. I believe netflix needs social features — and users agree!., 2019.

[55] Sitaram Asur and Bernardo A Huberman. Predicting the future with social media. In Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology-Volume 01, pages 492–499. IEEE Computer Society, 2010.

[56] Digital Spy. Sonic the hedgehog movie has been delayed following trailer criticism, 2019.

[57] Khurram Soomro, Amir Roshan Zamir, and Mubarak Shah. Ucf101: A dataset of 101 human actions classes from videos in the wild. arXiv preprint arXiv:1212.0402, 2012.

[58] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 2818–2826, 2016.

[59] O.V. Ramana Murthy and Roland Goecke. Ordered trajectories for human action recognition with large number of classes. Image and Vision Computing, 42:22–34, 2015.

[60] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. Large-scale video classification with convolutional neural networks. In 2014 IEEE Conference on Computer Vision and Pattern Recognition, pages 1725–1732, 2014.

[61] Ghulam Mujtaba, Muhammad Tahir, and Muhammad Hanif Soomro. Energy efficient data encryption techniques in smartphones. Wireless Personal Communications, 106(4):2023–2035, 2019.

[62] Calif Redwood Shores. Giphy selects oracle data cloud to measure viewability across billions of gifs, 2019.

[63] Ghulam Mujtaba, Seondae Kim, Eunsoo Park, Seunghwan Kim, Jaesung Ryu, and Eun-Seok Ryu. Client-driven animated keyframe generation system using music analysis. In Proceedings of the Korean Society of Broadcast Engineers Conference, pages 173–175. The Korean Institute of Broadcast and Media Engineers, 2019.

[64] Paige Cooper. 23 youtube statistics that matter to marketers in 2020, 2019.

[65] François Chollet. Xception: Deep learning with depthwise separable convolutions. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 1251–1258, 2017.

[66] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE conference on computer vision and pattern recognition, pages 248–255. Ieee, 2009.

[67] Chen-Wei Xie, Hong-Yu Zhou, and Jianxin Wu. Vortex pooling: Improving context representation in semantic segmentation. arXiv preprint arXiv:1804.06242, 2018.

[68] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. arXiv preprint arXiv:1711.05101, 2017.

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

[70] Andrew Howard, Mark Sandler, Grace Chu, Liang-Chieh Chen, Bo Chen, Mingxing Tan, Weijun Wang, Yukun Zhu, Ruoming Pang, Vijay Vasudevan, et al. Searching for mobilenetv3. In Proceedings of the IEEE International Conference on Computer Vision, pages 1314–1324, 2019.

[71] Gao Huang, Zhuang Liu, Laurens Van Der Maaten, and Kilian Q Weinberger. Densely connected convolutional networks. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 4700–4708, 2017.

[72] Y. Shu, Y. Shi, Y. Wang, Y. Zou, Q. Yuan, and Y. Tian. Odn: Opening the deep network for open-set action recognition. In 2018 IEEE International Conference on Multimedia and Expo (ICME), pages 1–6, 2018.

[73] N. Babaguchi, Y. Kawai, T. Ogura, and T. Kitahashi. Personalized abstraction of broadcasted american football video by highlight selection. IEEE Transactions on Multimedia, 6(4):575–586, 2004.

[74] A. S. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson. Cnn features off-the-shelf: An astounding baseline for recognition. In 2014 IEEE Conference on Computer Vision and Pattern Recognition Workshops, pages 512–519, 2014.

[75] Mingxing Tan and Quoc Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In Proceedings of the 36th International Conference on Machine Learning, pages 6105–6114, 2019.

[76] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. Commun. ACM, 60(6):84–90, May 2017.

[77] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In International Conference on Learning Representations, 2019.

[78] Michael Gygli, Helmut Grabner, Hayko Riemenschneider, and Luc Van Gool. Creating summaries from user videos. In European conference on computer vision, pages 505–520, Cham, 2014.

[79] 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 Transactions on Pattern Analysis and Machine Intelligence, 39(1):102–114, 2017.

[80] Ghulam Mujtaba and Eun-Seok Ryu. Lightweight client-driven gif generation framework for soccer videos. arXiv preprint arXiv:1212.0402, 2021.