Badminton Stroke Prediction Based on Player's Pose

Karthik Viswanath Sriram
1222696995
Arizona State University
Team 42

Veena Madhuri Tumuluri
1232463962
Arizona State University
Team 42

Abstract

Machine Learning has been used in the field of sports from the time of its birth and has been around for almost 10 years. Sports such as Cricket, Soccer, Football and many more have benefited greatly from the ML boom. Cricket in particular has used ML techniques extensively to predict the course of the ball, the action of the batsmen based on motion data. Badminton has not benefited from ML technologies as much as other sports. In this study, we aim to predict the type of stroke the badminton player will use depending upon the position of the various joints in the body. The player's pose data is extracted from images and a classification algorithm is run to determine the type of stroke which may be played. Zero-shot classification technique is also explored by fine-tuning VGG16 and using it to classify images sourced form google. This study will assist many beginner players to analyse the pose of the opponents and be prepared for the game play. From a research standpoint, it also helps to understand and analyse the posture of professional badminton players thereby learning a lot from them.

1 Introduction

The sport of Badminton has only seen minor use cases of Machine Learning such as shuttle trajectory mapping and win/loss prediction based on the player's statistics. Beginner badminton players often get confused on what kind of shot to expect from the opponent and therefore cannot respond with the correct countermeasure. If we are able to predict the stroke of the opponent, then it will be easier to react with the correct response to the stroke securing the point for the rally. Predicting the type of stroke based on the player's position on the court will provide very useful information for coaching as well as for analysis purposes. In this work, the player's pose on the court is taken as an image and information on the various body joints is extracted using yolov3 pose estimation model. The keypoints are then run through a classification algorithm which can predict whether the stroke can be one of the three possible types such as lift, net shot or a smash.

SOTA techniques such as Zero shot classification is also explored and analysed for its performance. Using Pre-trained models and fine-tuning the model by freezing the parameters, training only the last layer and using it for classification tasks using test data from another dataset is referred to as zero shot classification. VGG16, a pre-trained model is taken and finetuned on an image dataset containing images of players in the mist of a rally. The model is tested using images sourced from google and the performance of a model trained on YOLOV7 pose coordinates and the pre-trained model finetuned on image dataset are analysed.

2 Related Work

Y. E. Tan et al. [1] conducted research focused on predicting rally outcomes by analyzing previous strokes. Their approach utilized Resnet18 for feature extraction of individual strokes and employed

Preprint. Under review.

LSTMs to forecast rally outcomes based on the sequence of preceding strokes. They utilized tournament videos, extracting frames to establish stroke sequences for rally result classification.

Weeratunga et al. [2] developed player profiles by categorizing player movements using player location and shuttlecock trajectory. Building upon this, Chu and Situmeang [3] expanded on prior work by integrating additional tasks like player tracking and stroke classification. This research also distinguished badminton strategies into offensive and defensive categories based on historical data.

Ghosh et al. [4] endeavored to automate badminton game analysis. Their automated processes encompassed game section segmentation, player detection and tracking, and stroke classification. The outputs generated by these automated tasks served as analytical metrics for further study.

Study done by Nuttachot Promrit et. al. [5] integrates Triplet-Loss embedding for video posture representation and One-Shot Learning for motion posture detection. It addresses crucial aspects of injury prevention and skill enhancement in badminton, offering a comprehensive solution for players of all ages and genders. The model's performance metrics demonstrate its effectiveness, with Precision of 0.856, Recall of 0.845, F-Measure of 0.847, and Accuracy of 0.845. This innovative approach aligns with the growing trend of utilizing advanced techniques in sports training, contributing to the development of efficient and effective training methodologies.

3 Datasets

3.1 Motion Data for Badminton Shots

Motion Data for Badminton Shots [6] is chosen as the dataset to train a custom neural network model. The dataset is obtained from Kaggle which is an online repository of datasets, models etc. This Dataset has 34 features in the form of X and Y coordinates for 17 different keypoints in the body such as eyes, ears, nose, elbows, knees, ankles etc. The dataset is part of the IC2 internship program and contains motion capture data acquired from online video broadcasts of professional badminton matches. The type of shot played by the player is also annotated in the dataset. A snapshot of the dataset is shown in Table. 1. The dataset contains 10348 rows and 34 columns.

type-of-shot	kpt-0-x	kpt-0-y	kpt-1-x	kpt-1-y	...	kpt-16-x	kpt-16-y
lift	861.3868	359.0673	865.5193	351.693	...	1019.442	572.4178
net shot	0.0	0.0	0.0	0.0	...	1198.1284	449.8348
smash	637.6047	338.5236	0.0	0.0	...	695.4926	681.2870

Table 1: Dataset

3.2 Shot-Type Detection Dataset

Since VGG16 is an image classification model, another dataset called Shot-Type Detection Dataset [7] is used which is an image dataset. This Dataset is used for finetuning VGG16 model and contains images of players in the midst of a rally. There are 5 classes of strokes - Drop, netshot, over defensive clear, serve and smash. There are 553, 31 and 17 images for Training, Validation and Testing respectively. A sample of the dataset is shown in Fig 1

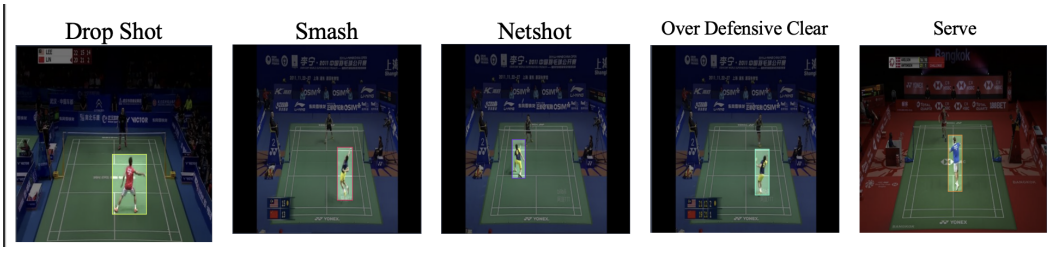


Figure 1: Shot Type detection dataset

4 Model architecture

The work is divided into various modules such as the pose estimation and the classification modules as described in the architecture diagram Fig 2.

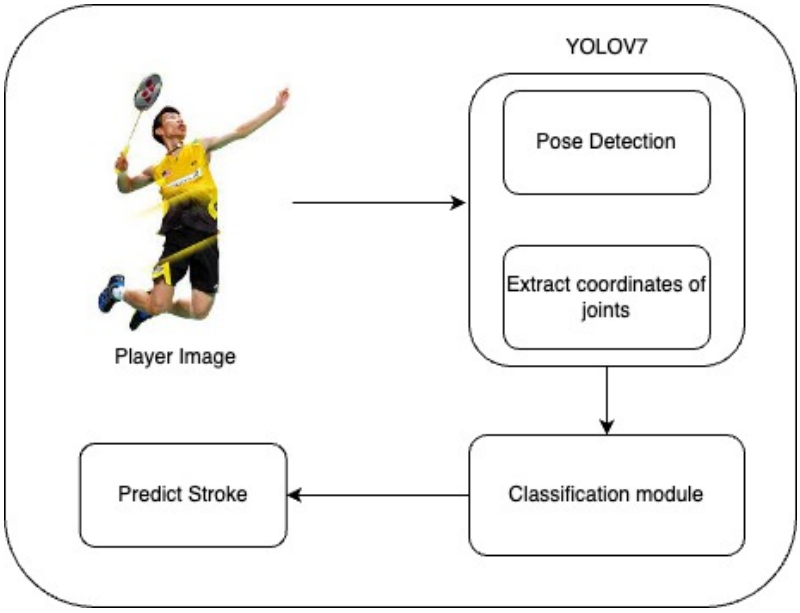


Figure 2: Architecture

The player's image will be given to YOLOV7 pose estimation model and the keypoints are extracted from the image. The keypoints are given to the model which is trained on the positional data to predict the type of stroke. Since VGG16 is finetuned on image dataset, the image can be used directly for classification and the stroke can be predicted.

5 Classification module

Two different approaches are explored in the classification task. A **neural network** is designed and trained on Motion Data for Badminton Shots Dataset which is a **numerical dataset**. **Zero-shot classification** is also explored and **VGG16**, a pre-trained model is retrieved from pytorch library and fine-tuned on Shot-Type Detection Dataset which contains **images**.

5.1 Neural Networks

A neural network is designed using layers from keras framework having 10 hidden layers. **Adam** is used as the optimizer and **sparse categorical cross entropy** is used as the loss function. This model is trained on the Motion Data for Badminton Shots Dataset which is a numerical data containing x, y coordinates for various keypoints of the body. Since this is not trained on acvtual image data, the image of the player has to be fed to the YOLOV7 pose estimation model to extract the keypoints before performing classification in the testing phase. The model is trained for 100 epochs with a batch size of 64. A train - test split of 80-10 is used for splitting the data. The neural network structure is shown in Fig. 3

5.2 Zero-shot classification

Zero-shot classification is a SoTA (State of The Art) technique used in industries where a model trained on a very large data corpus is used as it is by freezing most of its parameters and only fine-tuning the last classification layer on a custom dataset. This method saves a lot of time involved in training large models and also helps in getting SoTA performance when the data available for training is very less. In this study, we have used VGG16 which is a pre-trained model available from

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 128)               4480

 dense_1 (Dense)             (None, 128)               16512

 dense_2 (Dense)             (None, 128)               16512

 dense_3 (Dense)             (None, 128)               16512

 dense_4 (Dense)             (None, 128)               16512

 dense_5 (Dense)             (None, 128)               16512

 dense_6 (Dense)             (None, 128)               16512

 dense_7 (Dense)             (None, 128)               16512

 dense_8 (Dense)             (None, 128)               16512

 dense_9 (Dense)             (None, 3)                 387

=================================================================
Total params: 136963 (535.01 KB)
Trainable params: 136963 (535.01 KB)
Non-trainable params: 0 (0.00 Byte)
```

Figure 3: Neural Network Model

pytorch library. VGG16 is a convolutional neural network trained on a subset of the ImageNet dataset, a collection of over 14 million images belonging to 22,000 categories. Transfer learning is performed on the pre-trained VGG16 model by freezing the parameters and creating a custom dense layer to classify the results based on the number of classes available in our custom dataset. The architecture of the VGG16 model along with the section of adding the custom dense layer is shown in Fig.4

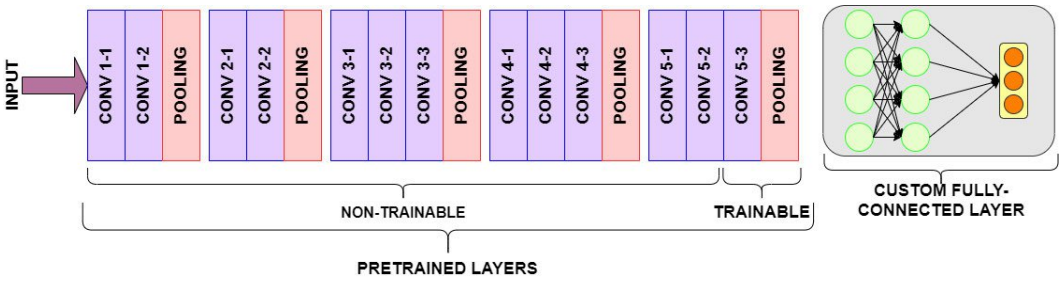


Figure 4: VGG16 model and Fine-tuning

6 Pose Estimation

The process of identifying the coordinates of the various joints in the body from an image is referred to as Pose estimation. Unlike conventional Pose Estimation algorithms, YOLOv7 pose is a single-stage multi-person keypoint detector. It is similar to the bottom-up approach but heatmap free. It is an extension of the one-shot pose detector – YOLO-Pose. It has the best of both Top-down and Bottom-up approaches. YOLOv7 Pose is trained on the COCO dataset which has 17 landmark topologies. It is implemented in PyTorch making it easier to customize as per our need. The YOLOV7 pose model parameters and the methods to download the weights and customize it are publicly available on github making it a very versatile tool to use for pose estimation.

During the testing phase of the neural network, random images from google are sourced and passed through the yolov7 model. The model first performs segmentation to identfy the people present in the frame and draws a bounding box surrounding the target. Then the various keypoints are identifed such as eyes, ears, nose, ankles etc as shown in Fig. 5. Once the Keypoints are identified, they are plotted over the original image and the keypoints themselves are returned which can be passed to the neural network as input for classification.

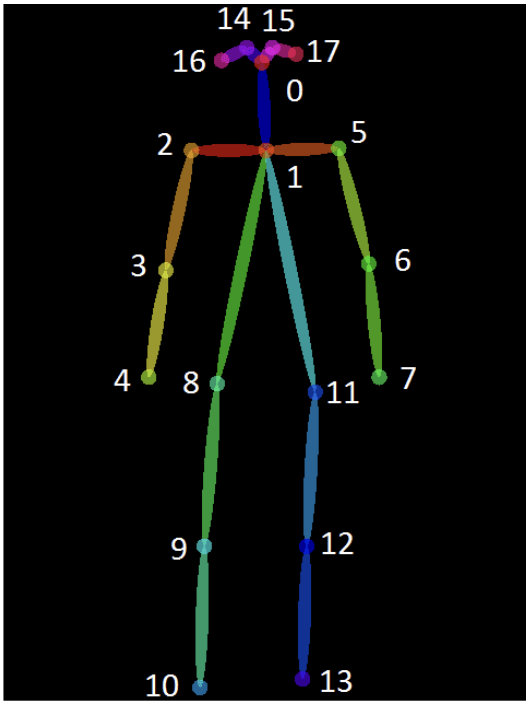


Figure 5: YoloV7 Keypoints

Fig. 6 shows a sample image in which the keypoints are overlayed on top of the original image. The keypoint array consists of 58 elements out of which the first 7 elements are metadata such as batchid, classid, X, Y coordinates of the center of the bounding box, width, height of the bounding box and confidence of the bounding box. The remaining 51 elements are x,y and z coordinates of the various joints of the body.

Figure 6: Pose estimation

The pose-estimation model is not required for zero-shot classification because VGG16 is trained on an image dataset and so the image can be passed directly for classification and the pose estimation coordinates are not required.

7 Results

The neural network is trained for 100 epochs on the numerical dataset ie. Motion Data for Badminton Shots. The model achieved a test accuracy of 81 percent on the dataset. When random images from google were passed through yolov7 to get the pose coordinates, the model was able to correctly classify half of the images into 3 classes such as smash, netshot and a drop. Fig. 7 shows the training phase of the neural network and the convergence of the sparse categorical cross entropy loss after 100 epochs.

```
Epoch 95/100
130/130 [==============================] - 1s 6ms/step - loss: 0.3458 - accuracy: 0.8507 - val_loss: 0.5486 - val_accuracy: 0.8039
Epoch 96/100
130/130 [==============================] - 1s 5ms/step - loss: 0.3348 - accuracy: 0.8583 - val_loss: 0.5287 - val_accuracy: 0.7826
Epoch 97/100
130/130 [==============================] - 1s 10ms/step - loss: 0.3166 - accuracy: 0.8674 - val_loss: 0.5148 - val_accuracy: 0.8092
Epoch 98/100
130/130 [==============================] - 1s 8ms/step - loss: 0.3560 - accuracy: 0.8520 - val_loss: 0.5364 - val_accuracy: 0.7870
Epoch 99/100
130/130 [==============================] - 1s 7ms/step - loss: 0.3265 - accuracy: 0.8619 - val_loss: 0.5188 - val_accuracy: 0.8135
Epoch 100/100
130/130 [==============================] - 1s 7ms/step - loss: 0.3494 - accuracy: 0.8553 - val_loss: 0.4872 - val_accuracy: 0.8111
65/65 [==============================] - 0s 2ms/step - loss: 0.4872 - accuracy: 0.8111
Test Accuracy: 0.8111110925674438
```

Figure 7: Training phase for Neural Network

The training and test loss calculated after each epoch is plotted and is shown in Fig.7. The convergence of the loss function was observed after 100 epochs and the test data accuracy achieved was 81 percent.

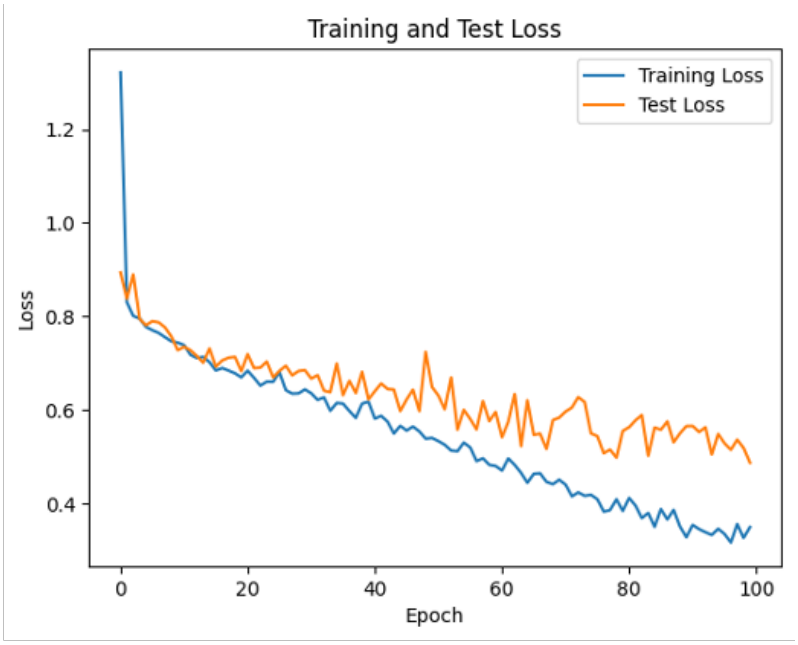
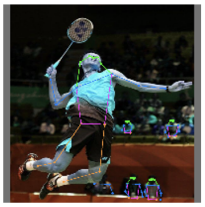


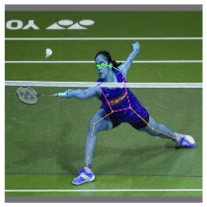
Figure 8: Training and test loss

The model was tested by obtaining random images from google, passing through yolov7 pose estimation model and feeding it to the neural network. The model was able to predict some of the images whereas it also failed to classify some of the images. Fig.9 shows the performance of the neural network model

Images correctly classified:



OP: Smash
GT: Smash



OP: netshot
GT: netshot

OP: lift
GT: lift

Images wrongly classified:

OP: lift
GT: Smash

OP: Smash
GT: lift

OP: Smash
GT: netshot

Figure 9: Neural network performance

VGG16 model was fine-tuned for 20 epochs and achieved a validation accuracy of about 58 percent. Since the images from google would not be from the same distribution as the dataset, the model was not tested on random images of players. Instead, the model was tested using the testdata available in the shot type detection dataset and it achieved an accuracy of about 66 percent.

```
Epoch [1/20], Loss: 1.5069
Epoch [2/20], Loss: 1.2893
Epoch [3/20], Loss: 1.0572
Epoch [4/20], Loss: 0.8700
Epoch [5/20], Loss: 0.5516
Epoch [6/20], Loss: 0.3034
Epoch [7/20], Loss: 0.1252
Epoch [8/20], Loss: 0.1407
Epoch [9/20], Loss: 0.0818
Epoch [10/20], Loss: 0.1003
Epoch [11/20], Loss: 0.0420
Epoch [12/20], Loss: 0.0217
Epoch [13/20], Loss: 0.0456
Epoch [14/20], Loss: 0.1535
Epoch [15/20], Loss: 0.0422
Epoch [16/20], Loss: 0.0386
Epoch [17/20], Loss: 0.0577
Epoch [18/20], Loss: 0.0347
Epoch [19/20], Loss: 0.0082
Epoch [20/20], Loss: 0.0021
Accuracy on test set: 0.5806
```

Figure 10: VGG16 Finetuning

Fig.10 shows the Fine-tuning phase of VGG16 model and Fig.11 shows the Testing phase of the pre-trained model.

```
Predicted class: 0       Ground Truth: 0
Predicted class: 0       Ground Truth: 0
Predicted class: 2       Ground Truth: 0
Predicted class: 0       Ground Truth: 0
Predicted class: 0       Ground Truth: 0
Predicted class: 0       Ground Truth: 0
Predicted class: 1       Ground Truth: 1
Predicted class: 1       Ground Truth: 1
Predicted class: 4       Ground Truth: 1
Predicted class: 1       Ground Truth: 1
Predicted class: 1       Ground Truth: 1
Predicted class: 1       Ground Truth: 1
Predicted class: 0       Ground Truth: 1
Predicted class: 0       Ground Truth: 2
Predicted class: 0       Ground Truth: 2
Predicted class: 2       Ground Truth: 2
Predicted class: 4       Ground Truth: 3
Predicted class: 4       Ground Truth: 4
0.6666666666666666
```

Figure 11: VGG16 Testing phase

8 Conclusion and Future work

The Neural network got a test accuracy of 81 percent by training on numerical dataset containing pose coordinates from yolov7. The VGG16 model got test accuracy of 66 percent after fine-tuning on an image dataset. The pre-trained model was expected to perform better but due to insufficient and noisy training data, the model had limitations. If there was a dedicated image dataset properly annotated, then the VGG16 model is expected to perform better. The Neural network model performed better than expected and it proved that the approach of using pose estimation followed by classification could efficiently categorize player's strokes into smash, drop and netshot. Fig.2 shows the final results of the models used for classification.

Dataset	Model	Test Accuracy
Motion Data for Badminton Shots	Neural Network	81 percent
Shot Type Detection Dataset	VGG16	66 percent

Table 2: Analysis

The system still fails to correctly classify around 30 percent of the images fed into it and it classifies most of the strokes as smashes when the ground truth is either a netshot or a drop. This is a limitation of the dataset which is biased with values from smashes. Furthermore, the model is not able to classify the stroke, when the player is executing a fake shot. In this scenario, the pose might suggest a smash, when in reality, the player might be performing a drop shot. More study in this direction is required to enhance the model so that it can identify fake strokes as well.

A more comprehensive dataset has to be collected and annotated to help the model perform better. Pre-trained models such as VGG16 has potential to improve with better datasets and more training data. The idea of classification of badminton strokes can be extended to real time classification of strokes played by players during a live match.

This system will be useful to train budding badminton professionals by helping them to understand and predict what stroke will be played by their opponent when they are moving at a certain pose. Once they know their opponent's stroke, it will be easier to counter that. Moreover, badminton being the fastest racquet sport, can certainly benefit from having systems that can classify strokes made by professionals.

References

[1] Yong En Tan et al. "A deep learning based framework for badminton rally outcome prediction". In: *2022 International Symposium on Intelligent Signal Processing and Communication Systems, ISPACS 2022*. 2022. DOI: 10.1109/ISPACS57703.2022.10082764.

[2] Kokum Weeratunga, Anuja Dharmaratne, and Khoo Boon How. "Application of Computer Vision and Vector Space Model for Tactical Movement Classification in Badminton". In: *IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*. Vol. 2017-July. 2017. DOI: 10.1109/CVPRW.2017.22.

[3] Wei Ta Chu and Samuel Situmeang. "Badminton video analysis based on spatiotemporal and stroke features". In: *ICMR 2017 - Proceedings of the 2017 ACM International Conference on Multimedia Retrieval*. 2017. DOI: 10.1145/3078971.3079032.

[4] Anurag Ghosh, Suriya Singh, and C. V. Jawahar. "Towards Structured Analysis of Broadcast Badminton Videos". In: *Proceedings - 2018 IEEE Winter Conference on Applications of Computer Vision, WACV 2018*. Vol. 2018-January. 2018. DOI: 10.1109/WACV.2018.00039.

[5] Nuttachot Promrit and Sajjaporn Waijanya. "Model for Practice Badminton Basic Skills by using Motion Posture Detection from Video Posture Embedding and One-Shot Learning Technique". In: *ACM International Conference Proceeding Series*. 2019. DOI: 10.1145/3375959.3375981.

[6] *Motion Data for Badminton Shots*. accessed on 2024-03-26. URL: https://www.kaggle.com/datasets/dylanyves/motion-data-for-badminton-shots/.

[7] *Shot-Type Detection Image Dataset*. accessed on 2024-03-26. URL: https://universe.roboflow.com/badminton-project/shot-type-detection.

[8] *Python OpenCV - Pose Estimation - GeeksforGeeks*. accessed on 2024-03-26. URL: https://www.geeksforgeeks.org/python-opencv-pose-estimation/.

[9] *Top 9 Pose Estimation Models of 2022 | by Ritesh Kanjee | Augmented Startups | Medium*. accessed on 2024-03-26. URL: https://medium.com/augmented-startups/top-9-pose-estimation-models-of-2022-70d00b11db43.

[10] *Transfer learning and fine tuning Model using VGG 16 | by Roshan Kumar Gupta | Medium*. accessed on 2024-03-26. URL: https://medium.com/@roshankg96/transfer-learning-and-fine-tuning-model-using-vgg-16-90b5401e1ebd.