Automatic Generation of Cricket Highlights Framework Comprising Score Extraction and Action Recognition

*

Abhijit Saha*
Dept. of Computer Engineering
DJSCE
Mumbai, India
abhijits1802@gmail.com

Neel Kothari*
Dept. of Computer Engineering
DJSCE
Mumbai, India
neelkothariak@gmail.com

Priyansh Shah*
Dept. of Computer Engineering
DJSCE
Mumbai, India
priyanshah01@gmail.com

Rajvi Parekh*
Dept. of Computer Engineering
DJSCE
Mumbai, India
rajviaparekh@gmail.com

Narendra Shekokar
Dept. of Computer Engineering
DJSCE
Mumbai, India
narendra.shekokar@djsce.ac.in

***Abstract*—Generation of highlights is the method of pulling the most intriguing clips from a sports video. It is an important activity for sports broadcasters, especially for the sport of cricket. This research proposes a simple yet efficient approach for the automated generation of Cricket highlights from the entirety of a match video by key event detection. The proposed approach comprises of a score extraction technique with the help of the EAST model and OCR (Optical Character Recognition) techniques which helps detect the keyframes, followed by an action recognition pipeline using CNN and LSTM applied on those detected keyframes. The Observed outcomes exemplify that the precision of the proposed method for automated cricket highlights generation is productive and valuable.**

***Index Terms*—cricket highlights, key event, score extraction, optical character recognition, action recognition, computer vision, CNN, LSTM**

I. Introduction

YouTube, In July 2015 disclosed that every single minute it receives more than 400 hours of video content, and when this is translated it comes to 65.7 years' worth of content uploaded every day[1]. Technology is used more frequently than ever in the sports sector too, as sports organizations provide fresh, cutting-edge experiences to the consumers. Today, there are multiple sports played across the globe. Most of these sports are broadcast, across countries and continents garnering views from millions of viewers and producing revenue. This generates a significant amount of multimedia content—primarily videos, quickly. Each of these videos in this enormous amount of digital sports data has its share of essential intuition behind it. This calls for the processing, filtration, and reduction of these videos to generate a final sequence to be presented to the viewers.

A few examples of the application domain for automatic video summarizing include media organizations using such technologies to efficiently index, retrieve, browse, and promote their assets in media; and video sharing platforms to enhance the viewing experience, increase viewer engagement, and increase the consumption of content.[1]

The sports industry produces such kind of large-length videos, which call for summarising techniques to be applied to them. Many different key event detection and highlight generation techniques have been created by researchers for multiple sports. We focus more specifically on cricket because it is one of the most popular sports, particularly the third-most-popular, worldwide after football and basketball. Cricket has enormous viewership in many parts of the world, especially in India, the UK, and Australia. Thus, there is a huge amount of cricket multimedia produced by this widely spread-out broadcast. Major broadcasters like Sony Sports Network, Star Sports, Fox Cricket, Sky Sports, etc, have huge cricket match databases of current matches as well as from the previous years.

However, video summarizing and highlight generation for Cricket has not been explored as much as in other sports like Football, Basketball, and Tennis[2]. There are a few reasons for the same. Cricket is a game having complex rules as compared to a sport like Football. It is played over different rules, called formats, at different times which last for different periods. There are three main formats of the game, Tests, One-Day Internationals, and Twenty-20s. Test cricket is the longest format of the game which usually lasts for 4-5 days

*The authors assert equal contribution and joint first authorship.

with the fifth day being the last. Next are the ODIs (One-Day Internationals) which are shorter than Tests, but even they are played for an entire day, either from morning to evening or from afternoon to night. Each team plays an innings of 50 overs, which takes around 8 hours in all to complete. Twenty-20s, which is the most recently developed version of the game, first played in 2006, is the shortest format. T-20, as it is usually referred to, has a playing time of around three hours which is longer than a typical football game, which lasts about ninety minutes, and a hockey game, which lasts seventy minutes. As the media's target rating point (TRP) is enhanced for stations that can efficiently present the news before any competing station can, there are currently very few systems in place for the development of cricket highlights.

II. Related Work

Existing methods for video summarization can mainly be classified into two categories, namely, learning-based[3],[5] and non-learning based[6]. Learning-based methods have better accuracy but have higher complexities as compared to non-learning methods. A literature survey for both learning and non-learning methods is explained next.

Anjaneya Basappanavar et.al, in their learning-based method, has used a deep-learning-based approach to detect different phases and areas of a cricket video[3]. Their suggested technique includes two phases, a training section, and a testing section. Within the training section, the video frames are processed to extract the Discrete Wavelet rework(DWT) capabilities. Those features are then used to prepare the Probabilistic Neural Network(PNN) which then detects and classifies the cricket pitch frames, non-pitch frames, replay frames, and non-replay events. This method involves complex deep-learning concepts and is limited to detecting only a few events in a cricket video.

An approach that makes use of the visible, audio functions, and heuristic rule facts to mine the categories of events in a cricket video is proposed in [4]. The method primarily extracts the visual and audio features from the video and then detects key events based on these extricated features. Deeplai Bhawarthi in their research paper, propose discrete algorithms for the detection and identification of different events and then classify them as key or non-key events[5]. Visual features, namely, Edge Pixel Ratio (EPR), Grass Pixel Ratio (GPR), and Skin Colour Ratio (SCR) are used in these algorithms for event recognition.

For detecting crucial occurrences in a cricket match, Tang et al. suggested a multilevel framework based on Histogram of orientated Gradients (HOG) and shade Histograms (CH) [6]. Hidden Markov Models(HMM) were used to refine the events that had been observed. For highlight production of cricket matches. Namudri et al offered a MPEG-7-based approach [7]. The game was divided into video pieces, i.e., clips belonging to that particular camera shot. The extracted key frames have been divided into 5 groups based on the field perspective. The following step was to use the key-frame sequences to extract events using a Hidden Markov model (HMM).

Pushkar Shukla et.al, offer a high-level method for producing cricket highlights [8]. Their method takes under consideration both excitement-based and event-based factors to become aware of and highlight the vital moments in a cricket match. Some of the indications used to file critical moments include participant celebration, replays, playfield situations, and audio intensity. It uses learning-based techniques by making models for precise occurrences as well as audio-based capabilities.

The approach proposed by Ali Javed et.al, inspired the methodology proposed by us in this paper [2]. The method proposed, falls in the category of non-learning-based approaches. It makes use of the match score displayed in the video input. The score is extracted and if a significant change in the score is detected, a certain portion from that change point is appended in our highlight. This method approximates the segment from the key change point, which includes some supplementary parts, usually non-essential.

In comparison to the proposed literature, our presented approach is undemanding and is competent enough to handle various aspects of automated cricket highlights generation. It is a non-learning-based method in phase one, which is then acted upon by a learning-based method in the second phase.

III. Proposed Method

In this research work, we present a technique for automated cricket highlights generation from the match video. We divide the video into its constituent image frames and work on the score caption displayed to detect any significant changes. The extracted clips are then passed through our action recognition model for detecting the bowling action from that clip. These are the two main phases of the proposed work and further, the paper is laid out in two phases, i.e., Score Extraction and Action Recognition. The system was then tested on a variety of videos from different series after which the uprightness was determined. Fig.1 shows the high-level system architecture.

A. Data Description

We created 2 custom datasets named ”Bowling” and ”Non-Bowling” containing 139 and 68 videos respectively. ”Bowling” mainly consisted of bowling actions before a ball is bowled whereas ”Non-Bowling” mainly consisted of other in-match actions/activities unrelated to bowling.Also for testing purposes we created another dataset named ”Cricket” having 30 videos which consisted of the complete bowling action. Video clips are gathered from several sports broadcasters such as Star Sports, Sony Sports Network, Fox Cricket, Sky Sports, etc. The dataset contains sample videos from Sri-Lanka vs Bangladesh vs India - Nidahas Trophy 2018 , Australia vs England Ashes 2019, India vs Australia ODI Series 2018, England vs West Indies series, etc. The video clips are recorded in unique illumination situations like artificial light, daylight, and so on.

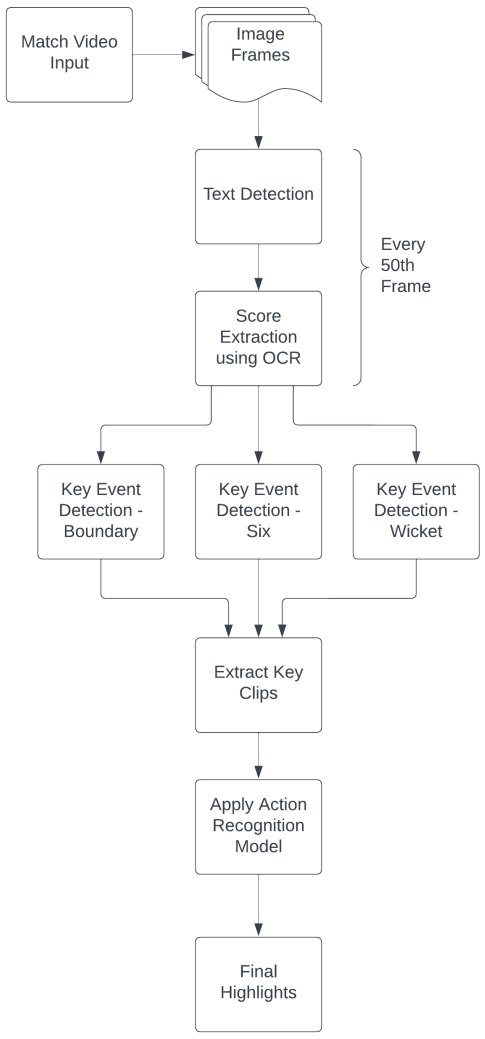

Match Video Input
Image Frames
Text Detection
Score Extraction using OCR
Every 50th Frame
Key Event Detection - Boundary
Key Event Detection - Six
Key Event Detection - Wicket
Extract Key Clips
Apply Action Recognition Model
Final Highlights

Fig. 1. System Architecture

B. Phase I.

1) Converting the video to frames: As discussed, working with videos is a taxing activity. It requires a large amount of memory and can work swiftly only on higher configuration machines. So, as opposed to studying the video enter without delay, we convert the video into its elemental image frames. In the video, each frame is captured at a fixed frame per second(fps) rate – 30 fps. The entire video is now available to us in these frames and we can apply image extraction algorithms to them. While converting the video, we also capture the timestamp of each image frame. Thus, we now have all the component image frames of the video along with their timestamps. This will be essential as we extract the important clips from the input video.

Next, we have to extract the score from these images. There are two major steps involved for the same, namely, Text Detection followed by Text Recognition. The process for it is described next. Text Detection is done using the EAST(Efficient and Accurate Scene Text Detection).

Fig. 2. Video divided into frames.

2) EAST Model:

a) Pre-processing for applying EAST model: After getting the frames, we need to make them equipped to be given to the EAST model for text detection. So, we first convert the input frames to the standard height and width to be given. The dimensions are typically saved as a multiple of 32, that's taken into consideration suitable for the model to be able to work properly. We additionally calculate the ratio between the unique picture and the resized photograph. This calculated ratio could be put into use for translating the bounding field location of the authentic image.

If the working of the EAST model is to be described at a high level, it runs through the image and creates bounding boxes in areas where it detects the presence of any text. It may discover horizontal and rotated bounding containers. It can then be utilized with any technique of text recognition. The text detection procedure in this research has excluded the redundant and intermediate steps and has 2 stages.

The first one makes use of the complete convolutional neural network to at once produce textual or word content-line degree prediction. These predictions which were produced, which may be in the form of quadrangles or rotated rectangles are then further processed via the non-max-suppression step to get the last output. Next, we construct blobs (Binary Large Objects) from the images to further forward them to the model and here our EAST model finally comes into play.

b) Text Detection using EAST model: Now we have blobs ready for the EAST model to perform its task. We are expecting two outputs from this EAST model: 1. Probability values for areas whether or not it consists of textual content. 2. Geometry of the text content – The bounding box's coordinates detecting textual content

- feature$_{fusion/concat_3}$
- feature$_{fusion/Conv_7/Sigmoid}$

Images are forward passed through the above layers. On the application of the layers, it returns two results. Firstly, it returns the bounding box where the text is detected and its corresponding probability score to check if the score is more

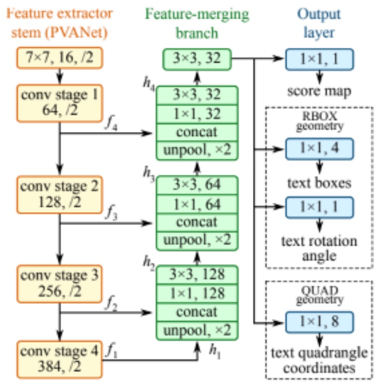


Fig. 3. Internal structure of Text Detection[9]

than the pre-fixed minimum confidence. This is called non-maxima suppression.

After examining the confidence score of the box, the subsequent step is to extract the angle of rotation for computing and predicting the cosine and sine. After the usage of the geo-volume, we can produce the dimensions of the bounding field. Using it, we then compute begin and stop for the textual content prediction field. Figure 4. shows the bounding boxes on the image.

3) OCR using Tesseract:

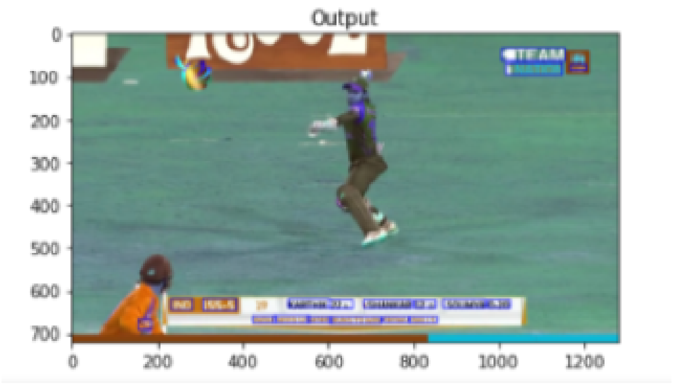
a) Text Recognition using Tesseract: After the completion of step B., The EAST model has detected and given the coordinates of all the boxes having the presence of any text. The next step is to recognize this text and unsheath the text desired for our purpose, the match score. This is done using Optical Character Recognition (OCR). For this, one of the best available OCR engines, Tesseract, is used.

Methods based on deep learning usually perform better for unstructured data. With an LSTM network (a type of RNN)-based OCR engine that is focused on line recognition but supports the legacy Tesseract OCR engine of Tesseract 3 that functions by identifying character patterns, Tesseract 4 added deep-learning-based functionality. The unstructured text is also much more accurate in this edition. Tesseract configuration is possible in a variety of ways. In this instance, we have utilized a particular Tesseract arrangement. We run through the text detected in all the bounding boxes and the text is recognized through Tesseract. The tesseract configuration options used are:

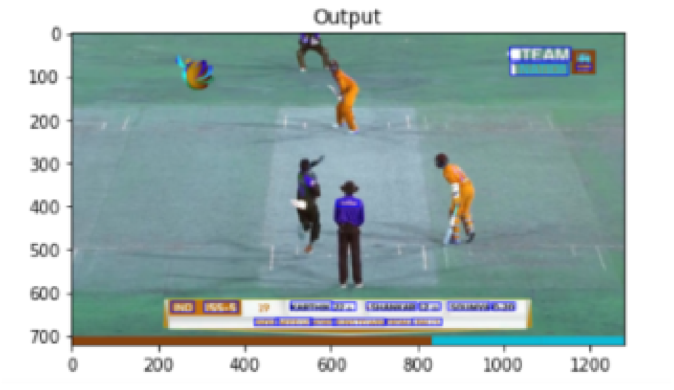
```
-l eng –oem 1 –psm 8
```

b) Getting the runs and wickets: Tesseract recognizes the different text from bounding boxes, present all over the images. However, we are only concerned with getting the score and storing them as runs and wickets. We use a regular expression for identifying the score out of all the text recognized.

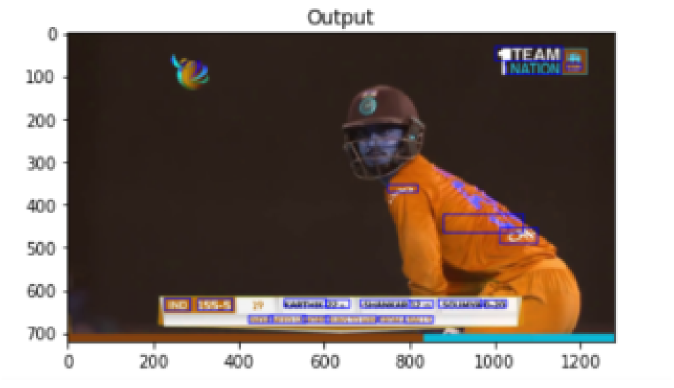
If we observe a cricket match score, it is text containing digits followed by a hyphen, followed by digits again.



[a]



[b]



[c]

Fig. 4. [a],[b] and [c], show the bounding boxes where the EAST model has detected text

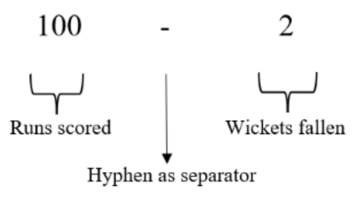


Fig. 5. Pattern of Score Display

Thus, when we find such a pattern, we recognize it as the score. The text is separated by the hyphen and the digits on either side are stored as runs and wickets respectively.

This is the format of score display used in most matches and by most broadcasters. However, it is notable to mention that matches played in Australia and broadcasted by Australian companies display the score in an inverted manner, i.e., runs at the place of wickets and vice versa. Apart from that, the separator is a '/' and not a hyphen. Suitable cases have been handled efficiently too.

4) Key Event Detection: Once, the score has been extracted from the images separately as runs and wickets, the next step is to detect the presence of key events by analyzing the score. So, a cricket highlight usually contains three main events, fours, sixes, and wickets. For computational complexity reduction, we do not extract the score from every image frame but every 50 frames. This would not adversely affect our result, and will also be reducing the load on the system.

The wickets and runs values are then compared with the values identified preceding it (i.e., the score recognized 50 frames earlier). In the existence of a consequential change in the scores of the counters of the runs and wickets, a key event would be said to be detected in the video input. For boundary occasions, key-frames are classified as follows:

$$Four\ Event : \{if \quad runs^i - runs^{i-1} \ = \ 4\} \tag{1}$$

Similarly, for a six event, the classification is as follows:

$$Six\ Event : \{if \quad runs^i - runs^{i-1} \ = \ 6\} \tag{2}$$

For a wicket, if the wicket counter increases by 1, it is to be intercepted that a wicket has fallen and is thus a key event. It is detected as:

$$Wicket\ Event : \{if \quad wick^i - wick^{i-1} \ = \ 1\} \tag{3}$$

Once, the key events are detected, we need to get the key clip corresponding to that keyframe. Thus, video clips for every key event are generated by joining the frames, ten seconds before the keyframe (that is the start point) and the key frame itself (this is the final frame of that key event). Each of these frames is appended in a chronological fashion to generate the highlights by making use of the preserved temporal information, stored earlier in section A. of this phase.

C. Phase II

Similar to how CNNs are used for image recognition, spatial characteristics from RGB video frames may be simply retrieved. However, how efficiently gathering and handling temporal data is the key difficulty in video human action recognition. The video includes crucial temporal information that, when compared to still photos, can improve the accuracy of action recognition. The challenge of video classification has evolved to include an important task on how to obtain and use temporal data. There are now two different methods for solving such a problem. One approach is to employ models in the final model which extracts the temporal information like the LSTM [11]. The LSTM employs three gates to determine whether a cell should be ignored or transmitted to the next layer. As a result, it can preserve the video's temporal information. An LSTM would, however, have a significantly larger input size than a CNN. Therefore, if these systems solely rely on the LSTM, their training speed may be poor. Adding input stream—which may be the temporal features that a CNN extracted—is an alternative strategy. In this method, optical flow is frequently used as an input. It is a collection of images that show how the object and background in a video move with respect to one another. As a result, the optical flow comprises the features in chronological order. Since the CNN has two inputs, the conventional idea is to train two independent CNNs, one of which will handle the RGB frames and the other the optical flow. The final recognition result is obtained by combining the outcomes of the two training streams. This is called the bi-directional CNN model [13]. The bi-directional CNN methodology does, however, still appear to be flawed. The RGB video frames' original temporary information is absent from the model. Despite recording the temporal aspects, the optical flow only captures pixel movement along the x- and y-axes. It is intended to combine these two strategies. For a CNN to continue processing temporal information from the optical flow, the new model retains the temporal stream. To obtain further temporal details from the RGB frames for the spatial stream, we swap out the conventional CNN for an LSTM-based model. Thus, Long-Short Term Memory (LSTM) networks are splendid for interacting with sequence data, whereas Convolutional Neural Networks (CNN) are splendid for interacting with image data. However, by combining the two, you can solve difficult computer vision issues like video classification and exploit the advantages of both. This model is known as LRCN (Long-term recurrent convolutional network) LRCN is a class of models that is deep in both ways, spatially and temporally [10]. It uses an LSTM (decoder) to produce natural language strings and a ConvNet (encoder) to encode deep spatial state vectors.

ConvolutionalNeuralNet : Time-invariant and independent at each timestep

- enables concurrent training overall input timesteps
- Independent batch processing LSTM: enables modeling of various duration sequential data.

A sequence of T inputs is used by the LRCN architecture in Fig. 6 to produce a sequence of T outputs.

Activity Recognition:

- Sequential Input [A1, A2, A3,...,At](various visual frames)
- Scalar Output [W] (detected action)

How is the LRCN model used for Activity Recognition Task?

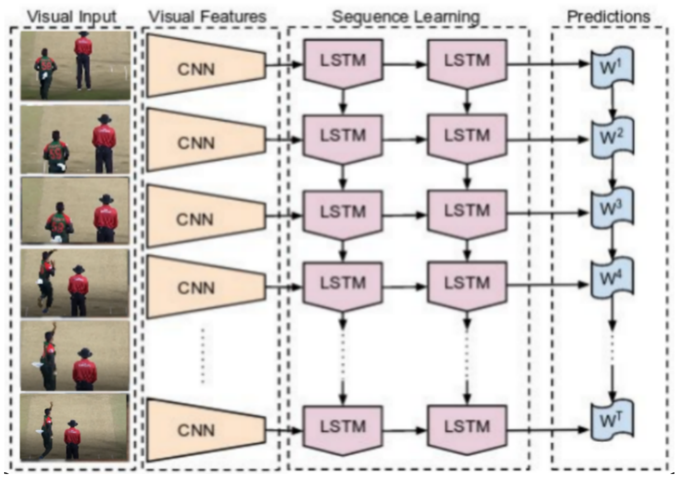


Fig. 6. LRCN architecture

- Predict video class at each time step [W1, W2, W3,.,Wt]
- Average these predictions for final classification (W)

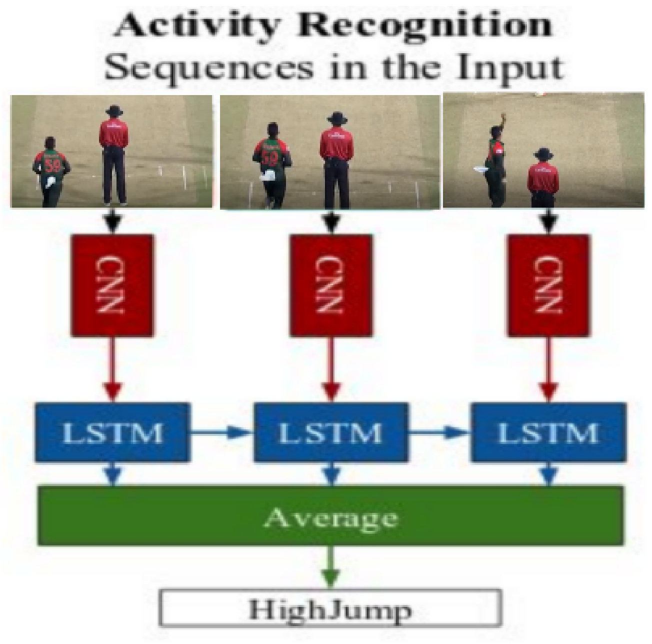


Fig. 7. LRCN flow

A weighted average of both inputs—RGB and Optical Flow—is employed to make the final prediction..

1) Implementing the LRCN Approach: In this step, we put the LRCN approach into practice by fusing the LSTM and Convolution layers to make a single model as shown in Fig.6 and 7. For tasks requiring sequential data (inputs or outputs, or both), whether linguistic, visual, or otherwise, an LRCN model combines a deep hierarchical visual feature extractor (like a CNN) with a model that can learn to recognize and synthesize temporal dynamics. To model the temporal sequence, the spatial data from the frames are sent to the LSTM layer(s) at each time step using the convolutional layers as shown in Fig.7. In this manner, a robust model is produced as the network directly learns spatiotemporal properties in an end-to-end training.

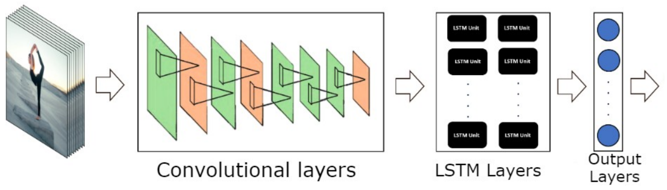


Fig. 8. LRCN overview

2) Video Input: The output video of Phase 1 serves as the input for Phase 2 i.e. this LRCN model. We take N evenly spaced frames from the entire video and provide them to the LRCN model where N is the sequence length.

3) Output: The LRCN model is used to detect the action by its Human Action Recognition feature in the input video. We design our model in a way such that Each frame is assigned a class name from 'Bowling' and 'Non-Bowling'. We obtain the timestamp where cricket bowling action ie. the class 'Bowling' is first detected in any frame of the input video as shown in Fig.9 . The frame at this timestamp is our keyframe. We merge all the frames from the keyframe to the end. Thus we obtain a shorter and more precise video output. This output then passes on to become the cricket highlight.

IV. Results and Discussion

Since our research is divided into two parts, i.e., keyframe detection and bowling action recognition, we have calculated both their results individually as well as the final result of both of them combined. The first part consists of the East Model for text detection and Tesseract for text recognition. To correctly identify the keyframes and the result for fours, sixes, and wickets, we use the formula given below:

$$Performance = \frac{DetectedEvents}{TotalEvents} \times 100 \tag{4}$$

where "DetectedEvents" refers to the number of events that were correctly detected. and "TotalEvents" are the total number of events in that video clip.

Events	Performance
Fours	92.30%
Sixes	93.33%
Wickets	93.02%

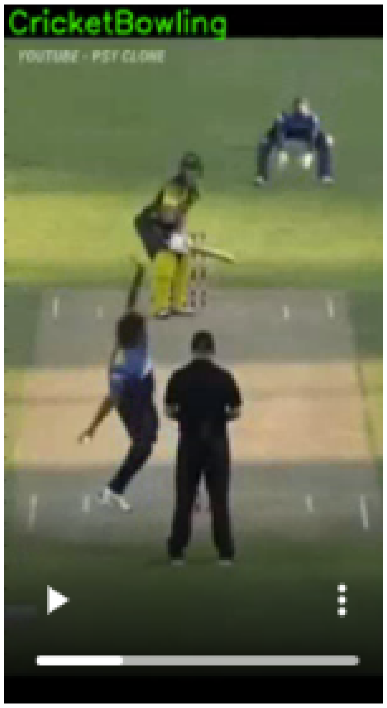


Fig. 9. Key Frame

The second part included passing the 20-25 sec clips in the LRCN model for bowling action recognition. This is done to reduce the length of the clip so that it would contain only the necessary details and not the background details such as close-ups and the audience before the ball throw. It was tested upon different datasets which yielded different results, which are as follows:

Datasets	Performance
Cricket	93.59%
Bowling	98.10%
Bowling and Non-Bowling	100.00%

The final result was calculated separately by deducing the number of clips that are formed correctly with the appropriate bowling action detection after the keyframe detection, which comes out to be 92.7 percent accurate.

V. Conclusion

The proposed framework proposes an efficient method for detecting key events in input cricket films for summarising that is based on textual features and action detection. The score captions are examined to find relevant changes in the run and wickets counters, which are then utilized to detect four, six, and wicket events. The key frame is the frame that contains the key event. To construct the summary, video skims are made against each crucial frame and then passes on to the next model to shorten the clips and generate an integrated highlight for the whole match. The suggested method is validated using a large dataset of cricket videos from multiple sports broadcasters. The experimental findings show that the suggested method is potent in terms of important event recognition for video summarising, with an average accuracy of 92.7 percent.

References

[1] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris and I. Patras, ”Video Summarization Using Deep Neural Networks: A Survey,” in Proceedings of the IEEE, vol. 109, no. 11, pp. 1838-1863, Nov. 2021, doi: 10.1109/JPROC.2021.3117472.

[2] Nasir, M., Ali Javed, Aun Irtaza, Hafiz Malik, and M. Mahmood. ”Event detection and summarization of cricket videos.” Journal of Image and Graphics 6, no. 1 (2018): 27-32.

[3] Anjaneya Basappanavar, S A Angadi, ”A Video Processing System for Detection and Classification of Cricket Events”, International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056, p-ISSN: 2395-0072. Volume: 04 Issue: 09 — Sep -2017.

[4] Vijayakumar, V. ”Event detection in cricket video based on visual and acoustic features.” Journal of Global Research in Computer Science 3, no. 8 (2012): 26-29.

[5] Bhawarthi, Deepali, and Shriniwas Gadage. ”Enriching Feature Extraction for Cricket Video Event Detection.” International Journal of Engineering Research and Technology (IJERT) 1, no. 7 (2012).

[6] Hemlata Sinha, Rohit Raj Singh,Prashant Singh, Sakshi Yadav, Sakshi Wagh, ”Automatically Extracting Excitement Clips For Cricket Match Programs”, International Journal of Mechanical Engineering, Vol. 6 No. 3 December, 2021, ISSN: 0974-5823.

[7] Namuduri, Kamesh. ”Automatic extraction of highlights from a cricket video using MPEG-7 descriptors.” In 2009 First International Communication Systems and Networks and Workshops, pp. 1-3. IEEE, 2009.

[8] P. Shukla et al., ”Automatic Cricket Highlight Generation Using Event-Driven and Excitement-Based Features,” 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW), 2018, pp. 1881-18818, doi: 10.1109/CVPRW.2018.00233.

[9] Zhou, Xinyu, Cong Yao, He Wen, Yuzhi Wang, Shuchang Zhou, Weiran He, and Jiajun Liang. ”East: an efficient and accurate scene text detector.” In Proceedings of the IEEE conference on Computer Vision and Pattern Recognition, pp. 5551-5560. 2017.

[10] J. Donahue et al., ”Long-Term Recurrent Convolutional Networks for Visual Recognition and Description,” in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39, no. 4, pp. 677-691, 1 April 2017, doi: 10.1109/TPAMI.2016.2599174.

[11] A. Ullah, J. Ahmad, K. Muhammad, M. Sajjad and S. W. Baik, ”Action Recognition in Video Sequences using Deep Bi-Directional LSTM With CNN Features,” in IEEE Access, vol. 6, pp. 1155-1166, 2018, doi: 10.1109/ACCESS.2017.2778011.

[12] Zhao, Y., Man, K.L., Smith, J. et al. Improved two-stream model for human action recognition. J Image Video Proc. 2020, 24 (2020). https://doi.org/10.1186/s13640-020-00501-x.

[13] W. Dai, Y. Chen, C. Huang, M. -k. Gao and X. Zhang, ”Two-Stream Convolution Neural Network with Video-stream for Action Recognition,” 2019 International Joint Conference on Neural Networks (IJCNN), 2019, pp. 1-8, doi: 10.1109/IJCNN.2019.8851702.