Cricket Shot Classification

Miles Murphy

October 2024

1 Introduction

Technology has become ingrained in sport analytics in recent years, with large groups of people working for many sports teams across almost all sports. Cricket has been a sport that this pioneering technology has often been introduced in which examples such as ball tracking and the Decision Review System. Cricket has a rich history of naming specific shots, having application to these analytics teams into categorizing them. In this Paper we propose a lightweight neural network (NN) to classify shots and we show initial results of this model.

2 The Architecture

Existing models have been built to classify shot types with impressive accuracy. However, these examples often use Convolutional Neural Networks (CNNs) which can pose a computational hurdle for teams with less computing power [1] . This motivation has led us to building a NN with pre-processed data from Mediapipe's Pose Detection Algorithm.

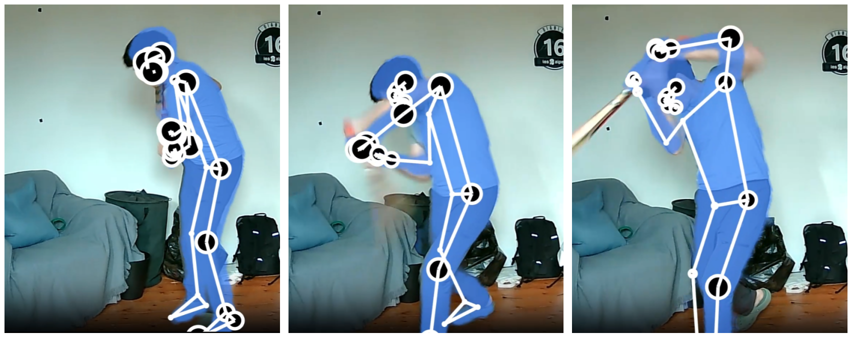

Figure 1: Cover Drive Sequence with Mediapipe overlay

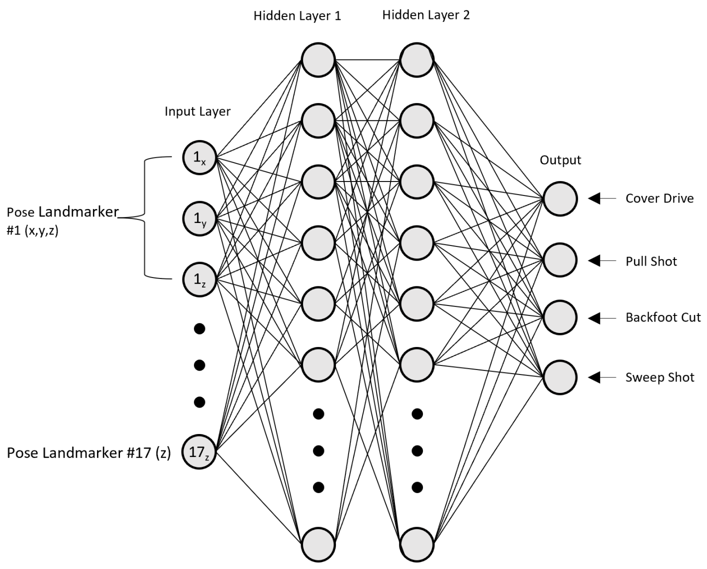

Figure 2: Simplified View of NN structure (Fully Connected and ReLU's omitted)

2.1 Pose Detection

Pose detection algorithms have been produced since the dawn of computer vision technology, with a wide range of applications from medical to the video game industry. These algorithms are able to detect key-points on the user's body that can be overlaid "drawing the user out". Google's Mediapipe team have released a real-time pose detection model that is lightweight and open source [2].

This model can take photo, video and livestream input and produce 33 key-point landmarkers as seen in Figure 1. We use this to preprocess the video input. To further reduce the data, we take only the relevant land markers as the mediapipe model has a number of facial point markers which are not relevant to our use case. Taking only the relevant points we run the training videos through the mediapipe API outputting 17 keypoints for every frame. This gives us a total input size of $17 \times 3 = 51$ (due to the x,y,z components) per frame.

2.2 Neural Network Structure

We implement a Long Short Term Memory (LSTM) structure to the network to capture the sequential data we have. We use two LTSM layers with a layer size of 256 to capture the temporal dependency (shown in Figure 2) connected with two fully connected layers taking the neural net from 256 to 64 to the final four node classes. We introduce the non-linearity with a ReLU layer in between the fully connected layers. To reduce over fitting we use dropout. The LSTM layer only passes on the hidden state of the last frame which for relatively short sequences (compared to other LSTM use cases) will be sufficient.

3 Data

Initially, we intended to use publicly available broadcast footage of examples of the respective shots, however this caused issues due to the zoom the camera operator does as the bowler approaches the crease. In order to have autonomy over camera position we chose to use a webcam and performed the shots in front of it. This relies on the batter to play varied version of each shot (limitations of this method are considered in the discussion). In total around ≈ 50 of each shot were played for each shot type. We discuss how we increase the effective size of this dataset.

3.1 Data Augmentation

A first method to increase the size of the dataset was to add noise to each data point. A script was implemented to add a Gaussian noise to each key point for each frame of the key point set. We set a mean of 0 and standard deviation of 0.05. In doing this, we reduce the dependence on the example batter and effectively double our dataset.

We also double our dataset a second time by mirroring our videos to add a left hand batting style examples. Using the same process of data augmentation with the random noise we are able to have ≈ 200 examples of each shot class.

3.2 Future Data Modifications

In the future we would like to have other batter's examples of shots. For professional examples, we note that many County Championship broadcasts do not have the aforementioned zoom which helps the pose detection algorithm perform better. Further access to existing footage, or the ability to record more shots is also possible to improve the model with more examples.

4 Model Performance

4.1 Model Training

The model is trained on the training data with a batch size of 16. We have a training split of 70% training, 20% Validation and 10% testing. The model then learns for 2000 epochs with a training accuracy of 98.38% with a Validation accuracy of 89.31%. Our model boasts an accuracy on par with previous work in CNNs with LSTM structure [3].

We use a log-loss function to calculate our loss values as it is suited towards the wider structure of our model and is differentiable. The loss curve seen in Figure 3 shows some instabilities, further work should be done the address these effects.

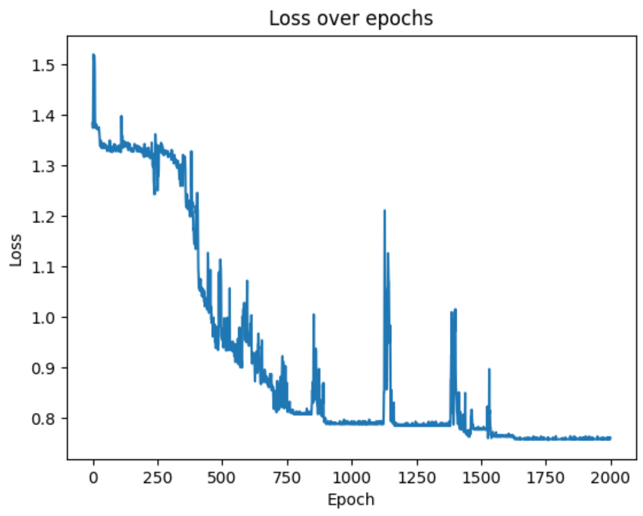

Figure 3: Training loss over epochs

4.1.1 Model Use

In the model testing phase we see an accuracy of 76.25% when fed unseen similarly framed videos. The model performs fairly well on videos of other batters playing in front of a camera like the training. Issues arise more in the tracking implementation such as in scenarios where the batter is crowded. The pose detection algorithm can track multiple people however, implementation of this would require identification of the batter prior to feeding the landmarks into the model.

4.1.2 Use Cases

This project has shown a proof of concept for a lightweight NN to classify four types of cricket shots. There is plenty of room for expansion of this model and the use cases. For now a such a model would have place for those teams or persons with less technological faculties.

5 Conclusions

We have presented a novel application of this NN structure to classify four types of cricket shots. Cricket Shot classification has use cases in batting analysis and statistical logging. The lightweight model can run on a personal computer and use regular video cameras. The project has room to grow to more cricket shots and more sophisticated use cases.

References

[1] M. Jagadeesh, Rithesh S, and Sagar Y. "Cricket Shot Detection Using Deep Learning: A Comprehensive Survey". In: *2023 International Conference on Networking and Communications (ICNWC)*. 2023, pp. 1–8. DOI: `10.1109/ICNWC57852.2023.10127412`.

[2] Ivan Grishchenko Valentin Bazarevsky and Eduard Gabriel Bazavan. *MediaPipe BlazePose GHUM 3D*. Computer Program. 2021. URL: `https://storage.googleapis.com/mediapipe-assets/Model%20Card%20BlazePose%20GHUM%203D.pdf`.

[3] gouthamvgk. *deep$_c$ricket*. Computer Program. 2018. URL: `https://github.com/gouthamvgk/deep_cricket`.