IoT and Computer Vision for Analysing the players on the IISERB Badminton court

Group Name : PAI

1st Alli khadga Joyth
khadga19@iiserb.ac.in
19024

2nd Anushka Sanjay Shelke
anushka19@iiserb.ac.in
19290

3rd Yasmin Manavar Tadavi
yasmin19@iiserb.ac.in
19321

***Abstract*—Computer Vision is the branch of the science of computers and software systems that can recognize and understand images and scenes. Computer Vision consists of various aspects such as image recognition, object detection, and many more. In this paper, we are detecting players on the court with the help of IoT and Computer Vision, and sports classification from images. We are using the YOLOv3 pre-trained model for player detection with OpenCV as its backend. The sport Classification model is a fully custom-built CNN model able to classify different sports from the images of the court. Our player detection model takes images, recorded videos, and live webcam footage of the sports match form, which can successfully detect players on the court and the audience watching. The model we built gave us satisfactory results yielding an accuracy greater than 95% for our sport classification. With these two models perfected, the sports analysis industry can be hugely beneficial. The sport classification model can automatically detect the sport played, and the player detection model can extract all the relevant information regarding the game in real-time [1].**

***Index Terms*—CNN, DNN, OpenCV, YOLOv3, Tensorflow, Object Detection, Sport Classification**

I. Introduction

Object detection for Computer Vision is one of the fastest evolving branches of Artificial Intelligence in the past 20 years [2]. It deals with detecting object instances in a visual frame with objects belonging to various classes(such as animals, trees, fashion, etc.). With its applications ranging from detection and mapping of tree seedlings [3], to assistance of visually impaired [4], to star/galaxy classification[5], and everything in between.

Similarly, it is being used in sports analysis[6]–[8] by coaches and analyst experts to understand and improve player performance. Considering the popularity of badminton, it has become the most prevalent sport around the world. We aim to combine the techniques of Object detection and classification with, Internet of Things (IoT) to make the data accessible and feasible for implementation by the institute.

This technique can be used easily to detect and identify objects present in an image. With a large Dataset, faster GPU, and better algorithm, we can train computers to detect players and classify various sports using OpenCV. Object detection deals with identifying and detecting object instances in a visual frame belonging to various classes (including animals, trees, humans, fashion, etc.).A wide range of computer vision applications has become available for Object Detection and tracking. Real-world applications such as healthcare monitoring, autonomous driving, video surveillance, Star/galaxy classification, and everything in between. Sports Analysis is one of the fields it has been used in/for. For better understanding and analysis of strategies.

The Convolutional Neural Network (CNN, or ConvNet) is a class of deep neural networks that are most commonly applied to analyzing visual imagery. Their other applications include video understanding, speech recognition, and understanding natural language processing. Deep neural networks with several layers have recently become a highly successful and popular research topic in machine learning due to their excellent performance in many benchmark problems and applications. One of the most significant tasks in computer vision is computerized human motion analysis from video sequences, which allows the machine to identify, recognize, and interpret human motion to monitor human activities, communicate with humans, and control various equipment. The study of human motion analysis is fascinating and gaining tremendous attention, particularly in the last decade, with the development of applications in surveillance, human-computer interaction, sport sciences, entertainment, as well as other areas[9].

II. Contribution

The model we have developed will help future researchers implement their own models for real-time analysis or can also be used for a proper in-depth review of a match after it is over. The model for game classification will be helpful in many other tasks involving game detection and classification. The model we proposed for predicting and detecting fights on the court will become an essential asset for keeping peace on the playground.

III. Background

There has been a rapid and successful expansion of computer vision research(for object detection and classification). Parts of this success have come from adopting and adapting Machine Learning (ML) methods. Tracking players that use the Tracking-Learning-Detection (TLD) model is a viable alternative, and it is exact in tracking objects across time. The TLD model, on the other hand, is strongly influenced by pixel value gradients. This is a concern because most badminton court cameras are multiple monocular cameras at a great distance from the players. As a result, one of the players would be partially obscured by the net. Because the net and the background have such a great contrast, the TLD will miss-track the player. Image Compensation approaches, such as Image Inpainting [10], can help to mitigate this issue.TLD (Tracking-Learning-Detection) model is a viable option that has accuracy in tracking objects for a long time. However, it is highly affected by gradients of pixel values. And the solution for this is to minimize by the application of Image Compensation techniques. However, this method is not perfect and requires manual input to work at all times. Transfer learning is one of the most popular methods for customizing the model to fit our needs. Another research by[11] etc. all discussed various deep learning methods for object detection. The proposed work provided a comprehensive overview of the traditional and modern application of object detection. Authors also concluded that the ultimate goal and researchers had developed so much as constructing new architecture, extracting rich features, exploiting good representations, improving processing speed, training with the most extensive and finest dataset, solving sophisticated scene issues (small objects, occluded objects), improving post-processing NMS method increasing localization confidence.

A.

IV. Materials and Methodology

A. Player Detection Model

The process of player detection is divided into following steps:

- Model Identification
- Data Collection
- Using model to detect objects

1) Model Identification: The main objective of our project was to find the occupancy of the IISERB badminton court. For this, they need a model which can accurately detect any person in the image/video. So we started researching different pre-trained models which were able to detect humans. We researched into YOLOv3, ImageNet, Alexnet, etc. After going through these models, we found that YOLO was our best option because of its speed and accuracy. It was able to excel in our prototype sample data to finalize the model used for our project.

a) What is YOLOv3? :: You Only Look Once (YOLOv3)[12] is a real-time object detection algorithm that identifies specific objects in videos, live feeds, or images. YOLO uses features learned by a deep convolutional neural network to detect objects [13]. Unlike region-based CNN, YOLO takes a single look and divides the image into a $S \times S$ grid of cells. For each object on the image, one grid cell is supposed to be responsible for having that object at its center. It allows the model to look at the whole image at test time, so the global context in the image informs its predictions. YOLO and other convolutional neural network algorithms "score" regions based on their similarities to predefined classes. High-scoring regions are noted as positive detections of whatever class they most closely identify with. For example, in a live traffic feed, YOLO can be used to detect different kinds of vehicles depending on which regions of the video score highly compared to predefined classes of vehicles.

b) The YOLO Architecture at a glance : : The YOLOv3 algorithm first separates an image into a grid. Each grid cell predicts some number of boundary boxes (sometimes referred to as anchor boxes) around objects that score highly with the aforementioned predefined classes.Each boundary box has a respective confidence score of how accurate it assumes that prediction should be and detects only one object per bounding box. The boundary boxes are generated by clustering the dimensions of the ground truth boxes from the original dataset to find the most common shapes and sizes.Other comparable algorithms that can carry out the same objective are R-CNN (Region-based Convolutional Neural Networks made in 2015) and Fast R-CNN (R-CNN improvement developed in 2017), and Mask R-CNN.However, unlike systems like R-CNN and Fast R-CNN, YOLO is trained to do classification and bounding box regression at the same time.

2) Data Collection: First, we identified the problem we wanted to solve, i.e., detecting the players on the court and the audience. Then for this purpose, we needed to collect data to feed into our model. So, we used a badminton video we found on the internet.

3) Using model to detect objects: With YOLO as our preferred model for object detection, we made a is object detection setup with OpenCV as backend. OpenCV(Open source Vision Library) is an open-source computer vision and machine learning library. Using it, one can process images and videos to identify objects, faces, or even handwriting of a human. When it is integrated with various libraries like Numpy, python can process the OpenCV array structure for analysis. We use vector space to identify image patterns and their various features and perform mathematical operations on those features. For our model, it is used as the backend for YOLO V3. In this section, we will load the model using the function ”cv2.dnn.ReadNet()”. This function loads the network into memory and automatically detects configuration and framework based on the specified file name.

4) Working of model: First, we started by loading the video of players playing badminton, then set the framerate at five fps so that model would not be slowed down due to performance constraints. Each frame of the loaded video is converted into a blob as an input to our model. The blob is also passed to the trackbar2 function, which displays the blob with all the initial YOLOV3 predicted bounding boxes for each image and the confidence values that we want for each predicted box. From the output of the trackbar2 function, we got a slider that can be used to adjust the prediction confidence that the user requires dynamically.

The outputs received has the detected class, confidence, and bounding box values. Then these outputs are passed into the function, which takes the bounding box values and overlays a bounding box around the detected object along with its confidence values and predicted class. After successfully collecting data and loading the YOLO V3 model using OpenCV, we loaded it into the blob and passed it into the model. Then the model took input and gave us output containing confidence values, class ids, predicted bounding boxes. Then these outputs are passed into the function, which outputs a result as an image with confidence values greater than 50%, class ids.

B. Sports Classification

The purpose of building the sports classification model was to complement our previous player detection model. Coaches can use these two models to work together in players’ performance enhancement[14].

- Data Collection
- Data Preprocessing and Labeling
- Model Creation

1) Data Collection: The goal of this model was to classify different sports using images of grounds and equipment accurately. For this purpose, we tried making our dataset of images of different sports, but due to the lack of high-quality free images, we scoured the internet for the pre-made dataset of sports images and found one. This dataset had about 12000 images belonging to 22 different sports.

2) Data preprocesing and Labelling: The dataset that we got had a lot of unusable images like having bad quality, unrelated images. Then, this dataset had to be cleaned and processed to extract all the valuable images for our model training and testing.

a) Data Cleaning and Labelling: Since all the 12K images present in the dataset were not usable, we had to do much manual cleaning. At last, after deleting all the unusable images, we were left with a fraction of the dataset. Furthermore, from that, we selected five sports that are to be classified. This resulted in a new dataset containing around 650 images of the five selected sports classes: badminton, basketball, hockey, cricket, and volleyball. For our convenience, the dataset collected was already split into different sport classes.

3) Model Creation: Sports Classification model required usage of various libraries like Keras [15], [16], matplotlib, and NumPy. This model used CNNs [17] to classify Sports images by extracting various features of the input images 4.

a) What are CNNs: Neural networks are a subset of machine learning and the core of deep learning algorithms. They comprise node layers containing an input layer, one or more hidden layers, and an output layer.

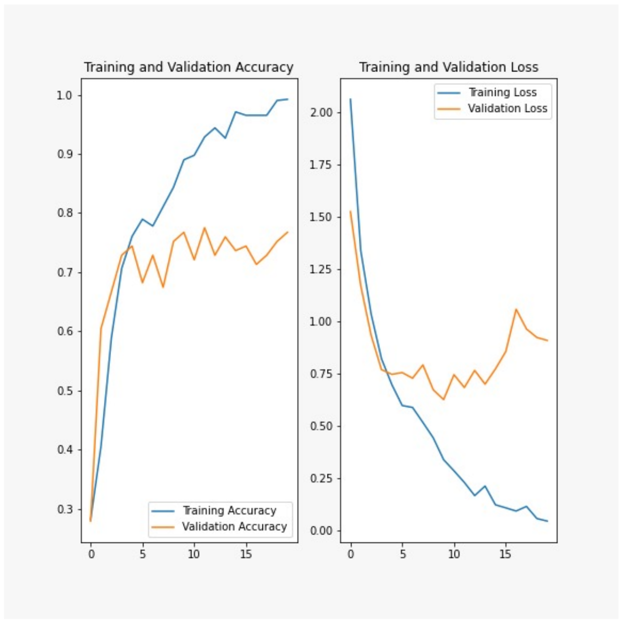

Fig. 1. Training and Validation Graph for Sport Classification Model

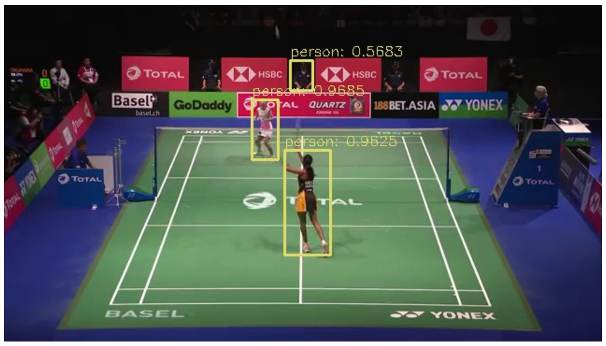

Fig. 2. Player detection

Each node connects to another and has an associated weight and threshold. If the output of any individual node is above the specified threshold value, that node is activated, sending data to the next layer of the network. Otherwise, no data is passed along to the next layer of the network.

Convolutional neural neworks are distinguished from other neural networks by their superior performance with image, speech, or audio signal inputs. They have three main types of layers:

- Convolutional layer
- Pooling layer
- Fully connected (FC) layer

Convolutional Layer is the core building block of CNN, and it is where the majority of computation occurs. It requires a few components, which are input data, a filter, and a feature map.
The feature detector is a two-dimensional array of weights that represents part of the image. With typically a filter size of $3\times, 3$ matrix, which is then applied to an area of the image, the dot product is calculated between input pixels and the filter. This dot product is fed into an output array, then the filter shifts by a **Stride**. This process is repeated until the kernel has swept across the entire image. The final output from this series of dot products from the input and the filter is a feature map, activation map, or a convolved feature. Three hyper-parameters affect the volume size of the output that needs to be set before the training of the neural network begins.

1) The **number of filters** affects the depth of the output. For example, three distinct filters would yield three different feature maps, creating a depth of three.
2) **Stride** is the distance or several pixels that the kernel moves over the input matrix. The larger the Stride smaller the output it would yield.
3) **Zero-padding** is used when filters do not fit the input image. This sets all elements outside the input matrix to zero, producing a larger or equally sized output.
   - **Valid adding** is also known as no padding. In this case, the last convolution is dropped if dimensions do not align.
   - **Same padding** ensures that the output layer has the same size as the input layer
   - **Full padding** increases the size of the output by adding zeros to the border of the input

Pooling layer conducts dimensionality reduction, reducing the number of parameters in the input. The pooling operation sweeps a filter across the entire input, but the difference is that this filter does not have any weights. Instead, the kernel applies an aggregation function to the values within the receptive field.

- **Max Pooling:** As the filter moves across the input, it selects the pixel with the maximum value to send to the output array. As an aside, this approach tends to be used more often compared to average pooling.
- **Average Pooling:** As the filter moves across the input, it calculates the average value within the receptive field to send to the output array.

Fully Connected Layer The name of the full-connected layer aptly describes itself. The pixel values of the input image are not directly connected to the output layer in partially connected layers. However, in the fully-connected layer, each node in the output layer connects

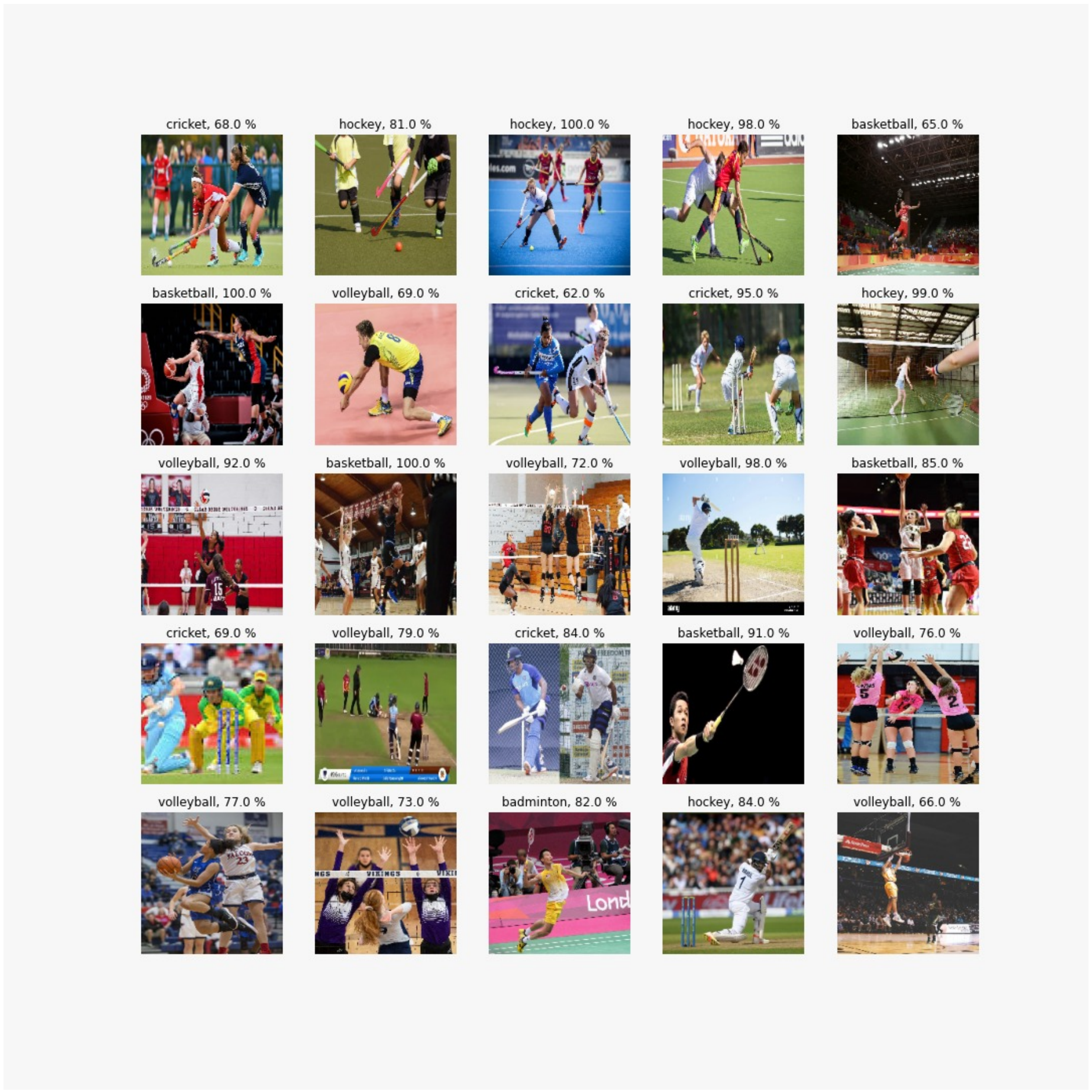

Fig. 3. Sport Classification

directly to a node in the previous layer.
This layer performs the task of classification based on the features extracted through the previous layers and their different filters. While convolutional and pooling layers tend to use ReLu functions, FC layers usually leverage a softmax activation function to classify inputs appropriately, producing a probability from 0 to 1.

b) Data Augmentation: Data Augmentation is a process of increasing the available limited data to large, meaningful, and more diverse amounts. In other terms, we are artificially increasing the size of the dataset by creating different versions of the existing data from our dataset. In order to train a model for accurate results, we need to have parameters to learn almost all the features from the data. To accommodate all these parameters, we need to have a good amount of data. Deep learning models often require more data which is not always available. For the model we used **Horizontal flip** and **Rescaling Techniques**

4) Compilation of Program: To train the model we are using **Adam** as optimizer function and **Categorical Crossentropy** as loss function.

- **Adam** optimization is a stochastic gradient descent method that is based on adaptive estimation of first-order and second-order moments.
- **SparseCategoricalCrossentropy** is used for computing the loss value of your neural network.

$$J(\mathbf{w}) = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)]$$

  - w refer to the model parameters, e.g. weights of the neural network
  - y_i is the true label
  - $\hat{y_i}$ is the predicted label

5) Training the model: After all the data preprocessing, data augmentation and model building the final step was training the model. Setting epochs to 20, we trained the model which gave us an acccuracy of $> 95\%$, with a validation accuracy of $> 70\%$.

V. Results

A. Player Detection

This model was built on YOLOV3 and OpenCV as its backend. YOLOV3 is an object detection algorithm trained on the COCO dataset. We leveraged its detection capabilities and applied them for the detection of players on the court. Our model was successfully able to detect players on the field with greater than 90% accuracy 2. Since this model is built on OpenCV, it can be used with a wide variety of inputs such as images, pre-recorded sports videos, and live footage.

B. Sports Classification

With complimentary to our model mentioned above, our sports classification model helps classify different sports played in IISERB 3. We trained this large model dataset that we extracted and cleaned from a publicly available dataset. This model produced up to the mark results, producing an accuracy greater than 95 % 1, considering it was only trained for 20 epochs.

VI. Discussions

We tried applying the transfer learning technique to the YOLO model heavily used for the player detection model. The purpose was to build a robust and accurate player detection system that would detect players in challenging circumstances like shaky, blurry video or from the poor quality webcam present at the court. This meant we had to create our dataset of images. For this, we had first to collect many images of players playing badminton, then clean all data and label them in YOLOV3 format. However, the model kept failing at the training step, which delayed its progress.

VII. Conclusions

With both models working complimentary to each other, we can detect moving players from a match, detect what game they are playing, and dynamically change the relevant information that needs to be extracted from the game. With some more work and fine-tuning of our models, we expect our setup to be used at any court for real-time player analysis without prior manual work setting up the model.

References

[1] S. Lu, B. Wang, H. Wang, L. Chen, M. Linjian, and X. Zhang, "A real-time object detection algorithm for video," *Computers & Electrical Engineering*, vol. 77, pp. 398–408, 2019.

[2] Z. Zou, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *arXiv preprint arXiv:1905.05055*, 2019.

[3] G. D. Pearse, A. Y. Tan, M. S. Watt, M. O. Franz, and J. P. Dash, "Detecting and mapping tree seedlings in uav imagery using convolutional neural networks and field-verified data," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 168, pp. 156–169, 2020.

[4] F. S. Bashiri, E. LaRose, J. C. Badger, R. M. D'Souza, Z. Yu, and P. Peissig, "Object detection to assist visually impaired people: A deep neural network adventure," in *International Symposium on Visual Computing*, Springer, 2018, pp. 500–510.

[5] S. Andreon, G. Gargiulo, G. Longo, R. Tagliaferri, and N. Capuano, "Wide field imaging—i. applications of neural networks to object detection and star/galaxy classification," *Monthly Notices of the Royal Astronomical Society*, vol. 319, no. 3, pp. 700–716, 2000.

[6] M. Burić, M. Pobar, and M. Ivašić-Kos, "Object detection in sports videos," in *2018 41st International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, IEEE, 2018, pp. 1034–1039.

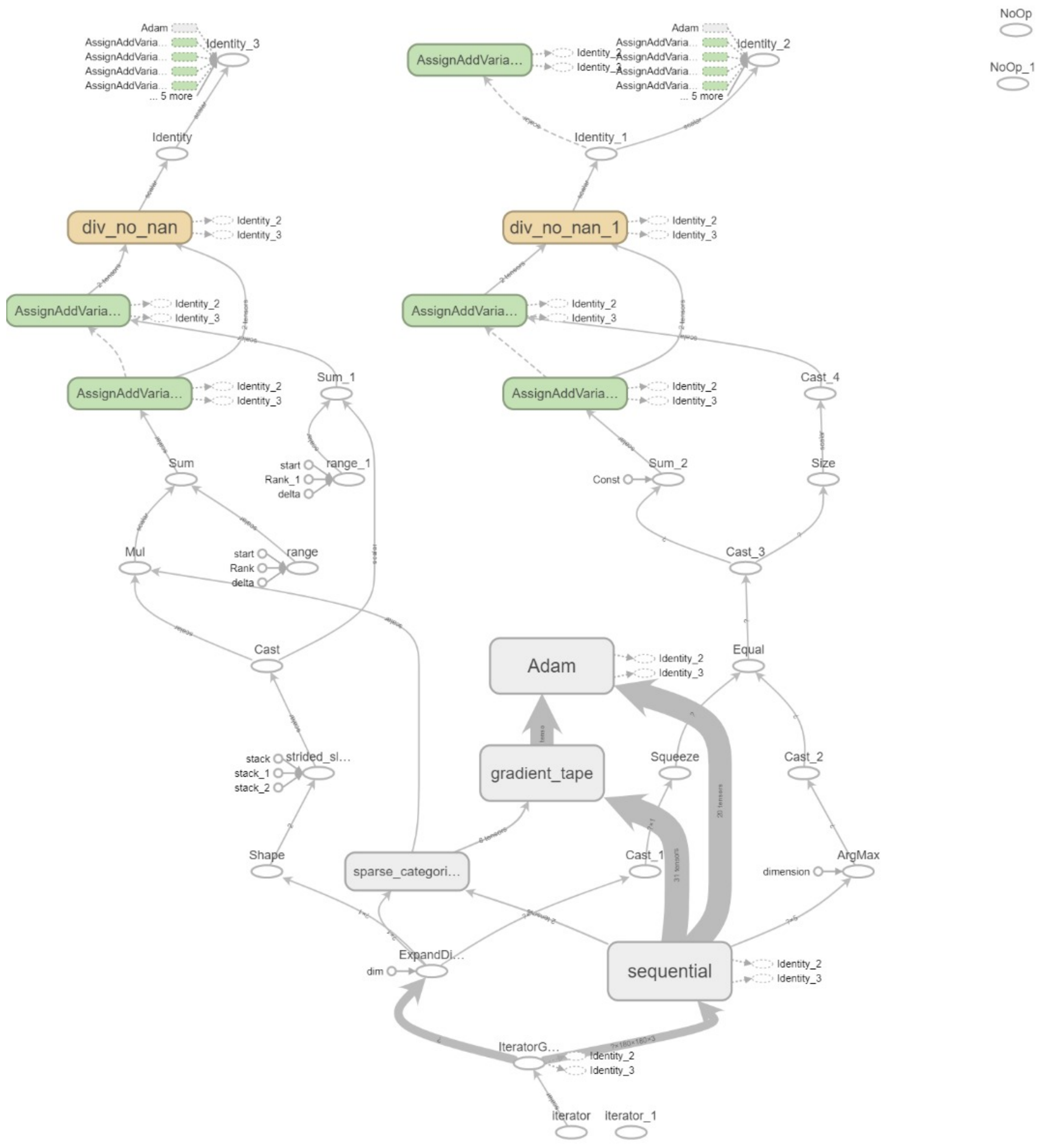

Fig. 4. Model architecture of Sport Classification Model

[7] R. Manikandan, R. Ramakrishnan, and R. Scholar, "Human object detection and tracking using background subtraction for sports applications," *International Journal of Advanced Research in Computer and Communication Engineering*, vol. 2, no. 10, pp. 4077–4080, 2013.

[8] O. Utsumi, K. Miura, I. Ide, S. Sakai, and H. Tanaka, "An object detection method for describing soccer games from video," in *Proceedings. IEEE International Conference on Multimedia and Expo*, IEEE, vol. 1, 2002, pp. 45–48.

[9] D. Tan, H. Ting, and S. Lau, "A review on badminton motion analysis," in *2016 International Conference on Robotics, Automation and Sciences (ICORAS)*, IEEE, 2016, pp. 1–4.

[10] T. Kamiyama, Y. Kameda, Y. Ohta, and I. Kitahara, "Improvement of badminton-player tracking applying image pixel compensation," *ITE Transactions on Media Technology and Applications*, vol. 5, no. 2, pp. 36–41, 2017.

[11] L. Jiao *et al.*, "A survey of deep learning-based object detection," *IEEE access*, vol. 7, pp. 128 837–128 868, 2019.

[12] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.

[13] L. Zhao and S. Li, "Object detection algorithm based on improved yolov3," *Electronics*, vol. 9, no. 3, p. 537, 2020.

[14] P. Chelladurai, "A classification of sport and physical activity services: Implications for sport management," *Journal of Sport Management*, vol. 6, no. 1, pp. 38–51, 1992.

[15] K. Seetala, W. Birdsong, and Y. B. Reddy, "Image classification using tensorflow," in *16th International Conference on Information Technology-New Generations (ITNG 2019)*, Springer, 2019, pp. 485–488.

[16] J. D. Kothari, "A case study of image classification based on deep learning using tensorflow," *Jubin Dipakkumar Kothari (2018). A Case Study of Image Classification Based on Deep Learning Using Tensorflow. International Journal of Innovative Research in Computer and Communication Engineering*, vol. 6, no. 7, pp. 3888–3892, 2018.

[17] V. K. Gunjan, R. Pathak, and O. Singh, "Understanding image classification using tensorflow deep learning-convolution neural network," *International Journal of Hyperconnectivity and the Internet of Things (IJHIoT)*, vol. 3, no. 2, pp. 19–37, 2019.