

# Automation of Cricket Scoreboard by Recognizing Umpire Gestures



Vaishnavi K. Nair, Raakhi Rachel Jose, Parvathy B. Anil, Minnu Tom, Lekshmy P.L.

**Abstract:** *Imagine how tiresome it is for the scorers to update the scoreboard after each ball delivery during a cricket match. They need to be alert during any point in the match, watch every single ball, record ball by ball events, modify the score and coordinate with the umpire the entire time. A system that can update the scoreboard automatically after every ball will lessen their effort by half; the time taken for the updation and the chances of errors will also be reduced. A novel method for umpire pose detection for updating the cricket scoreboard during real-time cricket matches is suggested in this work. The proposed system identifies the events happening in the pitch by recognizing the gestures of the umpire and then updates the scoreboard accordingly. The concept of transfer learning is used to accelerate the training of neural network for feature extraction. The Inception V3 network pretrained on the visual database ImageNet is culled as the primary prospect for feature extraction. Instead of initializing the model with random weights, initializing it with the pretrained weights reduces the training time and hence is more efficient. The proposed system is a combination of two SVM classifiers. The leadoff classifier tells apart the images that contain an umpire from the non-umpire images. These 'umpire' images are then carried forward to the event detection classifier while the 'non-umpire' images are repudiated. The second classifier is able to identify four gestures – 'Six', 'Wide', 'No ball' and 'Out' from the images, following which the scoreboard is updated. In addition to these four classes, one more label is defined to group those umpire frames within which the umpire does not show any signal, namely the 'No Action' class. The cricket video given as input is first split into number of shots and each frame is considered as a test image for the combined classifier system. A majority voter is used to confirm the final classification result which decreases the chances of misclassifications. The preliminary results suggest that the intended system is efficacious for the purpose of automating the updation of scoreboard during real time cricket matches.*

**Keywords** — *Image Classification, InceptionV3, SVM classifier, Transfer Learning, Umpire Gesture Recognition.*

Revised Manuscript Received on May 12, 2020.

\* Correspondence Author

**Vaishnavi K. Nair\***, Department of Computer Science & Engr., L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala. Email: nair\_vaishnavi@yahoo.com

**Raakhi Rachel Jose**, Department of Computer Science & Engr., L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala. Email: raakhirachel@gmail.com

**Parvathy B. Anil**, Department of Computer Science & Engr., L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala. Email: parvathyani32@gmail.com

**Minnu Tom**, Department of Computer Science & Engr., L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala. Email: minnutom06@gmail.com

**Lekshmy P.L.**, Department of Computer Science & Engr., L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala. Email: lekshmy\_vinod@gmail.com

© The Authors. Published by Blue Eyes Intelligence Engineering and Sciences Publication (BEIESP). This is an [open access](http://creativecommons.org/licenses/by-nc-nd/4.0/) article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>)

## I. INTRODUCTION

The history of Cricket started in the 16th century in England. In the five centuries since, the sport has advanced to different levels. From carving notches on sticks, cricket scoring has now progressed to PC softwares and mobile applications. The scoring system in cricket consists of two factors - the number of runs earned together with the number of wickets that is broken. In professional cricket matches, two scorers are designated, most often one allotted by each team, to chronicle all runs scored, all wickets taken and, where appropriate the number of overs bowled. However, it is the umpire that calls the runs and wickets lost before signaling to the scorer who then records this score. The updated score is then displayed on the giant screen visible to the players and crowd. Notwithstanding the fact that their roles have been defined as merely the recording of runs, wickets, and overs, in reality it is much more complicated as they are responsible for other game statistics as well.

There are two methods used in cricket scoring: manual and computerized. In the manual or paper based scoring method, the scorer makes use of a preprinted scorecard, informally known as the scorebook to enter details of each ball in the match. Cricket scoring softwares were developed to eliminate the complications associated with the traditional method. However, we cannot say that even this process is fully computerized. The scorer needs to press buttons to record ball by ball events. Doing so after each and every delivery can be tedious and repetitious. If this scoring process is to be automated, the effort of the scorekeepers will be significantly reduced as they do not need to tap the screen after each delivery and can focus solely on the other game statistics. Automating the process can reduce the time taken to execute this dreary task. It also reduces the likelihood of errors in updating the scoreboard.

The initial systems proposed for this purpose worked by tracking the motion of the ball in the pitch. This required using multiple cameras and complex algorithms. Hence the chances for deploying this system in real time matches was very low. The concept behind the second wave of intended systems was entirely diverse. Since scorers were directed by the hand signals of the umpires, Machine Learning enthusiasts came up with the idea of recognizing the umpire gesture and then updating the scoreboard accordingly and thus the system fell into the domain of image classification. A method for umpire pose detection for automating the scoreboard based on transfer learning is proposed in this work. Transfer learning is a booming arena of Artificial Intelligence that is concerned with the potential of a machine learning algorithm to be reused as the starting point for a model on a second task which has significantly less training data when compared to previously learned tasks.

# Automation of Cricket Scoreboard by Recognizing Umpire Gestures

The Inception V3 image recognition neural network is used as a feature extractor. The Inception V3 network represent a quantum leap in image recognition as it is capable of automated feature extraction without any human supervision. For example, given many pictures of umpires and players it learns distinctive features for each class by itself. The model, previously trained on the ImageNet database is computationally more efficient. The algorithm scans the data to search for features that correlate and combine them to enable faster learning. These features are then used to train a linear support vector machine for identifying various umpire gestures. The system at present is able to recognize four actions – Six, No Ball, Out and Wide but further down the road, it can be retrained to identify other gestures as well and can then be deployed in real time matches.

The system intended in this work consists of two stages. The first phase is a classifier which groups the images either into the umpire or non-umpire category. The presence of this classifier is what enhances the efficiency of our system when compared to the existing ones. It assures that only the gestures signaled by the umpire and not the players are taken into consideration for updating the scoreboard. The ensuing phase is gesture classification and updating the scoreboard in accordance with the predicted label.

The system takes a cricket video as input, identify the frames in which an umpire is present and then checks if he/she is signaling any relevant events; if so the scoreboard is updated. The keyword here is relevant. Nothing happens if the gesture is signaled by a player or if the umpire does not signal any genuine gesture.

## II. LITERATURE REVIEW

In the domain of automatic score update, several works such as [1], [2] and [3] have been reported. L. Bhansali and M. Narvekar has explored the scope of subtraction technique, gradient technique and region of interest technique to recognize umpire gestures [1].

A. Shahjalal et. al makes use of Haar Cascade Algorithm to detect human wrists from the video stream and then classifies the gesture using logistic regression based on the pixel intensity values. Though this system has a pretty good accuracy for static gestures like out, no, six, wide etc., the efficiency was found to be poor for dynamic gestures like four. In addition, the system fails when the frame comprises of multiple subjects [2].

In the method D.T. John et. al propose [3], a pre-trained model from OpenCV called Caffe is used to extract the skeletal joints from the images given as input. A deep neural network is subsequently trained on the skeleton images constructed from these skeletal joints which is supposedly unique for every gesture. Since the umpires have different physiques the length between two skeletal points can vary. This results in the corresponding attribute having a greater value range. Consequently, the system must be trained on colossal amounts of data to make certain that the gesture is not misclassified. Extracting skeletal points is difficult if the figure in the image is not properly oriented or if any limbs of the figure is foreshortened or occluded. Another limitation of this work is that only images can be given as input. There are also chances of the system misclassifying the actions out and

bye. During pose estimation using Caffe Model, the joints in the fingertips are not identified. With the only skeletal joint identified in the hand being the wrist, the skeletal images obtained for Out and Bye will be the same. Therefore, there is a 50% chance of the image being wrongly classified.

Hari et al. have proposed a method based on intensity projection profile of umpire gestures for detecting events such as Six, Out and Wide [4]. The entire input video is first segmented into individual scenes, and then the presence of the umpire in each scene is checked using threshold based color segmentation technique, since umpires wear uniforms of distinct color. The users are required to input the color of the umpire's uniform before using this system.

M.H. Kolekar and K. Palaniappan has suggested a framework for event detection and classification from cricket videos by extracting audio and other low level features [5]. West et.al have presented a technique of using accelerometers for video annotation [6]. Harikrishna et.al has proposed a scheme for summarizing the visual contents of a cricket video [7]. A. Ravi et.al in a prior study have explored the use of Inception V3 network in event detection from cricket videos for the purpose of match highlights generation [8]. This work also proposes a benchmark dataset called the SNOW dataset for umpire gesture recognition from which images have been selected to constitute the dataset which was used to train and test the classifier models proposed in this work.

## III. METHODOLOGY

The proposed system is constructed in two phases. The first phase involves the designing of the classifier models. The first classifier sorts the images into 'umpire' and 'non-umpire' classes while the second classifier identifies the event gestured by the umpire. The next phase involves updating the scoreboard based on the recognized gesture.

### Phase 1 – Classifier Design

For the purpose of cricket scoreboard updation two classifiers are designed: Classifier 1 and Classifier 2. The Classifier 1 assigns either an 'umpire' or a 'non umpire' label to the images. It has been trained on 450 umpire and 450 non umpire images. The succeeding, Classifier 2 is devised for the event detection task and is trained on 500 umpire images: 100 images, each belonging to five classes of events - Six, No Ball, Out, Wide and No Action. The steps involved in designing the classifiers are common to both Classifiers 1 and 2. Intensity normalization and resizing of the images to 299 x 299 pixel size are performed as preprocessing steps before feeding them into the CNN for feature extraction. The methods `cv.normalize()` and `cv.resize()` inbuilt in python's OpenCV library are respectively used for normalization and resizing. The features are then extracted from the processed images; from the last fully connected layer of the Inception V3 network. These features, also known as bottleneck features, represent the activation map from the last convolutional layer and are extracted for both classifiers.

This feature vector is saved as a floating point NumPy array of length 2048 for further processing.

The SVM algorithm plots each feature vector procured from the training images as data points in the 2048 dimensional space with the value of each vector being the value of a specific coordinate. Then, classification is carried out by finding the hyper-plane which breaks up the two classes very well. When a test image is given, the feature vectors are drawn out in a similar fashion and depending on which side of the hyperplane, the new point is plotted, the label of the image is forecasted.

SVM is inherently a binary classifier. However, the gesture recognition part of our work is essentially a multiclass classification problem. The algorithm, in this case makes use of a one against one approach that takes splits the dataset into one dataset for each class versus every other class.

The Classifier 1 makes use of a linear kernel to sort the images. The regularization parameter was inferred as 10 using Grid Search. Classifier 2 designates one of the five labels- 'Out', 'Wide', 'No ball', 'Six' and 'No Action' to the gesture in the image. The regularization parameter for this classifier is 10 as well but a polynomial kernel was used.

The Python package scikit-learn was used to train the SVM, and NumPy was used to process and store the data during the experiments. Python's pickle library was used to store and retrieve the trained classification models. The classes for the new data instances were predicted using the finalized classification model in scikit learn using the predict () function. The two classifiers are combined into one system for score updation.

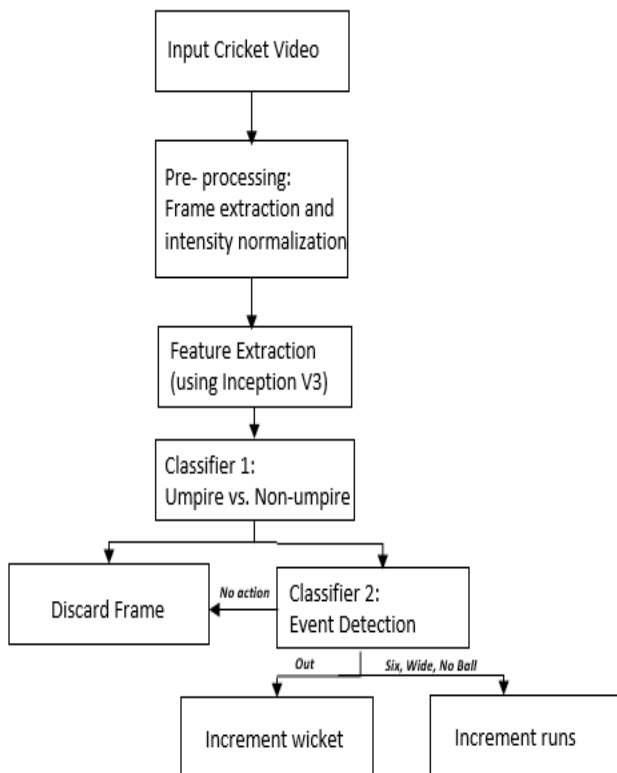


Figure 1: Overall design of proposed system

### A.Phase 2 – Updating the Scoreboard

We are proposing a system that automatically updates the

scoreboard by detecting the gesture that is signaled by the umpire after each critical event. The saved classification models are combined into one system to realize this. Fig. 1 illustrates the overall design of the suggested combined system. The main steps involved in the proposed framework are frame extraction, preprocessing, feature extraction, umpire vs. non umpire classification, event detection and finally, score updation. The first step is the extraction of frames from the input video. The frame rate of the system is 25 fps, i.e. a video of duration one second is converted into 25 frames. Each frame is then subjected as a test image for the pipelined classifier system. Intensity normalization is performed to ensure comparability across images. The Inception V3 model was originally trained for input of size 299x 299. Hence resizing is done before the feature extraction stage. In the next step, the features are first tested on Classifier 1 and if grouped into the 'umpire' category, the images are directed to Classifier 2 to detect the pose of the umpire and hence identify the event. If the frame was classified as an image belonging to one of the three classes of Six, No Ball or Wide, then the run count is incremented; if the predicted label is out, then the wicket count is incremented. All the other actions are sorted into the label 'no action'. If the classified frame belongs to the 'no action' category, then it is discarded as it is not relevant for the purpose of event detection.

Once all the frames in an input video are processed, the scoreboard is updated based on a majority vote. This increases the reliability of the proposed system. Every frame in the video, that has the presence of an umpire figure will be classified into one of the five classes. Depending on the label predicted for each frame, the global counter variable associated with that particular label is increased. Ideally, most of the frames in a video must be grouped into the category corresponding to the event happening in the video. Once, all the frames in the video is processed, the label corresponding to the counter variable with the highest value is predicted as the output class.

The working of the proposed system can be explained with the help of Fig. 2. A video containing an event 'Out' is fed into the system. The video is segmented into multiple frames and each frame after preprocessing is classified into the 'umpire' or 'non-umpire' category. The first two frames which shows the bowler and the fielders are identified as 'non-umpire' images and are thus discarded. The remaining images deemed as umpire images are given as input to the second classifier which predicts the action the umpire is signaling. The SVM predicts a label for each of these frames and finally the action in the video is identified as out as the frames are sorted more in number into the Out category than the others (3 out of 6 frames are classified as 'Out').

The scoreboard is maintained using a spreadsheet. When the script is run, the current values of 'runs' and 'wickets' are retrieved and modified depending on the predicted output class. The altered values are then written back to the excel sheet. The output of the script is a dialog box that projects the intended label and the current



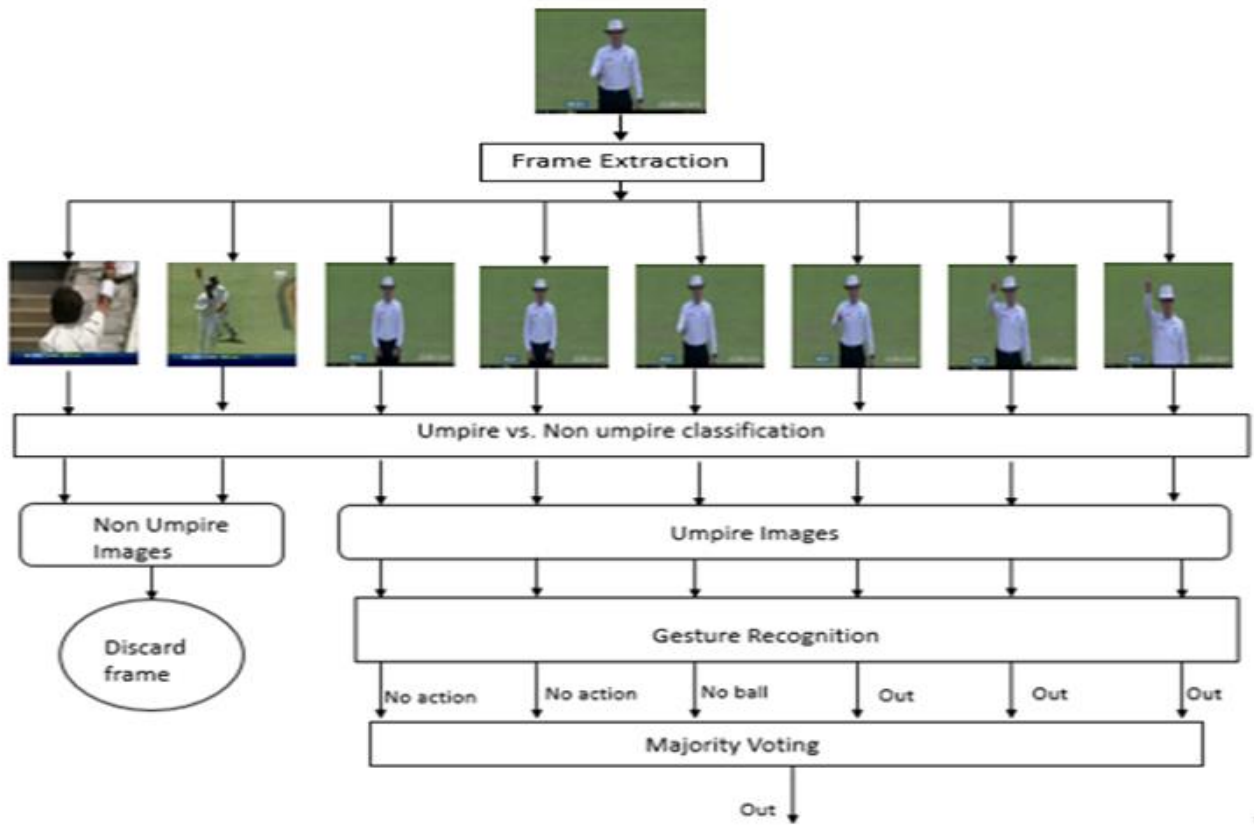


Fig. 2: Working of the proposed System

score. The dialog box has two input buttons – ‘reset’ and ‘continue’. Clicking on the reset button resets the values of ‘runs’ and ‘wickets’ to 0 in the spreadsheet. The ‘continue’ option saves the modified values to the spreadsheet, and the script runs again. Fig. 3 shows the score board popup before and after a video containing an out event is given as input.

IV. RESULTS AND DISCUSSION

A. Classifier Design Results

The classifiers are designed based on features extracted from the last fully connected layer of Inception V3 network. Classifier 1 and Classifier 2 are trained on 50% of the dataset and the remaining 50% of the dataset is used for testing the classification performance. These classifiers are validated based on a 10-fold cross validation on the training data. The test accuracy is calculated based on the remaining 50% of the unseen data.

Classifier 1 is trained for umpire detection. This classifier groups the umpire images together while pictures of the players and crowd is sorted into ‘non umpire ‘category. Fig. 4 illustrates some examples of correct classifications while Fig. 5 shows some misclassifications of the classifier. Although the precise grounds for wrongly classifying non umpire images into the umpire class is not known, examining certain characteristics in the misclassified images can provide some insight.

In Fig.5, the picture on the left shows cricketer Virat Kohli whose action resembles that of an umpire signaling ‘Out’ whereas the image on the right has Lasith Malinga donning a

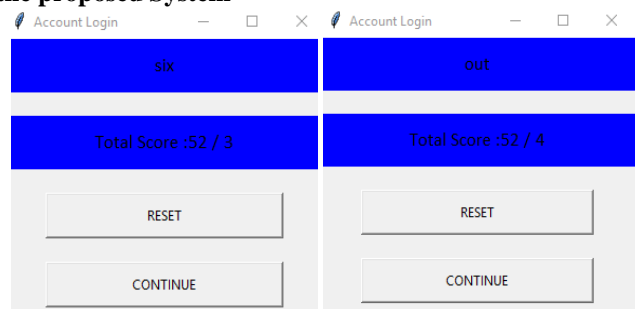


Fig. 3: The scoreboard before and after an ‘out’ video is given as input

panama hat that is similar to the ones worn by umpires. Therefore, it can be deduced that the presence of the hat and specific gestures are criterion for sorting the images into the umpire class. The confusion matrix for the first classifier is given in Table I. 400 umpire and 400 non umpire images were tested on the model and the accuracy was found to be 96.125%.

Classifier 2 is trained on 500 images -100 images each belonging to the five output labels: out, wide, no ball, six and no action for recognizing the umpire gesture. 390 images were taken from the SNOW dataset; the rest were obtained from Google Images and YouTube Videos. This classifier was tested on another 500 novel images that are not part of the training data. Accuracy is found to be 93.4 %. The confusion matrix for the same is given in Table II. It has also been observed that more





Fig. 4. : ‘Non umpire’ and ‘Umpire’ images correctly classified by Classifier 1



Fig 5: Images misclassified by Classifier 1

number of umpire frames are classified as belonging to the no action class when compared to the number of frames correctly classified as a No Ball. There is huge diverseness in the images used to train the classifiers. There are photographs taken at different lighting, camera angles and orientations.

### B. Event Detection from the Video

Classifiers 1 and 2 are combined such that both are trained using the same feature extraction method. The combined system is evaluated for its effectiveness in automatic score update. The proposed cricket scoreboard automation system is assessed for its ability to identify the action from novel test videos and update the scoreboard accordingly.

The videos used for this purpose are based on a collection of cricket videos available on YouTube. Five short video segments each, containing events belonging to the output labels except ‘no action’ are synthesized from the collection of videos. Videos containing multiple events were trimmed so as to reduce the processing strain of the GPU. The system performance is measured based on the metrics of Sensitivity (also called the true positive rate) and Accuracy. Sensitivity is the ratio exists between the number of rightly identified events and the actual number of events belonging to a specific class while accuracy is the ratio between correctly identified events and all events. They are defined as:

$$Sensitivity = TP \div (TP + FN) \quad (1)$$

$$Accuracy = (TP + TN) \div (TP + FP + TN + FN) \quad (2)$$

Table I: Confusion Matrix of Umpire vs. Non Umpire classification

	Umpire	Non Umpire
Umpire	378 (TP)	22 (FN)
Non Umpire	9 (FN)	391 (TN)

Table II: Confusion Matrix of the Event Detection Classifier

	Out	Wide	No Ball	Six	No Action
Out(100)	97	-	-	1	2
Wide(100)	-	93	4	-	3
No Ball (100)	-	-	88	4	8
Six (100)	1	-	-	95	4
No action (100)	2	-	1	3	94

Table III : Results of Event Detection from video

Event Category	Actual no. of events	TP	FP	FN
Six	5	4	0	1
No Ball	5	5	0	0
Out	5	5	0	0
Wide	5	5	0	0
<b>TOTAL</b>	<b>20</b>	<b>19</b>	<b>0</b>	<b>1</b>

where a true positive (TP) is reported if the system identifies an event correctly as the actual event. A false positive (FP) is reported if the system identifies an event as belonging to one of the four categories when the actual event is not present in the video. If the system fails to detect an event, it is recorded as a false negative (FN). In the ideal case, a good performing system is required to have a high sensitivity and precision. Table III presents the results of the combined system tested on the twenty synthesized cricket videos. The system achieves very high sensitivity and accuracy of 0.95.

The time complexity of the system should also be taken into consideration while assessing its performance. The proposed system on an average requires about 3 minutes to update the scoreboard after the synthesized videos are given as input. The experiments were performed on a machine with Intel Core i3 processor and 6GB RAM space.

## C. Future Scope

The model is trained to detect only four umpire gestures – Six, No Ball, Wide and Out. Other events when given as input is sorted into the ‘no action’ category. In the future, the model can be trained to recognize the other gestures as well.

Identifying dynamic gestures like four can be made possible by providing extensive training to the model. Images showing the various positions of the umpire’s hands while gesturing the action should be included in the dataset for the corresponding label. Ample images must be included so that the model can effectively identify the action as ‘four’ irrespective of the position of the hand. At the same time, more images belonging to the other classes must also be added to equalize the dataset. If this titanic task is to be achieved, the system can be employed in real time matches.

Also, the system at present does not have a roll back mechanism in case a video has been misclassified. The only way to alter the score is to edit the spreadsheet used to record the run and wicket count. A proper rollback mechanism can be implemented before the system is deployed.

## V. CONCLUSION

We proposed as system that can update the cricket scoreboard during real time matches without any human interaction. The system works by identifying the signal gestured by the match umpire after critical events. The only limitation of this system will be that the scorers should still pay attention to the runs taken by the batsmen by running between the wickets, for the umpire does not signal 1s and 2s.

The preliminary classification results indicated that the features extracted from the Inception V3 network is an excellent alternative to skeletal image and pixel value variations used in the previously proposed works. The system was put to test on cricket videos and was found to have an accuracy of 95% which is much higher when compared to the previously proposed systems. Even then, improvements can be made in the area of classifier design to minimize false negatives. Other classification algorithms apart from SVM can be employed for the same.

Unlike the existing ones, this system does not have any constraints that compromises its performance and efficiency. The model can be easily retrained to recognize the other umpire gestures and can also be adapted to other sports. Further down the lane, this system can be used in local matches and amateur tournaments as well.

The proposed system implemented with Intel Core-i3 processor and 6 GB RAM requires approximately 30 seconds to process 25 frames. i3 processors are usually not preferred for the high level processing required in AI/ML. Using an i5 or above considerably reduces the time taken for the processing of the input videos and altogether provides a better experience.

## REFERENCES

1. L. Bhansali and M. Narvekar, “Gesture recognition to make umpire decisios,” International Journal of Computer Applications, Volume. 148, No. 14, 2016.
2. A. Shahjalal, Z. Ahmad, R. Rayan, L. Alam, “An Approach to Automate the Scorecard in Cricket with Computer Vision and Machine Learning,” 3rd International Conference on Electrical Information and Communication Technology (EICT), 7-9 December 2017, Khulna, Bangladesh.

3. D. T. John, K.S. Kumar, V. T. Nair, P. Visakh, B. R. Poorna, “Cricket Scoreboard Automation using Umpire Gestures,” International Journal of Research in Engineering, Science and Management, Volume-2, Issue-7, July-2019.
4. Hari, R., & Wilscy, M. (2014, December), “Event detection in cricket videos using intensity projection profile of Umpire gestures,” in India Conference (INDICON), 2014 Annual IEEE (pp. 1-6). IEEE.
5. M. H. Kolekar and K Palaniappan, “Semantic Event Detection and Classification in Cricket video sequence,” Sixth Indian Conference on Computer Vision, Graphics and Image Processing, 2008, Page 382-389
6. West, G. A., Venkatesh, S., & Chambers, G. S., (2004, August), “Automatic labeling of sports video using umpire gesture recognition,” in Joint IAPR International Workshops on Statistical Techniques in Pattern Recognition (SPR) and Structural and Syntactic Pattern Recognition (SSPR) (pp. 859867). Springer, Berlin, Heidelberg.
7. Harikrishna, N., Sathesh S., Sriram, S. D., & Easwarakumar K. S., “Temporal classification of events in cricket videos” in Communications (NCC), 2011 National Conference on (pp. 1-5). IEEE.
8. A. Ravi, H. Venugopal, S. Paul, H. R. Tizhoosh, “A Dataset and Preliminary Results for Umpire Pose Detection Using SVM Classification of Deep Features,” IEEE Symposium Series on Computational Intelligence SSCI 2018.
9. Szedgy, C., Vanhoucke, V., Ioffe, S., Shlens, J., & Wojna, Z. (2016). Rethinking the inception architecture for computer vision. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (pp. 2818-2826).
10. Pan, Sinno Jialin, and Qiang Yang. "A survey on transfer learning." IEEE Transactions on knowledge and data engineering 22.10 (2010): 1345-1359.

## AUTHORS PROFILE



**Vaishnavi K. Nair** is a senior year undergraduate student of Computer Science & Engineering at L.B.S. Institute of Technology for Women, Thiruvananthapuram, Kerala affiliated to the APJ Abdul Kalam Technological University. She is presently the student editor of the technical magazine ‘Avenir’ published by the institution. She is passionate about machine learning and cyber security.



**Raakhi Rachel Jose** is an undergraduate student in the Computer Science program at LBS Institute of Technology for Women and will be graduating in 2020 with B. Tech degree. She holds the position of the college’s CS/IT Association secretary and has coordinated several workshops and other technical events. She is actively involved in the IEEE and is keen in exploring coding, cryptography and

information protection.



**Parvathy B. Anil** is currently pursuing her final year of B. Tech in L.B.S. Institute of Technology for Women. An active member of the IEEE, she is fascinated by natural language processing.



**Minnu Tom** is a final year engineering student in LBS Institute of Technology for Women. She will be completing her bachelor’s degree in 2020. she is highly focused on a competent academic career and has strong interests in the field of computer vision.





**Lekshmy P.L.** is an assistant professor in the Department of Computer Science & Engineering and has been on the faculty at L.B.S. Institute of Technology for Women since 2008. She received her Bachelor's degree in Information Technology from M.S. University 2004 and her Masters from Karunya Deemed University, in 2006. Her research interests include Privacy Preserving datamining

and Big Data Analytics.