Cricket Video Summarization Using Deep Learning

Rohit Hanumant Khalate[1] Anand Sunil Chaudhari[2] Omkar Balaso Jarad[3] Pramod Pandurang Malusare[4]

[1,2,3,4]Department of Information Technology

[1,2,3,4]Vidya Pratishthan's Kamalnayan Bajaj Institute of Engineering and Technology, Baramati, India

Abstract — In this paper, we present a novel approach to summarizing cricket match videos using deep learning techniques. Leveraging computer vision and deep learning algorithms, our methodology focuses on extracting key events such as sixes, fours, and wickets from cricket match footage. The process involves multiple stages, including frame extraction, scorecard detection, text extraction, bowler identification, and event boundary determination. We employ state-of-the-art deep learning models, such as YOLOv8, for object detection and Optical Character Recognition (OCR) for text extraction. By combining these techniques, we generate a comprehensive summary of the match, highlighting significant moments and key plays. Our approach not only automates the summarization process but also enhances the accessibility and comprehensibility of cricket match highlights for viewers. By conducting thorough experimentation and juxtaposing our approach with established methodologies, we showcase the efficacy and expediency of our innovative framework in crafting compelling and informative video synopses.

Keywords: Cricket Video Summarization, Deep Learning, Object Detection, Optical Character Recognition (OCR), YOLOv8

I. INTRODUCTION

Cricket is more than just a sport; it's a global phenomenon that entertains millions of fans worldwide. video highlights of cricket match increase the excitement of a user and saves time. While cricket highlights are immensely popular, the process of creating them manually can be time-consuming. Traditionally, highlight reels were curated by sifting through hours of match footage to identify key moments such as wickets, boundaries, and exceptional plays. However, this manual approach is time consuming and prone to human error. In recent years, advancements in artificial intelligence (AI) and computer vision technologies have revolutionized the way cricket highlights are generated. Automated highlight generation systems use algorithms to analyze match footage and automatically identify significant events in real-time. These systems use a multi-step process, starting with frame extraction to isolate individual moments from the match footage. Then, advanced object detection algorithms like YOLO v8 are employed to recognize elements such as the scorecard and bowlers. By automating the highlight generation process, AI-powered systems overcome the limitations of manual curation, ensuring that fans receive comprehensive and engaging highlight packages. These systems can accurately identify key events such as wickets, sixes, and milestones, allowing fans to watch the most thrilling moments of the match in a short format. Moreover, automated highlight generation enables broadcasters and digital platforms to deliver personalized content related to the preferences of individual viewers, thereby enhancing the overall viewing experience. In [?] presented an approach that used Yolov3 to detect scorecards. Their model only produces timestamps for key events. As an extension of this idea, we provide not just timestamps but also distinct boundaries, such as the beginning and conclusion of an event. We have trained an additional YOLO v8 model for bowler detection in order to obtain event boundaries. The presence of a bowler and a scorecard determines an event's boundaries, but score differences—such as 4,6, in runs for fours and sixes, and 1, in wickets for wicket event.

II. LITERATURE SURVEY

The author [1] introduced a player-specific framework for cricket highlight generation using deep convolutional networks. In this method, long videos are converted into short highlights for specific players. The first step in summarizing cricket videos involves extracting text summaries from score captions using optical character recognition (OCR) and identifying all frames featuring a specific player from the first frame to the last frame. Videos are typically generated at 24 frames per second. Utilizing pre-trained networks like AlexNet can expedite the classification process without compromising performance. Various frames from cricket videos are employed in the supervised training of AlexNet models for the classification of positive and negative instances. After classification, partner players are removed using fuzzy c-means clustering. An algorithm is then applied to generate bounding boxes for each person in the frame. These frames are cropped to a dimension of 137 x 197 containing the bounding boxes. The ScaleInvariant Feature Transform (SIFT) is employed to extract data from all players. Output video frames belonging to the positive class are combined to create frames specific to the player. The goal is to enhance the model's performance while reducing complexity. The authors of the paper [2] presented an approach that used Yolov3 to detect scorecards. Their model extracts frames from videos at the rate of 1 FPS. They have trained Yolov3 model for extracting scorecards from frames. They have used 1300 images to train their model. After detecting the scorecard, it was cropped. A custom OCR model was trained for extracting text from cropped scorecards. Regular expression was used on the output of OCR to give values of runs and wickets. Based on differences in runs events like four and sixes were recognized whereas wicket event was recognized by difference of 1 in wicket. Authors of [2] only produced timestamps for key events.

In [3] authors have proposed a comprehensive summarization model which begins with the extraction of excitement clips, where short-time audio energy is calculated and normalized to identify clips surpassing a predefined threshold. These clips serve as essential cues for identifying key events within cricket matches. Subsequently, the model identifies shot boundaries through hue histogram differences among adjacent frames, selecting keyframes based on surpassing a predefined threshold. This approach not only

enhances efficiency but also reduces analysis time significantly. Furthermore, the paper discusses the extraction of key events, covering a range of cricketrelated scenarios such as replays, field views, umpire actions, and player reactions. Various techniques, including scoreboard detection and gesture analysis, are employed to classify frames into different categories. Additionally, the paper introduces the HDNN EPO classifier, a Hybrid Deep Neural Network integrated with Emperor Penguin Optimization for concept annotation within exciting clips. This classifier undergoes several phases, including training, feature representation, and concept prediction, resulting in an accurate and informative summarization of cricket videos with an impressive accuracy of 93.71%.

In paper [4] authors proposed a framework to generate the highlights from broadcasted video. Author introduces an Inverse Hierarchical Framework and novel context-aware approach for event-initialization and a structural similarity Indexbased approach for event-closure detection. Author extracted the frames and then took the bowler view. To take the bowler view P-Ratio algorithm used. Author used Replaynet filtering to remove the duplicated frames. For the purpose of elimination of ball retrieval scenes a novel Structural Similarity Index (SSIM) based scene closure Algorithm used. At the end video sequences free of replays remaining to combine.

In [5] Cricket, an exciting sport, often requires brief summaries. This paper presents a novel method for automatically generating cricket highlights. Conventional manual editing methods are time-consuming. Instead, advanced algorithms are employed to identify key moments in cricket matches. These include wickets, boundaries, sixes, and other significant occurrences, enhancing the understanding of match dynamics. The framework consists of two main modules: one for video extraction and preprocessing, and another for highlight generation. Frames are extracted from videos at a rate of 25 frames per second and converted to grayscale to simplify feature extraction. The efficiency of the framework is validated using three datasets: TVsum50, SumMe, and ADL, covering various scenarios. Convolutional Neural Networks (CNNs) are trained to recognize important scenes, facilitating feature extraction and highlight generation. Validation against benchmark datasets confirms the method's accuracy, establishing it as a reliable tool for cricket highlight generation. This framework offers a promising solution, leveraging deep learning techniques to provide concise summaries that capture the excitement and key moments of cricket matches. In [?] author approach use video classification, accurately labeling different activities in videos is crucial. To achieve this, authors developed a method combining Convolutional Neural Networks (CNNs) and K-Means clustering. Their approach significantly improved accuracy compared to traditional methods. In this framework gathered 15,000 videos across 20 classes, each containing various activities. Using an Advanced Renamer tool, organized these videos into folders based on class, and named them systematically for easy identification. Long videos were trimmed, and classes were extracted to form datasets. To reduce computational load, author extracted frames from videos at intervals (every 3rd, 5th, and 7th frame). Author found that selecting every 5th frame worked best. Frames were saved and normalized.To select important frames, author employed KMeans clustering. This helped identify frames with significant content, enhancing accuracy.Author utilized CNNs to classify images based on shape, color, size, and location. Our CNN architecture included multiple layers, starting with convolution and pooling, followed by fully connected layers. Author used ReLU activation and applied weight regularization to prevent overfitting. Author experimented with different optimizers and pre-trained models (VGG16, VGG19, Inception-V3, and ResNet50). Adam optimizer and VGG16/VGG19 models yielded the best accuracy. After applying K-Means clustering, model achieved a remarkable 99 % accuracy with a five-layer CNN. This demonstrated the effectiveness of clustering in improving classification accuracy.

III. METHODOLOGY

In this research paper, we aim to extract events such as sixes, fours, and wickets from cricket match videos using computer vision and deep learning techniques. The methodology encompasses six main stages: frame extraction, scorecard extraction and detection, text extraction, bowler detection, Process summary and generating summary. Each stage involves specific techniques and tools to accomplish the task.

A. Frame extraction

The initial phase of the proposed methodology involves the extraction of individual frames from the input video file. OpenCV(Open Source Computer Vision) library has been used to extract individual frames from the video. Frames have been extracted at the rate of 1 Frame Per Second. Figure 5 shows extracted frames from input video at rate of 1 Frame Per Second.

Extracting frames from a video file involves the sequential extraction of individual images from the array of images constituting the video. Conceptually, a video file can be perceived as an array of images, wherein each image represents a frame of the video sequence. The process of frame extraction can be linked to traversing this array of images and capturing each frame iteratively. The extraction process involves placing a pointer at the beginning of the array of images, which corresponds to the starting point of the video file. At this initial position, the image pointed to by the pointer represents the first frame of the video, which is subsequently saved for further processing or analysis. Following the extraction of the current frame, the pointer is advanced by a determined interval corresponding to the frame rate of the video, denoted as Frames Per Second (FPS). The FPS value dictates the temporal spacing between consecutive frames in the video sequence. By incrementing the pointer by the FPS count, subsequent frames are accessed successively, allowing for the systematic extraction of frames from the video stream. This iterative process continues until all frames of interest have been extracted or until the end of the video file is reached.

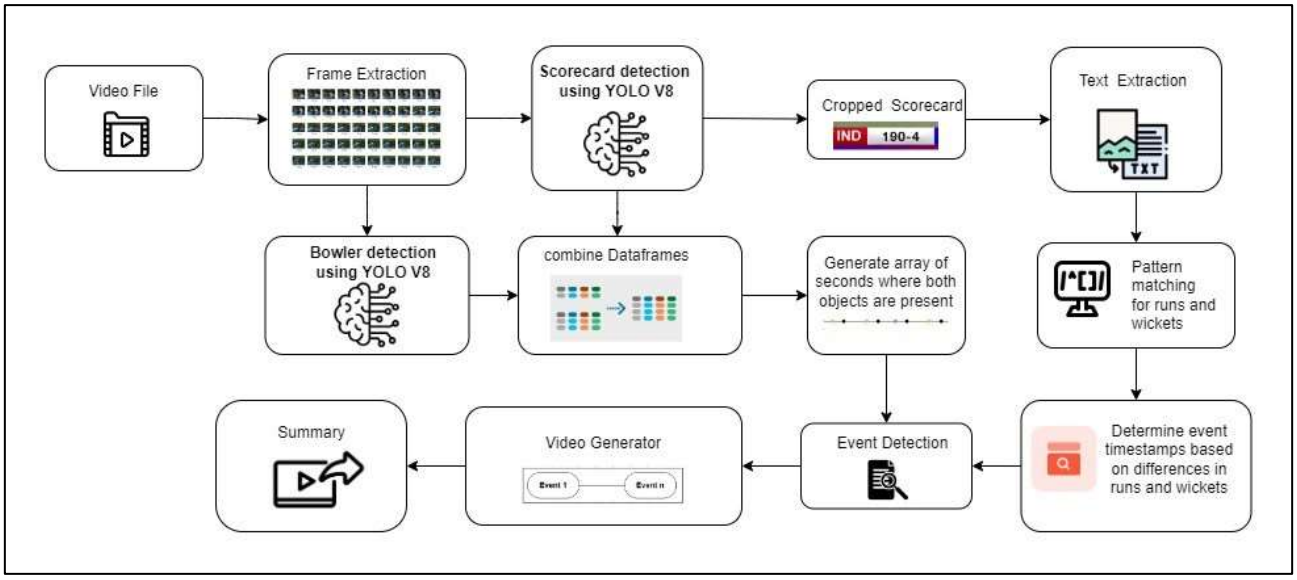

Fig. 1: System Architecture

Fig. 2: Frame Extraction

B. Scorecard Extraction and Detection

To detect and extract scorecards YOLOv8 object detection model is used. YOLOv8 is nothing but a You Only Look Once. For detecting and extracting cropped scoreboard YOLOv8 has been trained on the custom dataset. The dataset contains 1000 images of different types of scorecards used in Cricket. Annotation of images has been done using labelImg. The YOLO v8 model has been trained using several cricket matches available on YouTube. Every video has undergone the frame extraction procedure previously described, and using labelImg, the retrieved images have been annotated. Following training, the model's weights were saved in a file with extension .pt. The scorecard is detected from the extracted frame using YOLO v8 model. The scorecard's presence is saved as a dataframe for further analysis. Only two columns make up the dataframe: one is the scorecard, and the other is the frame number or seconds. In case the scorecard is available in the frame, the dataframe will be updated with the frame number and scorecard value as 1. If the scorecard is absent, the dataframe will be updated with the frame number and scorecard value as 0. The scorecard must be extracted and cropped before it can be stored in memory when it is found in a frame. The figure 5 shows the scorecard in frames 1, 2, 3, 4, and 6. The scorecard is missing from frame number 5. This phase has two outputs cropped scorecards and dataframe which contains the presence of a scorecard with a timestamp The figure below displays the cropped scorecard extraction's output. The figure 3 below displays the cropped scorecard extraction's output.

C. Text Extraction

Text from cropped scorecard as shown in figure 3 is extracted with the help of optical Character Recognition. On Each cropped scorecard OCR is applied to get the required

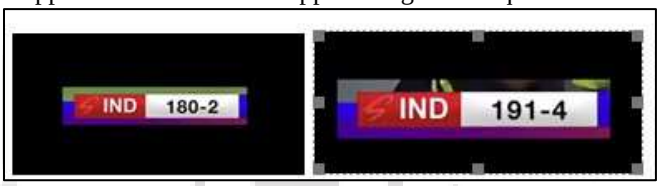

Fig. 3: cropped scorecard

secs	runs	wickets
1	180	2
5	180	3
6	130	3
23	130	3
34	120	49
36	10	4
54	180	4
55	130	4
59	130	4
64	190	4
67	190	4
68	100	4
72	190	4
74	191	4
75	11	4
77	191	4

a

Seconds	Scorecard
0	1
1	1
2	1
3	1
4	1
5	1
6	1
7	1
8	0
9	0
10	0
11	0
12	0
13	0
14	0
15	0

b

sec	scorecard	bowler	checkpoint
0	1	1	1
1	1	1	1
2	1	1	1
3	1	0	0
4	1	0	0
5	1	0	0
6	1	0	0
7	1	0	0
8	0	1	0
9	0	0	0
10	0	0	0
11	0	0	0
12	0	0	0
13	0	0	0
14	0	0	0
15	0	0	0

c

Fig. 4: Scorecard Dataframe

runs and wickets count. The resultant is saved in dataframe with respect to their timestamp. For OCR Pytesseract library has been used. Pytesseract is an Optical Character Recognition (OCR) tool used to extract the textfrom the image. Pytesseract module has 13 page segmentation mode. page segmentation mode 7 which treats input image as a single text line. Regular expression is used to find runs and wickets. Pattern used to get runs and wickets from extracted text is

\b\s*\d+\s*-\s*\d+\b.

Output of above extracted runs and wickets from each individual frame is added as record in dataframe. An example of this dataframe is shown in below figure??.

D. Bowler Detection and Combining Dataframes

Bowler detection is a crucial step toward identifying improved event borders. An additional YOLO v8 model that was trained on a specific dataset has been employed for bowler detection.Roboflow provides the dataset that was used to train the YOLOv8 model for bowler detection. 1360 images make up the Bowler dataset. From the retrieved frame, Bowler is identified. The bowler's presence is recorded as a dataframe for later processing. There are just two columns in the dataframe: bowler and set number or seconds. If the bowler is in the frame, a record with the frame number and bowler value as 1 is added to the dataframe; if the bowler is not in the frame, a record with the frame number and bowler value as 0 is added.

Fig. 5: Frame Extraction

Inner join on the frame count/seconds column is used to merge the bowler and scoreboard dataframes.

E. Processing Dataframes from Previous Steps to Generate Output

The OCR dataframe is used to identify timestamps for various occurrences based on differences in scores. For instance, a difference of 4 between two successive runs indicates a fourrun event, while a difference of 6 indicates a six-run event. Similarly, a one-wicket difference denotes a wicket event. To determine Event Boundaries combined dataframe of bowler and scoreboard dataframe is used. Based on differences in scores within the OCR dataframe, timestamps from various occurrences are retrieved. Event four is recorded if the difference between two successive runs is 4. Event six is noted if there is a six-run difference between two successive records. Similarly, if there is a one-wicket difference between two consecutive instances, that instance can be considered a wicket. Until now we have identified the type of event but boundaries of these events have not been identified yet.

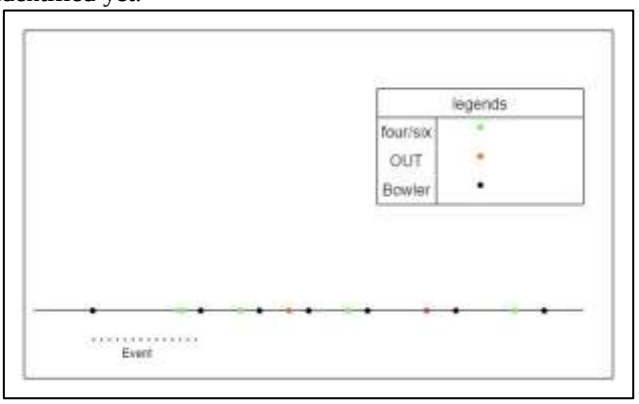

Fig. 6: Event boundary detection

A combined dataframe is used to determine the start and end, or interval, of the event. Black dots in the above figure 6 stand for simultaneous bowler and scorecard detection. Particular events, like as four, six, and wickets, are represented by green and red dots, respectively. Any event would ideally begin and end with a black dot. The idea is that every event would begin with a black point and terminate before another black point is found. The kind of dot that appears in between two successive black dots would now be used to describe the events. for example If a red dot appears in between two consecutive black dots, our needed event—a wicket—must occur, starting from the first black dot found and proceeding to its subsequent black dot.

F. VIDEO GENERATION

To generate the final video summary based on the dictionary containing timestamps of events, the Moviepy library is utilized for various operations on the video file, such as trimming and merging. Moviepy offers functionality to create subclips from a video, which can then be merged to form a cohesive video summary.

Firstly, subclips corresponding to each event timestamp generated in the previous step are created. These subclips are extracted from the original video file using the provided timestamps. Each subclip encapsulates a segment of the original video, starting from the specified timestamp and ending at the subsequent timestamp.

Additionally, transition animations are incorporated to precede each subclip, enhancing the visual appeal and providing context for the transition between different events. The choice of transition animation is determined based on the type of event associated with each subclip. For example, different animations may be employed for events such as fours, sixes, or wickets.

All subclips, along with their respective transition animations, are appended to an array or list data structure, sequentially arranging them in the desired order for the final video summary.

Finally, the individual subclips and transition animations contained within the array are concatenated or merged together to create the comprehensive video summary. The Moviepy library facilitates this merging process,

allowing for seamless integration of the subclips into a single coherent video.

IV. RESULTS

Our YOLOv8 model has successfully detected scorecards with accuracy of 0.8.

A. Dataset Description

The dataset used for evaluation consists of a 4-hour video capturing various cricket matches. The video contains a total of 40 different events, including fours, sixes, and wickets. These events serve as ground truth annotations for evaluating the performance of our methodology.

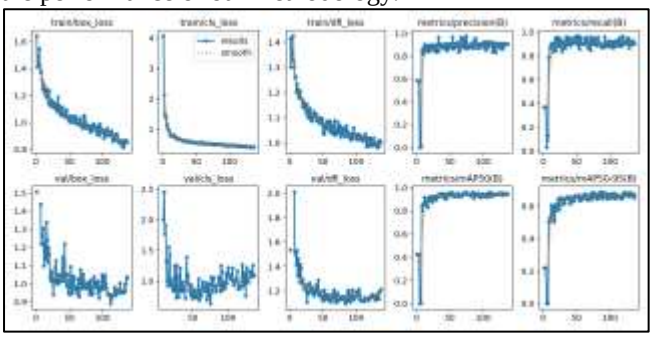

Fig. 7: Training results of YOLOv8 on scorecard datasets

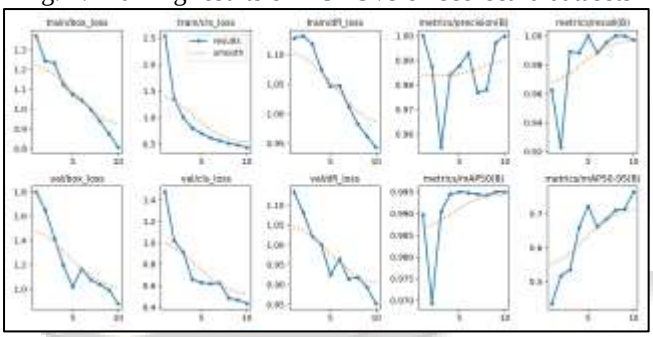

Fig. 8: Training results of YOLOv8 on bowler datasets

B. Performance Metrics

We measure the performance of our methodology using the count of correctly detected events as the primary metric. Additionally, we calculate the accuracy of our methodology, defined as the ratio of correctly detected events to the total number of events in the dataset.

Fig. 9: Sample Scorecard detection

C. Evaluation Results

To evaluate our model we manually highlighted cricket match which consisted of total 185 events like scoring 1,2,3 runs , wide balls, fours ,sixs, and wickets.Out of 185, total 40 events were fours,wickets and sixs. Our model identified 32 events with good event boundaries.

- True Positives (TP): 32 events correctly identified by the model.
- False Positives (FP): 0 non-highlight events incorrectly identified as highlights by the model. This includes all the events that the model failed to ignore as non-highlights.
- True Negatives: 0 non-highlight events correctly identified as non-highlights by the model. This includes all the events that the model correctly ignored as non-highlights.
- False Negatives: 8 events missed by the model. These are actual highlights that the model failed to detect.

Authors	Precision(%)	Recall (%)	F-Measure(%)
Hari et al. [6]	82.33	84.67	83.48
Javed et al., [7]	93.36	95.27	95.81
Javed et al., [8]	99.77	98.24	98.99
Nasir et al., [9]	93.99	87.01	90.36
Anjum et al.,[10]	68.46	75.42	71.77
K. Midhu et al., [11]	88.01	87.91	87.95
Javed et al., [12]	91.87	89.85	90.84
Javed et al. [13]	92.94	91.04	91.98
Khan et al. [14]	81.32	78.41	79.84
Shingrakhia et al. [3]	93.43	92.46	91.64
Shingrakhia et al. [15]	96.82	95.41	95.67
Proposed Model	100	80	95.68

Table I: Comparison of Different Methods

V. CONCLUSIONS

To sum it up, cricket video summarization is about making short and exciting videos of matches using computers. It's like showing all the best parts quickly without watching the whole game. Smart computer tools learn to find the most exciting moments, like wickets and big hits. Scientists are trying different ways to make sure these short videos are right. They're even making videos just for certain players. This makes watching cricket really fun and easy, and it's getting even better soon.

REFERENCES

[1] R. Mahum, A. Irtaza, S. U. Rehman, T. Meraj, and H. T. Rauf, "A player-specific framework for cricket highlights generation using deep convolutional neural networks," Electronics, vol. 12, no. 1, 2023.
[2] P. J. H. S. Chakradhar Guntuboina, Aditya Porwal, ""deep learning based automated sports video

summarization using yolo",” Electronic Letters on Computer Vision and Image Analysis, vol. 12, no. 1, 2021.
[3] H. Shingrakhia and H. Patel, “Emperor penguin optimized event recognition and summarization for cricket highlight generation,” Multimedia Syst., vol. 26, no. 6, p. 745–759, dec 2020.
[4] D. Gaikwad, S. Sarap, and D. Dhande, “Video summarization using deep learning for cricket highlights generation,” Journal of Scientific Research, vol. 14, pp. 533–544, 05 2022.
[5] S. S. Maram, N. Kumar, J. J. P. C. Rodrigues, S. Tanwar, and A. Jaink, “Images to signals, signals to highlights,” Journal Name, 2020.
[6] R. Hari and M. Wilscy, “Event detection in cricket videos using intensity projection profile of umpire gestures,” in 2014 Annual IEEE India Conference (INDICON). IEEE, 2014, pp. 1–6.
[7] A. Javed, A. Irtaza, Y. Khaliq, H. Malik, and M. T. Mahmood, “Replay and key-events detection for sports video summarization using confined elliptical local ternary patterns and extreme learning machine,” Applied Intelligence, vol. 49, pp. 2899–2917, 2019.
[8] A. Javed, K. B. Bajwa, H. Malik, and A. Irtaza, “An efficient framework for automatic highlights generation from sports videos,” IEEE Signal Processing Letters, vol. 23, no. 7, pp. 954–958, 2016.
[9] M. Nasir, A. Javed, A. Irtaza, H. Malik, and M. Mahmood, “Event detection and summarization of cricket videos,” Journal of Image and Graphics, vol. 6, no. 1, pp. 27–32, 2018.
[10] M. E. Anjum, S. F. Ali, M. T. Hassan, and M. Adnan, “Video summarization: Sports highlights generation,” in INMIC. IEEE, 2013, pp. 142–147.
[11] K. Midhu and N. Anantha Padmanabhan, “Highlight generation of cricket match using deep learning,” in Computational Vision and Bio Inspired Computing. Springer, 2018, pp. 925–936.
[12] A. Javed, K. B. Bajwa, H. Malik, A. Irtaza, and M. T. Mahmood, “A hybrid approach for summarization of cricket videos,” in 2016 IEEE International Conference on Consumer Electronics-Asia (ICCE-Asia). IEEE, 2016, pp. 1–4.
[13] A. Javed, A. Irtaza, H. Malik, M. T. Mahmood, and S. Adnan, “Multimodal framework based on audio-visual features for summarisation of cricket videos,” IET Image Processing, vol. 13, no. 4, pp. 615–622,
[14] A. A. Khan, J. Shao, W. Ali, and S. Tumrani, “Content-aware summa-rization of broadcast sports videos: an audio–visual feature extraction approach,” Neural Processing Letters, vol. 52, no. 3, pp. 1945–1968, 2020.
[15] H. Shingrakhia and H. Patel, “Sgrnn-am and hrf-dbn: a hybrid machine learning model for cricket video summarization,” The Visual Computer, vol. 38, no. 7, pp. 2285–2301, 2022.

