International Journal of Innovative Technology and Exploring Engineering (IJITEE)
ISSN: 2278-3075 (Online), Volume-8 Issue-11, September 2019

Automatic Annotation of Events and Highlights Generation of Cricket Match Videos

Sanchit Agarwal, Nikhil Kumar Singh, Prashant Giridhar Shambharkar

Abstract*: *In this paper, we propose a novel method for automatic annotation of events and highlights generation for cricket match videos. The videos are divided into time intervals representing one ball clip using a Convolutional Neural Network (CNN) and Optical Character Recognition (OCR). CNN detects the start frame of a ball. OCR is used to detect the end of the ball and to annotate it by recognizing the change in runs/wickets. The proposed framework is able to annotate events and generate sufficiently good highlights for four full length cricket matches.

***Keywords*: *Highlights Generation, Convolutional Neural Network, Optical Character Recognition*.**

I. INTRODUCTION

Analysis of sports videos has rapidly gained popularity in the multimedia research community. Sports videos appeal to a large number of people all over the world and are widely streamed over the internet and TV. The complexity of these videos makes their analysis difficult. In this paper, we have concentrated on the analysis of cricket match videos.

Cricket is a game of several variable factors as compared to other popular sports like football [1], tennis [2] etc. Cricket in itself is quite diverse, having three formats (Tests, ODIs and T20s), variable ground sizes and various pitch types. Also, even the shortest format of cricket i.e. T20 is played for approximately 3 hours which is greater than the duration football (approximately 90 minutes) and hockey (approximately 70 minutes) are played for. Due to this, cricket-specific methods need to be devised in order to extract information from match videos.

Reference [3] has proposed a method to extract highlights from cricket videos using MPEG-7 descriptors [4]. Their proposed approach first segments the video into shots from which key frames are selected from which low-level features are extracted. Hidden Markov Model (HMM) [5] has been used to represent the states and their transitions in the videos. Key frames are identified using a dissimilarity threshold between consecutive frames. These frames are classified as close-up, medium view, audience view and pitch view using MPEG-7 visual descriptors. The HMM model generates highlights using a probabilistic threshold based on the number of features used and number of shots present in the video.

Reference [6] have suggested highlights generation based on event driven and excitement-based features. The approach tries to extract information like scorecard, player celebration, scene understanding, replays and audio cues from the video shots and then uses them to generate highlights. Event driven features include Optical Character Recognition (OCR) [7] and play field scenarios, whereas the excitement based include audio and replays.

Reference [8] 's approach depends on low-level visual information like color histogram and histogram of oriented gradients. An unsupervised event detection framework represents video clips as uni-gram and bi-gram statistics of the detected events. A linear support vector machine [9] is trained post transcription.

Reference [10] have proposed a two-layered architecture for highlight generation. The first layer turns frame windows into feature vectors and the second layer classifies these feature vectors as highlights or non-highlights. Inception-v3 [11], Convolutional 3D [12], and SqueezeNet [13] have been tried as the base model with the top model being Fully Connected or LSTM.

In this paper, we propose a method to automatically annotate the events in cricket match videos and generate highlights by segmenting them into individual ball clips using a self-trained Convolutional Neural Network (CNN) to detect the start of the ball and Optical Character Recognition (OCR) for detecting the changes in the scorecard and signaling the end of that ball. Section 2 of the paper discusses the details of the proposed approach. Section 3 contains information about the dataset used for evaluation and the results we obtained. Conclusion and future scope are discussed in Section 4.

II. PROPOSED APPROACH

The proposed approach can be divided into the following steps:

1. Video Shot Segmentation
2. Ball Start Frame Detection
3. Ball Clip Annotation
4. Ball End Frame Detection
5. Clip Relevance Filtration and Highlights Generation

Manuscript published on 30 September 2019.
*****Correspondence Author(s)
Sanchit Agarwal*, Department of Computer Science and Engineering, Delhi Technological University, Delhi, India. Email: sanchit_bt2k15@dtu.ac.in
Nikhil Kumar Singh, Department of Computer Science and Engineering, Delhi Technological University, Delhi, India. Email: nikhil_bt2k15@dtu.ac.in
Prashant Giridhar Shambharkar, Department of Computer Science and Engineering, Delhi Technological University, Delhi, India. Email: prashant.shambharkar@dtu.ac.in

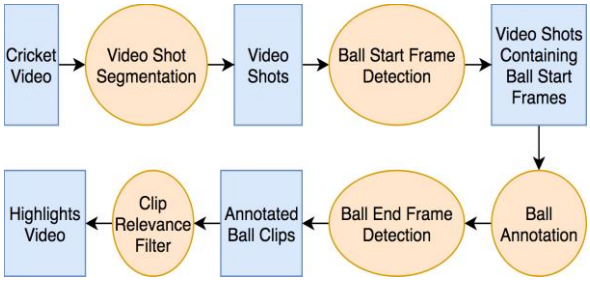

Fig. 1.Architecture of the proposed approach

A. Video Shot Segmentation

The first step of the approach is to divide the cricket match video into segments. We have used PySceneDetect for this purpose, which is an open-source command line application and a Python library for detecting scene changes in videos, and automatically splitting the video into separate clips. The ContentDetector module is used, which compares the difference in content between adjacent frames against a set threshold, which if exceeded, triggers a scene cut. This allows detection of cuts between shot scenes both containing content. Frames are compared using Hue, Saturation, Value (HSV) color space difference [14]. HSV makes the change detection invariant to changes in illumination strength as the intensity of the light is explicit and separated from the color. Since the difference between frames is used, only fast cuts are detected with this method. We have set the threshold to 10 using hit and trial method. Each video shot is represented by 3 key frames [15] - denoting its start, middle and end. This approach greatly cuts down on processing cost as now only select frames need to be analyzed by further steps instead of each and every frame from the entire video.

Fig. 2.Key frames of a video shot

B. Ball Start Frame Detection

In cricket, a ball starts when the bowler runs in to bowl the delivery. In Figure 3, it can be clearly seen that in such a frame, the bowler, pitch, batsman and non-striker are always visible. In order to classify a given frame as start-of-ball based on these observations, we have trained a Convolutional Neural Network (CNN) [16,17]. This CNN takes as input a frame from the match video and classifies it as a start-of-ball-frame or non-start frame.

Fig. 3.A ball start frame

The training data for the CNN has been generated by taking a random start-of-ball frame as the sample and dilating it. Dilation involves convoluting the image with a kernel - as the kernel is scanned over the image, the maximal pixel value overlapped by the kernel is calculated and the image pixel at the center of kernel is replaced with the maximal value. Dilation has been performed with the help of OpenCV Python Library [18].

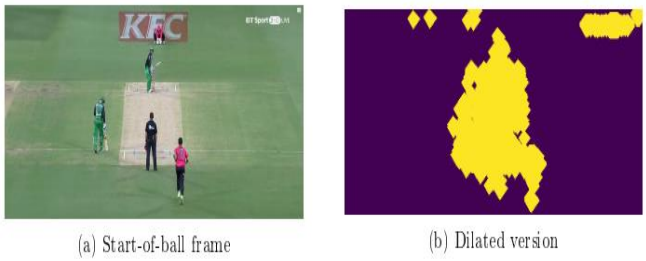

Fig. 4. A start-of-ball frame and its dilated version

After this, frames are picked from the same match video and their dilated versions are compared with the dilated version of the sample image. The comparison is carried out by calculating frame difference which is defined as the sum of the bitwise XOR of the corresponding pixels of the frames. If the difference is greater than a certain threshold (107 in our case), the frame is collected. In this way, 3395 positive frames and 2219 negative frames have been collected as training data. While training, the scorebox in the frames is not considered.

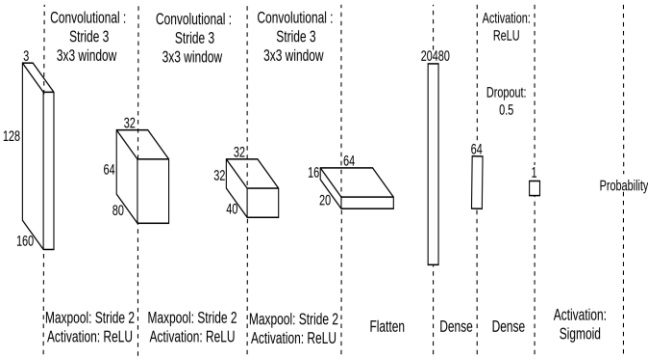

Fig. 5.Ball start CNN architecture

The CNN has been implemented using the Python library Keras's [19] Sequential module and takes as input individual frames downsized to dimensions 160x128.

The frames are kept in RGB and not converted to grayscale. Figure 5 describes the exact architecture of the CNN. All convolutional layers are 2-dimensional, have stride equal to 3 and a 3x3 convolutional window. All maxpool layers have stride equal to 2 and are followed by ReLU activation function. The first convolutional layer increases the number of channels from 3 to 32, whereas the third convolutional layer further increases the channel count to 64. After the input has passed the 3 convolutional layers, it is flattened to obtain a 20480x1 dimensional vector. This vector is then passed to a dense layer with 64 units, which in turn applies ReLU function and a dropout of 0.5 on it and forwards it to a dense layer consisting of a single neuron. Application of sigmoid activation function on this neuron's output gives the probability of the input frame being a start-of-ball frame.

(a) 0.99999118 (b) 0.03960786

Fig. 6.Probabilities of two sample frames being the start of a ball as predicted by the CNN

The key frames detected in the video shot segmentation step are fed as input to the CNN and the timecodes of all the frames with probability greater than 0.9 are saved.

C. Ball Clip Annotation

In order to annotate a ball clip, we use OCR technique over the scorebox area of the start-of-ball frames. The coordinates of the rectangular area enclosing the current over count are determined manually once which remain the same for the entire match video. Similarly, coordinates of the rectangular areas enclosing the current innings score and the individual scores of both batsmen are determined. In any given frame, the current over count, the current innings score and the batsmen's individual scores are determined by applying OCR technique over these rectangular regions. The current innings score consists of the number of runs and the number of wickets lost by the batting team. The number of runs scored on a particular ball is calculated by subtracting the runs on the previous ball from the current number of runs. Fall of a wicket is detected by calculating the difference between the number of wickets on the previous ball and the number of wickets on the current ball. The clip is marked as a milestone if any batsman's individual score crosses 50, 100, 150, 200 or 250 during it.

OCR has been implemented using the open-source Tesseract 4.0 [20] package with engine trained using Long Short Term Memory (LSTM) [21].

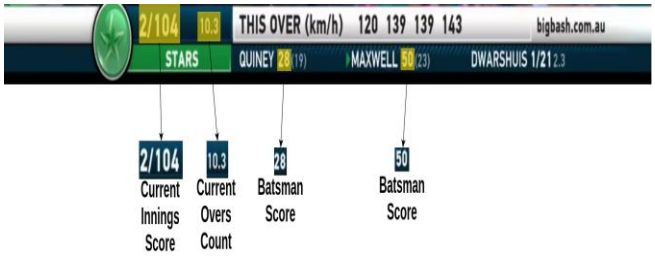

Fig. 7.Ball clip annotation using OCR over scorebox region

D. Ball End Frame Detection

To fetch the complete clip of any given ball, we need to find its start and end time. Start time has already been obtained by using ball start frame CNN classifier. In order to determine the end time of the ball clip, we use a ball end frame detector which works as follows –

a) The match video is skipped 3 seconds from the start time (as most balls last at least 3 seconds)

b) Now, frames are analyzed at the rate of 1 frame per second until –

- Current innings score changes
- Current over count changes
- Scorebox disappears

Using this ball end frame detector, the complete match video is segmented into multiple ball clips with each ball clip having its metadata which includes the innings score and the over count value before the start of that ball clip.

E. Clip Relevance Filtration and Highlights Generation

Since the complete match video has been annotated, we can generate a highlights clip of the cricket match. The criteria for including a ball clip in the highlights is as follows –

a) Four or more runs are scored on that ball
b) A wicket falls on that ball
c) Ball clip is marked as milestone
d) Ball clip is the last of the innings

All such ball clips are selected and stitched together to generate the final highlights clip using the Python library MoviePy [22].

III. EVALUATION AND RESULTS

A. Dataset Description

Our dataset comprises of 14 full length T20 match broadcast videos, consisting of both international and league games. Out of these, 10 matches have been used to generate training data for the CNN model. The remaining 4 match videos have been used to generate the highlights to be evaluated. The videos are of the resolution 1280x720.

B. Results and Discussion

The usefulness of a highlights video is largely a personal decision. Some people may prefer all the boundaries and wickets in the video, while some may want to see only the defining moments of the match.

Published By:
Blue Eyes Intelligence Engineering
and Sciences Publication (BEIESP)

We have used the Intersection over Union (IoU) metric [23] to evaluate our highlights with respect to the highlights provided by the official broadcaster. IoU is also known as the Jaccard index. It measures similarity between finite sample sets (broadcaster's highlights and generated highlights in our case). An IoU score of 1 denotes that the sets being compared are the same, whereas a score of 0 implies that the two sets are mutually exclusive. Reference [6] has also evaluated their approach using the IoU metric. IoU has been calculated as the ratio of the difference of number of events in broadcaster's highlights and number of events in broadcaster's highlights but not in generated highlights, and the sum of number of events in broadcaster's highlights and number of events in generated highlights but not in broadcaster's highlights.

The mean IoU score for the 8 innings of the 4 matches has been calculated and is equal to 0.829, while the maximum and minimum IoU scores were 0.944 and 0.633 respectively. In comparison, [6] has achieved a mean IoU score 0.731, max IoU score 0.86 and min IoU score 0.454.

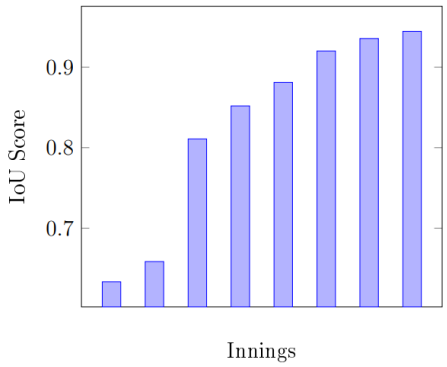

Fig. 8.IoU scores for the 8 innings

IV. CONCLUSION AND FUTURE SCOPE

We have proposed a method for automatic annotation of events and highlights generation in cricket match videos. It can be seen that a combination of a well-trained CNN and OCR can produce good results in this scenario. Further work in this direction can be to make the approach immune to the broadcaster's mistakes such as disappearance of the scorebox during a live match, and to generate player-specific highlights.

REFERENCES

1. J. Assfalg, M. Bertini, A. Del Bimbo, W. Nunziati, and P. Pala, "Detection and recognition of football highlights using hmm," in, 2002. 9th International Conference on, vol. 3. IEEE, 2002, pp. 1059–1062.
2. Y.-P. Huang, C.-L. Chiou, and F. E. Sandnes, "An intelligent strategy for the automatic detection of highlights in tennis video recordings," *Expert Systems with Applications*, vol. 36, no. 6, pp. 9907–9918, 2009.
3. K. Namuduri, "Automatic extraction of highlights from a cricket video using mpeg-7 descriptors," in *Communication Systems and Networks and Workshops, 2009. COMSNETS 2009. First International*. IEEE, 2009, pp. 1–3.
4. B. S. Manjunath, P. Salembier, and T. Sikora, "Introduction to MPEG-7: multimedia content description interface". John Wiley & Sons, 2002, vol. 1.
5. J. S. Boreczky and L. D. Wilcox, "A hidden markov model framework for video segmentation using audio and image features," in *ICASSP*, vol. 98, 1998, pp. 3741–3744.
6. P. Shukla, H. Sadana, A. Bansal, D. Verma, C. Elmadjian, B. Raman, and M. Turk, "Automatic cricket highlight generation using event-driven and excitement-based features," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 1800–1808.
7. S. Mori, H. Nishida, and H. Yamada, *Optical character recognition*. John Wiley & Sons, Inc., 1999.
8. H. Tang, V. Kwatra, M. E. Sargin, and U. Gargi, "Detecting highlights in sports videos: Cricket as a test case," in *Multimedia and Expo (ICME), 2011 IEEE International Conference on*. IEEE, 2011, pp. 1–6.
9. M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.
10. J.-T. T. Hsieh, C. E. Li, W. Liu, and K.-H. Zeng, "Spotlight: A smart video highlight generator," http://cs231n.stanford.edu/reports/2017/pdfs/708.pdf/, 2017. Last accessed on 27th July 2019.
11. C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the inception architecture for computer vision," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2818–2826.
12. S. Ji, W. Xu, M. Yang, and K. Yu, "3d convolutional neural networks for human action recognition," *IEEE transactions on pattern analysis and machine intelligence*, vol. 35, no. 1, pp. 221–231, 2013.
13. F. N. Iandola, S. Han, M. W. Moskewicz, K. Ashraf, W. J. Dally, and K. Keutzer, "Squeezenet: Alexnetlevel accuracy with 50x fewer parameters and< 0.5 mb model size," *arXiv preprint arXiv:1602.07360*, 2016.
14. S. Sural, G. Qian, and S. Pramanik, "Segmentation and histogram generation using the hsv color space for image retrieval," in *Image Processing. 2002. Proceedings. 2002 International Conference on*, vol. 2. IEEE, 2002, pp. II–II.
15. W. Wolf, "Key frame selection by motion analysis," in *Acoustics, Speech, and Signal Processing, 1996. ICASSP-96. Conference Proceedings., 1996 IEEE International Conference on*, vol. 2. IEEE, 1996, pp. 1228–1231.
16. Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
17. A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in neural information processing systems*, 2012, pp. 1097–1105.
18. G. Bradski, "The OpenCV Library," *Dr. Dobb's Journal of Software Tools*, 2000.
19. F. Chollet et al., "Keras," https://keras.io, 2015. Last accessed on 27th July 2019.
20. R. Smith, "An overview of the tesseract ocr engine," in *Document Analysis and Recognition, 2007.ICDAR 2007. Ninth International Conference on*, vol. 2. IEEE, 2007, pp. 629–633.
21. S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural computation*, vol. 9, no. 8, pp.1735–1780, 1997.
22. Zulko, "Moviepy," https://zulko.github.io/moviepy/, 2017. Last accessed on 27th July 2019.
23. K. Soomro and A. R. Zamir, "Action recognition in realistic sports videos," in *Computer vision in sports*. Springer, 2014, pp. 181–208.

AUTHORS PROFILE

Sanchit Agarwal is a senior year computer engineering student at Delhi Technological University. His fields of interest are computer vision in sports, natural language processing and algorithmic trading and finance.

Nikhil Kumar Singh is a senior year computer engineering student at Delhi Technological University. His fields of interest are deep learning, computer vision and Python programming.

Prashant Giridhar Shambharkar is an Assistant Professor at the department of Computer Science and Engineering, Delhi Technological University. His fields of interest are Data Mining, Real Time Systems, Mobile health monitoring. He has completed his B.E. From Amravati University and M.Tech From RGPV, Bhopal.

Retrieval Number: J11180881019/19©BEIESP
DOI: 10.35940/ijitee.J1118.0981119
Journal Website: www.ijitee.org

Published By:
Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP)