CE6051
Group Project

Classification of Cricket Shots using Pose Estimation

Group 7

CE21B017 Anshul Chawan
CE21B043 Dhawal Shukla
CE19B008 Koushik Abhiram
ME19B027 Mevit Mathew
ME19B176 Subham Roy
CE21B124 Suparn Gupta

Introduction

Human pose estimation (HPE) is a computer vision task that involves detecting and estimating the position of various parts of the human body in images or videos. This field focuses on localizing human body joints, also known as key points.

Pose estimation models can track objects or individuals, including multiple people, within real-world environments. Unlike object detection models, which identify objects in an image with bounding boxes providing rough localization, pose estimation models predict the exact positions of key points related to a specific object.

Human pose estimation technology finds applications in various domains:

1. Healthcare: It is used to analyze and diagnose patient movements during rehabilitation after injury or surgery and to assess gait for orthopedic and neurological purposes.
2. Fitness and Sports: It helps track and analyze workout form and technique to prevent injuries and enables the creation of virtual fitness trainers for personalized workouts.
3. Entertainment and Gaming: Human pose estimation enables gesture-based control in video games and virtual reality (VR) applications.
4. Retail and E-commerce: It improves virtual try-on experiences for clothing and accessories and enhances augmented reality (AR) shopping applications.
5. Gesture and Communication: This technology allows users to control technology and devices using gestures and body movements.
6. Autonomous Vehicles: Human pose estimation enhances driver monitoring systems to ensure alertness and safety. It also improves pedestrian detection and avoidance in self-driving cars, among other applications.

Problem Statement

The pose estimation problem that is tackled in this project is classification of cricket shots. In cricket, batsmen attempt different types of shots depending on the incoming trajectory and desired outgoing trajectory of the ball. Conventionally, the shots have been categorized into different classes like drive, sweep, pull, glance etc. This categorization is based on the posture and body movement of the batsman during the shot.

In this project, the objective is to classify the shot being played based on just a single still image captured during the action of batting.

Methodology

We use machine learning techniques, specifically pose estimation, to predict the class of the shot from the posture of the batsman.

1. Dataset

   The dataset used for training is a collection publicly available images of professional batsmen playing shots. The images were then labelled as per the class of shot being played into two classes, ‘pull’ and ‘sweep’. To accommodate for use cases where batsmen do not wear professional gear, we also captured our own images and labelled

them. A very small number of these images are added to the training dataset. For pre-processing, the images are reshaped to the 192x192 size that the pose estimation model expects. The low resolution is not an issue since we utilize only the pose. For validation, we use rest of the images captured of ourselves playing cricket shots.

Figure 1: Sample image from training dataset (Class: 'Pull Shot')

2. Environment

The entire project is hosted on Google Colab using python as the programing language. The dataset is stored in Google Drive which is then mounted on the notebook. The Nvidia T4 Tensor Core GPU, made available by Colab, is used as hardware accelerator.

3. Pose estimation

First, information about the pose of the batsmen is extracted from the image. For this, we use the open-source pose estimation provided by TensorFlow: MoveNet Lightning. The information is extracted in the form of key-points. These key-points estimate the pose of the target and can be easily visualized as depicted in figure 2. We store the coordinates of key-points as arrays for downstream processing.

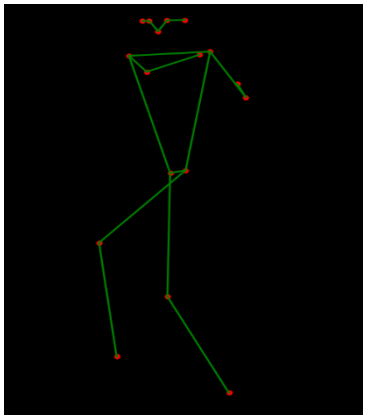

Figure 2: Pose extracted from the sample image

4. Classification

The extracted key-points are used in a machine learning algorithm to classify the images. A binary classification is done for the two classes, 'pull' and 'sweep'. XG Boost is the

classification algorithm used. We attempted different algorithms including Logistic Regression, Random Forest, Support Vector Machines and Multi-Level Perceptron. XG Boost was chosen based on better performance over these models. The input to the model is geometric coordinates of the key-points and the output is the class of shot.

Models

1. MoveNet

MoveNet is an efficient and effective deep learning model designed for the purpose of human pose estimation. Developed by Google, MoveNet is part of TensorFlow's Model Garden and is optimized to run in real-time on various devices, including those with limited computational resources such as mobile devices and edge computing platforms. It utilizes a Convolutional Neural Network (CNN) to detect human joints. The model is trained on images that include a diverse range of human poses.

CNN is a type of deep neural network that is particularly powerful for processing data that has a grid-like topology, such as images. CNNs use a mathematical operation called convolution which involves sliding a smaller array, known as a kernel or filter, over the data to produce a feature map that emphasizes certain features in the input. This process helps the network to capture spatial hierarchies in data and makes CNNs highly effective for tasks like image recognition, video analysis, and natural language processing. Their ability to learn complex patterns through layers of filters makes them a cornerstone of modern artificial intelligence applications.

We have used MoveNet as our pose estimation model because of its versatility and ease of use. It is straightforward to integrate with just a few lines of code and gives good results with quick inference time. MoveNet is available on TensorFlow Hub in two variants: Lightning and Thunder. Given that our application requires pose estimation of just a single target, we have used the Lightning model which is computationally leaner.

2. XG Boost

XG Boost stands for Extreme Gradient Boosting, a popular and powerful machine learning algorithm for regression and classification tasks. It is a very popular machine learning algorithm based on the gradient boosting framework, which builds models sequentially by combining the predictions of previously built models. Key Features of XG Boost are:

1. Gradient Boosting: XG Boost employs a sequential approach, correcting errors of preceding models to minimize loss.
2. Decision Trees: XG Boost's base learners are CART decision trees, adept at partitioning data space and assigning labels.
3. Regularization: With L1 and L2 regularization, XG Boost prevents over-fitting by penalizing large coefficients.
4. Tree Pruning: XG Boost prunes trees while building to prevent over-fitting, halting node splitting upon encountering negative loss.
5. Parallelization: Designed for speed, XG Boost supports parallelization for efficiently utilizing multicore processors and distributed computing platforms.

6. Feature Importance: XG Boost offers feature importance scores to gauge each feature's predictive significance, aiding feature selection and model understanding**.**

Methodology of XG Boost:

The gradient boosting ensemble technique consists of the following steps:

1. An initial model F_0 is defined to predict the target variable y. This model will be associated with a residual ($y - F_0$)
2. A new model h_1 is fit to the residuals from the previous step
3. Now, F_0 and h_1 are combined to give F_1, the boosted version of F_0. The mean squared error from F_1 will be lower than that from F_0: $F_1(x) = F_0(x) + h_1(x)$
4. To improve the performance of F_1, we could model after the residuals of F_1 and create a new model F_2: $F_2(x) = F_1(x) + h_2(x)$
5. This can be done for 'm' iterations until the residuals have been minimized sufficiently

The loss function that the algorithm minimizes is:

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(x_i)) + \Omega(f_t)$$

$y_i - \ Real\ value\ (label)$
$\hat{y}_i^{(t-1)} + f_t(x_i) - \ Analogous\ to\ f(x + \Delta x)\ where\ x = \ \hat{y}_i^{(t-1)}$
$l - A\ function\ of\ CART\ learners$
$\Omega(f_t) - \ Regularization\ term$

We have implemented XG Boost using Scikit-, a machine learning library of python that provides easy integration of different machine learning techniques.

Results

By using the methodology and model described, we have achieved perfect accuracy in classification of images in the validation set. This is despite the relatively less data used. XG Boost was the model that classified perfectly with least data. This is expected due to structured nature of key-points data. . It is worth noting here that the algorithm does not process the image itself or attempt image classification. It merely uses the key-points data extracted by MoveNet.

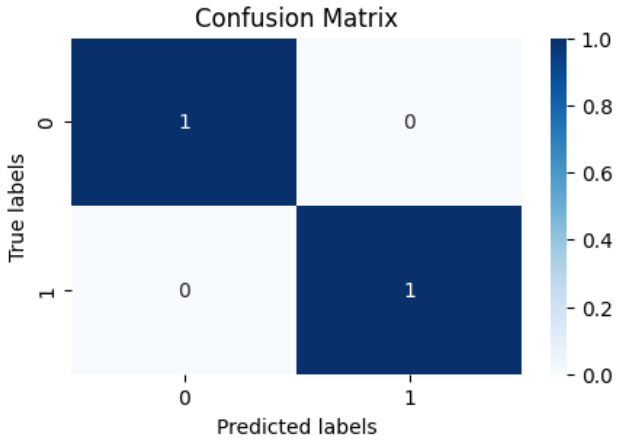

Figure 3: Confusion matrix of classification

Conclusion

The project has demonstrated the effectiveness of using pose estimation for classification tasks. The outcome highlights the potential of using data extracted by one machine learning model for further models downstream. This also improves the interpretability of the model over using only a traditional CNN for classification task. Further scope of work is to use this methodology for training athletes. For example, if data on the ‘right’ way to play a shot is obtained from sports experts, this model can be used to analyze and train batsmen.