# RUGBY EVENT DETECTION IN BROADCAST VIDEOS BASED ON VISUAL FEATURES USING DEEP LEARNING

Dulan Priyanga Jayasuriya

209337M

Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering
Faculty of Engineering

University of Moratuwa
Sri Lanka

May 2022

# RUGBY EVENT DETECTION IN BROADCAST VIDEOS BASED ON VISUAL FEATURES USING DEEP LEARNING

Dulan Priyanga Jayasuriya
209337M

Thesis/Dissertation submitted in partial fulfilment of the requirements of the degree
Master of Science in Computer Science and Engineering

Department of Computer Science and Engineering
Faculty of Engineering

University of Moratuwa
Sri Lanka

May 2022

# Declaration

I declare that this is my own work and this thesis/dissertation2 does not incorporate without acknowledgement any material previously submitted for a Degree or Diploma in any other University or institute of higher learning and to the best of my knowledge and belief it does not contain any material previously published or written by another person except where the acknowledgement is made in the text.

Also, I hereby grant to University of Moratuwa the non-exclusive right to reproduce and distribute my thesis/dissertation, in whole or in part in print, electronic or other medium. I retain the right to use this content in whole or part in future works (such as articles or books).

Student Name: **D. P. Jayasuriya**

Index Number: **209337M**

Signature: Date:

The above candidate has carried out research for the Masters dissertation under my supervision.

Certified by;

Supervisor Name: **Dr. S. Ahangama (B.Sc, PhD)**

Signature: Date:

# Dedication

This thesis is dedicated to my parents.

For their endless love, support and encouragement.

# Acknowledgements

I would like to express my deep sense of gratitude and a profound feeling of admiration to all those who helped me and encouraged me in any aspects to accomplish this project. My special thanks to the University of Moratuwa for giving me an opportunity to carry out this project.

I would like to take this opportunity to thank Dr. Sapumal Ahangama, Supervisor of this project for his guidance throughout the project. I extend my thanks to the external supervisor of my project, Dr. Chitraka Wickramarachchi for his guidance and continuous support throughout the whole duration of the project. I am really thankful to Mr. Hansa Perera for sharing the experiences and expert knowledge with the project matters.

Further, I would like to thank all my colleagues for their help in finding relevant research material, sharing knowledge and experience and for their encouragement. Finally, yet importantly, I thank all those who like to remain anonymous although the help they provided me was valuable.

Thank you.

# Abstract

A sports play event is an athletic activity that is performed by multiple players during a sporting event. Sports Event Detection is a challenging task in the domain of sports video analytics. Numerous attempts were made to detect events occurring in sports such as soccer, basketball, and cricket. Our primary objective in this research is to detect events in a Rugby sports video. In comparison to other sports, this one is more difficult due to the sport's chaotic nature. As a result, very little research is conducted on the Rugby sport. The Rugby Events Dataset is presented in this paper as a benchmark dataset for event detection in rugby. It contains videos with temporal annotations for events as well as images with bounding box annotations for the same. Nevertheless, using deep learning and computer vision techniques, this research was able to successfully train on this dataset and detect rugby events as well as temporally localize those events in broadcasted videos. A simple classification model is used to distinguish between sports fields and other scenes in these videos, while an object detection model is used to identify sporting events. Whereas current object detection models are used to detect objects, this research demonstrates that these models can be extended to detect sports events and still produce satisfactory results. Combining tracking with object detection models increased our accuracy of localizing events in the temporal domain even further. This project has released a Sports Event Detection Framework which can be deployed in any machine. The RugbyEvents dataset is publicly available in https://github.com/CodeProcessor/rugby-events-dataset and the event detection framework is available at https://github.com/CodeProcessor/sports-events-detection.

*Keywords : Sports Event Detection, Deep Learning, Broadcast Sports Videos, SriLankan Rugby*

# Table of Contents

# List of Figures

# List of Tables

# List of Abbreviations

| Abbreviation | Description |
|---|---|
| ANN | Artificial Neural Network |
| BOW | Bag of Words |
| CA | Classification Accuracy |
| CNN | Convolutional Neural Network |
| CSV | Comma-Separated Values |
| DBN | Dynamic Bayesian Network |
| DCNN | Deep Convolutional Neural Networks |
| DNN | Deep Neural Networks |
| ED | Euclidean Distance |
| EM | Expectation-Maximization |
| FFT | Fast Fourier Transform |
| GP | Genetic programming |
| GT | Ground Truth |
| HMM | Hidden Markov Model |
| HPE | Human Pose Estimation |
| JSON | JavaScript Object Notation |
| KNN | K-Nearest Neighbor |
| LDA | Linear Discriminant Analysis |
| LPC | Linear Prediction Coefficients |
| LPCC | Linear Prediction Coefficients |
| LSTM | Long Short-tern Memory |
| LTC | Long-term temporal convolutions |
| MFCC | Mel-Frequency Cepstral Coefficients |
| MLP | Multi-layer Perceptron |
| MSE | Mean Squared Error |
| PB | Part-Based |
| PS | Pictorial Structures |
| RF | Random Forest |
| STE | Short Time Energy |
| SVM | Support Vector Machine |
| VAE | Variational Autoencoders |
| ZCR | Zero-Crossing Rate |

# Chapter 1

# INTRODUCTION

This project aims to recognize different sports events in a broadcasted rugby sports video by analysing the visual features of videos using deep learning-based image classification and object detection techniques. A framework was designed to analyze the broadcast sports videos by identifying the main sports events in Rugby such as Lineouts, Scrums, and Rucks with the temporal position of each event. Nevertheless, this project contributes a novel event detection dataset to the community to improve the overall sports domain using computer vision techniques. This document provides comprehensive information about the different methods that have been tried and the primary method that is used to classify events from the raw broadcast videos.

## 1.1 Motivation

In the 21st century with Artificial Intelligence and machine learning technologies in the field of sports data analytics have the potential to deliver competitive advantage over the competition. Today, sports analytics data science can be considered as a viable product. Professional sports organisations are trying to go an extra mile with this precision information which will enable their athletes to perform the best action at the right time and make better decisions on the field faster than the competition. These analytics are made easier day by day with the development of computer vision applications in general.

A human action can be defined as a particular movement of the human joints. A sporting event is an athletic activity performed by one or more people in accordance with a set sequence or pattern of actions. In this context, event recognition and event localization are two critical and well-researched areas. Event recognition categorizes video segments based on predefined event classes, whereas event localization determines an event's spatial and temporal location. A sports game can be thought of as a series of actions that take place in order to achieve a specific goal.

Rugby sport is the main focus of this study, which is known as a popular full-contact team sport in which each side consists of fifteen players. It is based on running while holding the ball. The main goal is to take the ball with the help of the team to the opponent's territory placed behind the touchline, but the ball pass must travel parallel or towards the passer's own goal line.

There are few research publications available based on video analysis for Rugby

sports as it has become a more complicated sport compared to others. Further, the players are occluded from each other, which drives away from research interests. The lineout, scrum, ruck, penalty kick, and goal are the main events of rugby sport. The motivation of this research is to bring the sports analysis system to Rugby sports to uplift the game as a whole.

## 1.2 Problem Statement

Sports are more than enjoyable and entertainment for the masses. Athletes and coaches spend many hours per week training, traveling, and competing. Today, youth sports have become more competitive every day. To keep up with the training and prepare for the opponents' tactics, coaches and players extensively analyze sports videos and obtain valuable information. This information is vital to training a beginner player, identifying the weaknesses, and strengths, and also better preparing for the opponents by identifying their tactics and strategies. Today, in the information age, there are thousands of videos recorded and stored on digital media, which can be overwhelming to check one after another. Coaches and performance staff don't have time to sit in front of a computer to figure out the game pattern of the opposition played in early games or find the best sports play events that will help the players learn from those.

However, people are tagging specific sports events, which are considered to be the best in the videos, to check out the specific events later without watching the whole video. The task of video annotation is a time-consuming task; it requires watching the whole video and marking the time where the action or specific event takes place. This should be carried out by an experienced person who is an expert in this game. Besides, with the growth of the sports videos, it has become an extensive task.

In Spite of the video annotations, there are a number of advantages of what videos do well for athletes. This can be used by the coches to guide the athletes and discuss the view or the perspective of the video which enables both parties to be on the same visual page together. Discover opponents strengths and weaknesses and prepare to expect or exploit those before the match. These videos teach new movement strategies and how to perform those effectively in the field. The above mentioned advantages are not a complete list, but these points will definitely close the gap or widen the margin of the victory. Analysing these videos are extremely helpful for the athletes to improve their game. This analysis will be even effective if the video is annotated for specific sports events, then the interested parties can check only the relevant information on the video.

The aim is to implement a framework where the event detection is done using high-level feature extraction combined with deep learning methods to classify the events. When a video is provided to the system, the system will detect the players, objects, and their movements and then analyze the video. After summarising the time segments the system will output an event for a specific time frame which is similar to manual human annotation. Players can improve their movements by watching these events occur in the sport. Human intervention is no longer required for the video annotation process as it is fully automated. Finally, using these annotations a library of videos can be created like an encyclopedia where players or coaches can search and extract specific types of

plays in sports within seeking through the entire video.

## 1.3 Aims & Objectives

As a remedy for the aforementioned problems, this work provides a way to easily detect and categorize sports events from a broadcast video. This helps the coaches, athletes, and game observers to quickly jump to the exact positions of the event in the video. These categorized events can be used to train beginners and also be useful to analyze opponent teams' strengths and weaknesses. Event detection is also applicable for video summarization in which someone can easily create a highlight video using the information provided by our system. Also, this information can be useful for someone to detect these events in real-time such that they can use the less important video segments to replace those scenes with advertisements.

The objective of this research is to accurately identify different events in a Rugby sports video using only the visual features. The main research objectives are listed as follows;

- Create and publish a dataset of Rugby sports events.
- Identify sports activities in a broadcast video.
- Categorize the detected sporting events.
- Temporal localisation of Rugby sporting events.

First, there are a few datasets available for sports event detection. The first objective covers that. This dataset will also help to train a better model that can identify events in a video. The second objective is to detect the play and break events of the sports game, or else that activity area of the game. Then the third and fourth objectives are the primary objectives of detecting and localizing the events throughout the video. Overall, this research's ultimate objective is to give better sports insights to the players using sports video analysis. This can be further extended and be useful to train beginner players, correct mistakes, and summarise sports games in the future.

## 1.4 Scope

In the new era of information technology, almost all information is now within the reach of the tip of the finger. Now almost all the important sports information and footage are recorded digitally. Most of these videos are freely available in online video repositories. Furthermore, the best sports contests are broadcast to all the people around the world, and the archives grow exponentially. However, these videos are just low-level raw data. There is some high-level information about the videos, such as the date and the teams that played in the match. Each video consists of a significant number of play events that derive important semantic information. Hence, it is crucial to find a way to analyze that embedded information and annotate the specific events,

that would be beneficial for all sports enthusiasts. The main objective of this project is to predict sports events in a given broadcast video.

Rugby is a contact game that is played between two teams each consisting of fifteen players on the field. In the game of Rugby, there are many events happening inside a game. Some of these events are Lineouts, Scrums, Maul, Kick, Goal, and Ruck. In this research, we are mainly focusing on three event types. namely,

1. Lineouts [A.1]
2. Scrums [A.2]
3. Rucks [A.3]

The above-mentioned events are considered as the most frequent events in the entire Rugby game, and those are clearly distinguishable from other events. The main intention is to develop a framework that is capable of detecting those specific events in a broadcasting rugby video.

The remaining chapters are outlined with a detailed description of the model and other techniques that are used to extract events from a video. Starting from chapter two, which mainly provides a comprehensive review of the relevant literature, this section explains other similar systems and the technologies available to achieve good accuracy in this area of sports event detection. Chapter three presents the design of the solution, experiments conducted, and implementation details. One of the main contributions of this research is the event detection dataset. There are no sports event datasets available for Rugby at the moment of publishing this. It was decided to release the dataset to the public, which was used to train all the models in this project, so anyone can replicate the results of this paper. Chapter four is dedicated to the results and evaluation. The primary debate revolves around the accuracy values obtained and how well they reflect the framework's overall performance. Finally, the results and conclusion section summarise all the work done for this project and give an overview of the entire project.

# Chapter 2

# BACKGROUND

## 2.1 Literature Review

### 2.1.1 Sports Event Detection Systems

The term ”sporting event” has several meanings. In this study, a rugby sport event is defined as a specific type of event that occurs during the game, such as a lineout, scrum, or kick. These various events are generated by players collaborating on specific actions. Research has been done on how to detect sporting events. The literature on sports event detection can be divided into three categories: video-based, audio-based, and both audio-video-based. The video is the primary focus of this literature review. It is nearly impossible to categorize events using only audio. Video enhances the robustness and accuracy of event detection. Because the primary goal of this study is to classify events, we chose a video-based approach. The section that follows contains a literature review of existing event detection systems in various sports.

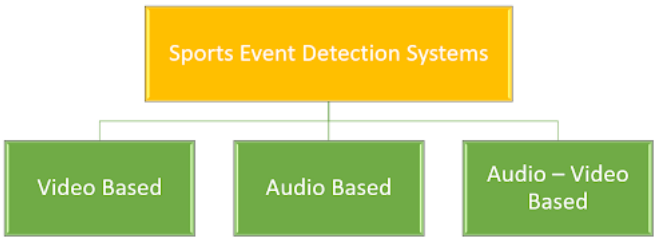



Figure 2.1: Existing sports event detection systems

In comparison to research on sports video analysis based on visual data, very little work has been conducted on sports video analysis based on audio data. Li Lu et al [20] has proposed a Support Vector Machine (SVM) based method to detect events using the audio clues. Here the paper has used a simple sliding window approach to segment the entire audio clip before extracting features. After extracting features a

SVM was used to classify the features into different categories with some rules. They have achieved an average F1 score above 79% for 8 hours of television program.

Another audio based event detection method is introduced by Mark Baillie et al [1] using broadcast video data. They have identified the main benefit of this method as the ability to detect high levels of crowd responses which is highly correlated to key events. They have categorised the audio to 6 pattern classes. These audio-based pattern classes were modelled with a continuous density Hidden Markov Model (HMM) and the likelihood was calculated using 'Viterbi's' decoding algorithm. They have defined an event window as when the crowd response triggered for longer than t seconds. A correctly detected event was assumed to be an 'event window' that overlapped the location of a true event. This research is mainly focused on detecting sporting events but they are not classifying those events into classes.

But Min Xu et al [37] proposed an event detection framework called audio keywords. This is a middle level representation that will fill the gap between high level and low level audio features. The low level audio features were created using SVM learning. The classifier initially classifies three classes using the audio clues. Then some subclasses were used as audio keywords, mainly some whiste patterns and commentator speech patterns. Combining these they were able to identify five events, goal, shot, foul, kick and game start/end. They have evaluated their method in 3 matches on FIFA world cup 2002 and obtained close to 60% of detection rate for five event classes. Web-casting Text was extracted to improve the event detection accuracy by Changsheng Xu et al [35]. This is also indirectly taking audio to help the event detection. They have detected keywords in web text and analysed context information before and after then according to some rules events were predicted. The video was used only to detect the start time of the video and map the event time to the video time. They have achieved nearly 100% accuracy for well-structured web text and around 90% accuracy for freestyle web text. They have effectively combined low-level features (video) with high level semantics(web cast text). While each of these audio-based event recognition systems produces acceptable results, their limited sound patterns are incapable of capturing all of the important events that occur in sports videos. Most of the events are detected as play events, without giving out much information about what exactly is happening in the video footage. Additionally, the accuracy of event detection is dependent on the performance of audio keyword recognition. These detections are highly volatile due to the nature of the audio captured from the events. Thus, relying solely on audio sources is insufficient to complete the task of event recognition. To improve the robustness and accuracy of event detection, we must enlist the assistance of video analysis. Which is exactly why it was useful to consider visual based features for event detection and classification. In the below section we mainly focus literature on the visual based event detection systems. Baoxin Li et al [15] proposed a general framework for event detection and video summarization. They consider a most basic segment where important action occurs as a play event. They have implemented both deterministic and probabilistic approaches for the detection of the plays. A rule based inference and a four state Hidden Markov Model (HMM) model is used for the aforementioned approaches. Also they have conducted three case studies for Baseball, American Football and Japanese Sumo Wrestling and achieved good results. HMM

was again used by Gu Xu et al [36] for a semantic analysis framework for sports event detection. They interpreted a sporting event as a rule-based sequential signal and justified their use of HMMs. They proposed Sports Game HMMs, a framework for sports game semantic analysis based on generic multi-layer HMMs (SG-HMMs). Each event is represented by a pre-trained HMM, whereas the rules are represented by HMMs composed of many layers. They evaluated the applications' performance in basketball and volleyball games, achieving a user satisfaction rate of 75% and 76.4 percent, respectively. Jinjun Wang et al [32] presented a novel event detection framework capable of detecting sports highlights from single camera video utilizing visual and audio data derived from the video. This framework has mainly three levels, first level it divides the video into video and audio streams. In the middle level the video stream was analyzed using three methods Position, ball trajectory and motion activity while audio stream was analyzed by keyword creation. Finally in the high level phase all are synchronized and event detection is obtained using Support Vector Machine (SVM). They have trained and tested the system for three events namely Goal, Foul and Other, using 2002 FIFA world cup soccer videos and obtained around 75% of accuracy for events. Event detection by multiple modalities was tried by A.B. Salunke et al [24], this method introduced a novel way to generate audio keywords by combining them with semantic basketball video shots and then detecting interesting events using domain-specific knowledge. The audio keywords generated can be utilised to recognize semantic events in sports videos using Hidden Markov Model (HMM) learning with video shots. Soccer event detection was researched by Abdullah Khan et al [12], and they have proposed object detection to extract information and spatial relations between objects to detect events. Single Shot Multibox Detector (SSD) model was used to detect the bounding boxes and confidence of the people and objects. They have used filtering methods to find out and remove noisy objects and then a distance-based rule engine to determine the events. That paper divides the events into simple events and complex events, which a complex event composes of a sequence of simple events. For a 5 min video they have detected 13 out of 14 ball possessions and 16 out of 19 kicking the ball events. They have shown the effectiveness of simple distance based measures on event detection. Hossam M. Zawbaa et al [38] used a Machine learning approach to predict events in a soccer video in order to get a summary of the play. They first segment the video into small video shots. Then it classified these video clips into different shot-type classes. Additionally, they utilised Support Vector Machines (SVM) and Artificial Neural Networks (ANN) to highlight key areas with logos and to detect the caption region carrying game score information. This was evaluated using videos from five international soccer championships, and the proposed system performs impressively, with high accuracy in its analysis results. Another soccer event detection was performed by Samuel F et al [4] using machine learning techniques. Field lines, goal mouth and ball were detected using image processing techniques. Then tracking was applied to the ball and the field players. They have supported four events, namely Touching, Passing, Ball possession and play break structure. They applied SVM to recognize the team of each player. Using rules they have defined each of the events and how those are related to the objects. Using these rules the events are classified into the mentioned categories. Their main focus is on semantic event detection due to the

nature that it can produce high level results. They have analysed many approaches to the event detection problem but the paper is all about information and possible paths to achieve the results. They have archived satisfying results for sports video event detection using audio keywords. Soccer goal event detection system was developed by Grigorios Tsagkatakis et al [30] with the use of deep learning. They have created a dataset that consists of goal and no-goal sequences. 20 frame moving windows initially select part of sequences of interest. Raw pixel values and optical flows are first independently encoded using the pre-trained deep CNN for extracting spatial and temporal features. Extracted features were fused together and used as extended input features for the classification. Deep Learning based event detection was proposed by Ali Karimi et al [11] mainly using variational autoencoders. Their solution consists of three stages, first they have used variational autoencoders to differentiate soccer images and then applied classification modules to classify images into different categories. They have divided events into nine classes and trained an image classification network using transfer learning. They have achieved 94% accuracy for event classification using efficientnetB0. If the event is a card event a fine grained image classification module is used to classify images of red and yellow cards, with an accuracy of 79.9%. The proposed method got 93.21% overall accuracy which is great.

### 2.1.2 Summary of sports event detection methods

There are many approaches to Sports Event Detection, some of the literature about sports event detection has been extracted and stated above. A more comprehensive comparison between those methods are added into a table as follows.

Table 2.1: Event Detection Papers - comparison

| Reference | Sport | Feature Type | Technology | Limitations |
|---|---|---|---|---|
| A SVM-based Audio Event Detection System [20] | NA | Audio | Sliding Window, SVM | Only cheering and applause events |
| Audio-based Event Detection for Sports Video [1] | Soccer | Audio | Hidden Markov Model | Two classes only event and non-event |
| Creating Audio Keywords for Event Detection in Soccer Videos [37] | Soccer | Audio | Audio keywords, Rule based, SVM | Difficult to detect sound transition |
| Live Sports Event Detection Based on Broadcast Video and Web-casting Text [35] | Soccer | Video, Web-casting Text | Keyword rule based events | Need web casing text |
| Continued on next page | | | | |

**Table 2.1 – continued from previous page**

| Reference | Sport | Feature Type | Technology | Limitations |
|---|---|---|---|---|
| Event Detection Based on Non-broadcast Sports Video [32] | Soccer | Audio, Video | Hidden Markov Model | Limited events, Goal, Foul and Other |
| Event Detection in Sports Video Game [24] | Basketball | Audio, Video | HMM, Dynamic programming, MFCC, ZCR, LPC | In-play and out-of-play detection only |
| Event Detection and Summarization in Sports Video [15] | Baseball, Soccer, Sumo Wrestling | Video | Bayesian networks, Rule-based inference, HMM | Detecting only play and non-play events |
| A HMM based semantic analysis framework for sports game event detection [36] | Basketball, VolleyBall | Video | Sports Game Hidden Markov Model | NA |
| Soccer event detection [12] | Soccer | Video | Object Detection, Distance Measures | Simple events, Ball possession and kicking |
| Event Detection Based Approach for Soccer Video Summarization Using Machine learning [38] | Soccer | Video | SVM, ANN, Hough Transform, Gabor Filters | Generic events such as Goal, Attack and Other |
| An Overview of Automatic Event Detection in Soccer Matches [4] | Soccer | Video | SVM, Ball trajectory | Only an overview to the problem, no real implementation |
| Goal!! Event detection in sports video [30] | Soccer | Video | CNN, Auto-encoders, raw pixel and optical flow encoding | Goal and No Goal only |

Continued on next page

**Table 2.1 – continued from previous page**

| Reference | Sport | Feature Type | Technology | Limitations |
|---|---|---|---|---|
| Soccer Event Detection Using Deep Learning Ali [11] | Soccer | Video | CNN | NA |

[20] [1] [37] and [35] are mainly based on audio but our main focus is on video because we need a more robust and accurate event detection. Also videos contain more information to predict different types of events which is harder to detect using audio based approaches. Video based event detections can be ported for other sports given a good annotated event dataset. [15] and [36] used HMM for the event classification task but with the advancement of deep learning there are better approaches to solve the classification problem. [12] shows a simple distance based measure to detect events, but this method couldn't capture more specific events since it is rule based. Sm [38] used many aspects into consideration including shot type, play or break event, replay detection, excitement detection and others to predict the events. They were able to classify the events into more generic classes such as Goal, Attack and Other events. [4] gives many ideas on Detection, Tracking and Event detection which can be used in our research. [30] used deep learning methods to extract features from continuous raw rgb frames and optical flow encoding then combined together to finally classify for the events. Also [11] proposed a method to train a simple CNN using thousands of event images extracted from soccer games and got good accuracy. These recent advancements in Deep Learning were really good, so this research also mainly focuses on deep learning techniques.

### 2.1.3 Challenges in Sports Event Detection

Based on the above literature review we identified many challenges related to sports event detection. Despite that, any video analysis system comes with many challenges. Scene change detection, camera movement compensation, occlusion, landmark detection and tracking are some of the main challenges in this context. These challenges are briefly discussed in this section.

Tracking is a technique that is frequently used in sports analysis. It has been used to track balls, players, and referees, among other things. It is critical in other analysis processes. If there are multiple cameras this is even more difficult since the information should be tracked across the cameras. In sports videos tracking should be applied to all the objects to identify the unique objects across frames. Since there is no mapping between frames for objects the tracking comes to play a major role in this area. Tracking will provide the information that needs to connect objects within frames. The camera motion, viewpoint change and occlusion are major limitations in tracking objects. Landmarks in sports video analysis include goal area, sideline, corners, ground areas. Detecting these landmarks are important since these landmarks can define areas such as the goal area or playground. Even these can be used to determine the rules in the

game. As an example, the 22 meter line will be an important landmark for the Rugby game. Another example would be determining the play area, whoever inside the play area could be considered as an active player, those who are outside the play area can be considered as the supporters or replacement players. So detecting these landmarks are necessary for sports video analysis. Object detection methods can be used to identify these areas properly. When interacting with broadcast sports videos, the camera is not in a still position. The view is changing with the scene. The camera can pan, tilt and zoom according to the scene. Also there can be multiple cameras shooting from multiple angles and the scene can be switched dynamically between those. Scene change detection is the process involving the detection of boundaries between uninterrupted segments. These camera movements and scene changes should not alter the final outcome of the event detection. Camera movement compensation system should be considered when processing these broadcast video clips. Soccer, Cricket, Volleyball or Rugby these outdoor team sports games have a dynamic nature of players moving around the play area. Their movement can occlude other players or important scenes which are called occlusion. This situation is worse when it comes to a contact sport like Rugby where most players contact each other creating huge obstructions to the camera view. Occlusion will cause the object detection to fail and reduce the accuracy of other operations that rely on that such as tracking. Also there can be other types of occlusions like advertisements, scoreboards and player cards which popup in the middle of the screen that need to be excluded before processing for event detection.

### 2.1.4 Event Detection Content Pyramid

In any video analysis system it's important to understand how the content is extracted from the raw video. In our Event detection system the content can be visualised as a content pyramid which consists of blocks. The content pyramid idea is used to analyse the context of a sports video entity. Each tier of the content hierarchy represents several critical components that contribute to the video's characterisation. The layer's volume indicates the number of implied notions. The compactness of the information decreases from top to bottom in this pyramid. Five layers of video content were created: raw video, activity clips, visual features, sports events, and a video summary.

The raw video layer serves as the foundation. It is composed of video clip frames, each of which contains raw video data. These videos were recorded live in parallel to the sports events and broadcast directly to the viewers via digital medium. Most of these live broadcasts are saved as videos for viewers to view later in video sharing sites. Our primary source of information is these broadcast streams saved as videos. There are many video sharing sites online such as YouTube, Vimeo, Twitch, Facebook watch and Daily Motion. The most popular and largest provider is YouTube which is provided by Google. These videos consist of both audio and video but our system is giving focus to the visual part of the video only.

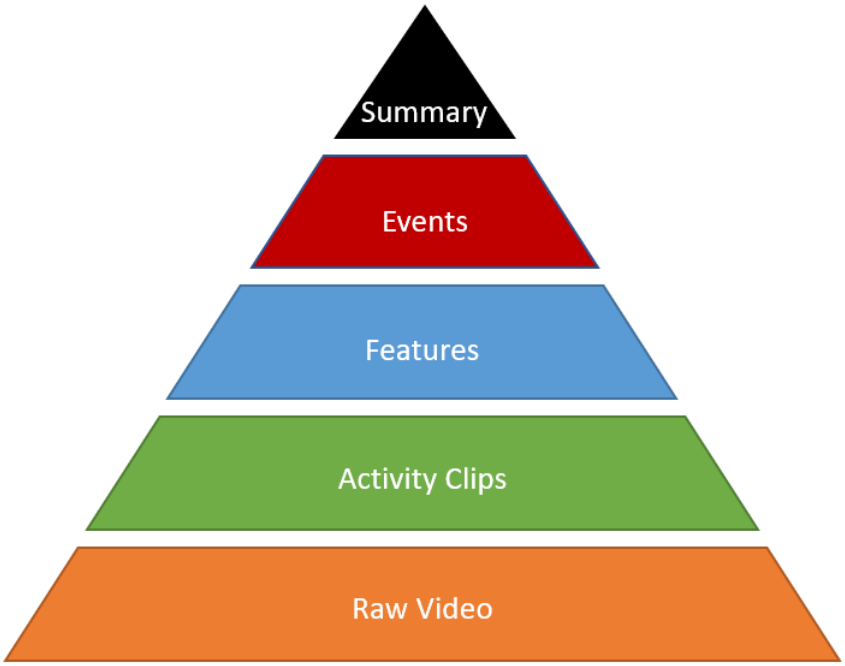



Figure 2.2: Event Detection Content Pyramid

Activity clips are the second layer where the raw videos were taken and processed to get the usable views which is generally the ground area with the players. Unwanted scenes such as advertisements, scoreboards, crowd scenes and slow motion replays have been removed in the process of converting the raw video to activity clips. Also the video transitions are detected and splitted since the transitions represent another scene of the game. These short clips are called activity clips since these video clips contain only the sports play in the playground area. This extraction is necessary since the analysis system only needs to focus on the play area; all other context can be considered as noise to the system.

In traditional machine learning there is a separate state called feature extraction. But with the improvement of the deep learning era, these feature extractions are already embedded in the deep neural networks. Due to this reason, the feature extraction part and sport event detection part can also be combined together. A sporting event is described as a video segment that depicts a group of participants performing an activity. These sporting events are referred to as Lineout, Scrum, and Kick in the context of Rugby.

The last layer which is summarization is the summary of the entire video which is presented to the user to get an understanding of the whole game. This is not addressed in this research and can be achieved after detection of the events with other information such as score, winning side, ball possession and other which is not covered in this research. Also sports highlights can also be extracted as a part of this layer, which has the most intensive moments of the match.

Event detection literature is already discussed yet the video information extraction can be broken down into sub levels as in figure 1. The proposed system follows the same stages when detecting events. Sports Activity Detection, Feature Extraction, Feature Learning and Event Detection are the main research areas in this context. There is literature that can be found under each section or layer of this content pyramid. In this section our main focus is to look into the literature which falls under aforementioned subsections.

### 2.1.5 Extracting activity clips

This section is mainly focused on past literature on how to extract the activity clips from the original broadcast video. Here we refer to activity clips as scenes where the actual game is played and recorded; non-activity clips as Unwanted scenes such as advertisements, scoreboards, crowd scenes and slow motion replays which should be removed from the prediction. These undesirable scenes can occur frequently and consume a significant amount of broadcast time in particular sports. This will save valuable computation power, reduce false positives and enable a hierarchical semantic analysis on sports events. Many researches have done to detect these activities and most of those research are called play break event detection, where they refer to play events as the time segments where the ball is in action and break events refers to the stoppage times of the events.

Lexing Xie et al [33]proposed a method to detect the play and break states of the game based on the rules of soccer. They have taken a fixed length sliding window approach to get the features and then classify the feature vector into either one of the Play or Break classes. This classification is performed with HMM followed by dynamic programming. They have achieved 83.5% accuracy over a diverse data set. Marc-Andr´e Carbonneau et al [2] presented a method to detect play and break detection on a video. This doesn't rely on the prediction cues which were added while broadcasting the video. The authors showed that this method runs in real time, without a tracking algorithm and can be extended to other sports as well. They used a two stage hierarchical method, in the first stage the event detectors are trained to detect key events such as preliminary play-breaks and line changes. The second stage, with the first stage output was also taken into account along with the novel feature based on spatio-temporal interest points to predict the final decision.On real-world hockey game footage, this approach attained a 90 percent accuracy rate. Another Play-break event detection in sports video was analysed by Ahmet Ekin et al [6] As discussed previously, sports video is made up of long, medium, and close-up shots. They've noticed that while play events are often long shots, medium shorts or player closeups will occasionally show during the play. To detect such instances, this study introduced a new parameter Tb, which represents the maximum duration allowed between two long shot play events. If the delay between two long shot plays is less than Tb, the entire segment is considered to constitute play. Another method to detect hard-cut video segments is proposed by Changsheng Xu et al [35]. They have computed the mean absolute difference (MAD) of successive frames and used an adaptive threshold to detect

sudden changes. They also proposed another method to handle short change shots with computing multiple pairwise MAD. False positives were observed for sudden camera movements and foreground occlusions. Replay detection was done by Hossam M. Zawbaa et al [38] using a logo detection model which often appeared at the start and end of a replay. This logo detection model was trained to identify a wide variety of logos used by different broadcasters. Score board detection model also developed to detect scoreboards. These two types of detection algorithms use Support Vector Machine and Artificial Neural Networks to classify the view.

Long, medium, and close-up shots based play event classification is [6] an effective way and could be slightly modified for our purpose. The above research is done for soccer and tennis which has huge differences compared to Rugby but there are some similarities in video recordings. When there is an event Rugby games would take that event as a close up shot. Taking Mean Absolute Difference MAD as mentioned by Changsheng Xu et al [35] is also a good approach but this could be more expensive and would error prone to sudden occlusions. Training a classifier purely based on images is a challenging task for rugby since the break events and play events in Rugby are close to each other compared to other sports. But these kinds of classification might be useful to detect the scoreboard or advertisements. Also for the playing field play break events as proposed by cite, we can go for a distance measurement using human pose features since pose is accurate and directly related to events.

### 2.1.6 Feature learning methods

Feature learning is a collection of approaches that enables a system to discover effective representations for feature categorization from raw data. These techniques can be classified into two categories: conventional approaches and deep learning techniques. Bayesian networks, dictionary learning, and Genetic programming are all examples of traditional approaches. These techniques were widely utilised before the invention of deep learning techniques.

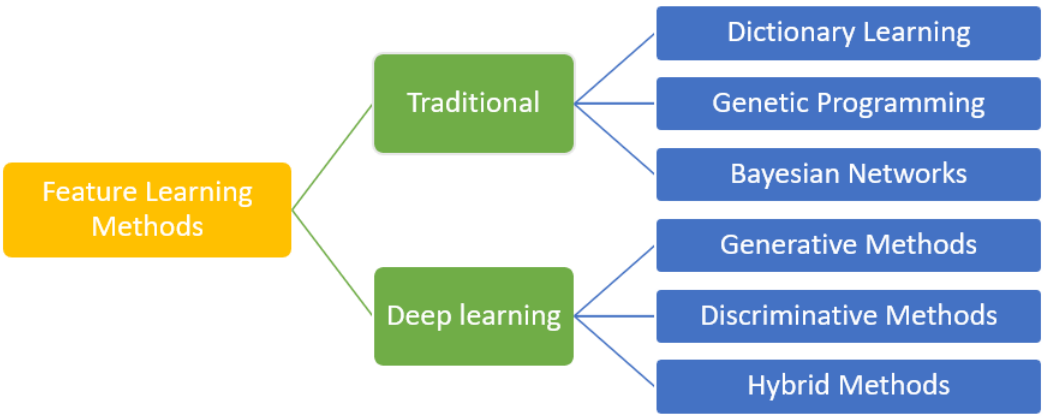



Figure 2.3: Feature Learning Methods

Dictionary learning attempts to refine a dictionary frame in which the training data admits a sparse representation. The better the dictionary, the more sparse the representation. Dictionary learning is classified into two types: unsupervised and supervised. The unsupervised method is used when the goal of the dictionary is to minimize the reconstruction error of the original samples. Supervised learning techniques can be used to create a category-specific dictionary that promotes class discrimination. Genetic programming (GP) is a technique for evolving programs in artificial intelligence. The primary operations in GP are selecting features for reproduction (crossover) and mutation based on predefined fitness measures, which are typically the proficiency of the desired task. Genetic programming has been used to generate holistic descriptions that enable action recognition tasks to be performed as efficiently as possible. Li Liu et al [18] applied this method to develop holistic descriptors that optimize performance on action recognition tests. A Bayesian network is also known as a belief network is a type of probabilistic graphical model. It uses a directed acyclic graph (DAG) to express a set of variables and their conditional dependencies. Li et al [16] propose a framework based on the Dynamic Bayesian Network (DBN) to systematically model the dynamic and semantic relationships among multilevel action unit intensities. Their primary objective is to identify facial expression analysis with the use of muscle motions of the face which are called action units.

The adobe methods are the traditional feature learning methods and most of those are outdated today. The new deep-learning era has emerged and the feature learning methods based on this area is widely explored. These types of methods need lots of data to train. These types of methods aim to learn multiple levels of representation and abstraction that allow a fully automated feature extraction process.

Generative methods are unsupervised techniques that can be used to depict any unlabeled data distribution. The new representation decreases the dimension of the data and conforms to its distribution. The primary objective of these generative models is to comprehend the data distribution, including the features associated with each class. The most commonly used approaches can be mentioned as Auto-encoder, Variational AutoEncoders [5], Generative Adversarial networks. In [26] the authors have used the LSTM Encoder-Decoder framework to learn video representations. They have tried different options, one is to recreate the same sequence as input and the other is to predict the future frames using all the past frame information. They have come up with the Composite Model that combined an autoencoder and a future predictor as their best model. In this paper [21] they have addressed the abnormality detection problem in crowded scenes using Generative Adversarial Nets (GANs). They have trained the GAN with the normal crowd data which are not able to generate abnormal events. In the testing phase, the local difference between the real and the generated images is used to detect possible abnormalities. Discriminative methods are supervised models that make use of a hierarchical learning model created of multiple hidden layers that are trained to classify features into distinct categories. Deep Neural Networks, Recurrent Neural Networks, and Convolutional Neural Networks are the most often utilized techniques under this method. There is much research work done using these techniques. An example [31] proposed a new way of learning video representations using

neural networks with long-term temporal convolutions (LTC). They have achieved the state of the art performance on two action recognition datasets UCF101 and HMDB51 using space-time convolutions over a large number of video frames. For better results, hybrid models combine both generative and discriminative models into a single model.

There are many feature extraction methods available. But deep learning gives promising results with the advancement of Computer Vision. In this research, the focus is to use a deep learning-based method to identify sporting events.

## 2.2 Image classification

Image classification is a machine learning task where the computer tries to analyse an image and decide what actually that image contains. For example, if there are two sets of images, cats and dogs, image classification is the process of the computer checking each image and applying a label into those images as cat or dog. This task is a pretty easy task for a human being. But for a computer this is a very difficult task.

A digital image is a collection of pixels with different pixel values. Computer sees an image as a collection of pixels. When checking these raw data, a small change in the environment will generate completely different raw pixel values. Example if the same cat appears in two different images, the raw pixel values will be completely different such that there is such that no connection exists at all. Two images which have the same object/animal can have different angles, poses, backgrounds, lighting conditions, etc. This alteration made it extremely difficult for a computer to correctly recognize the image's true content.

Deep learning is a subset of artificial intelligence (AI) which aims to mimic the human brain. The smallest unit of these networks are made of a module called neuron, these neurons combine together to form different types of networks. These learning algorithms try to teach these neurons to work together to predict these complex tasks with the help of large amounts of data. Deep learning in computer vision allows machines to extract the features in the images. The computer can analyse thousands of these images and train these small neurons on those images to interpret the image as a whole. These algorithms are designed in such a way that the programmers don't need to add rules by hand to extract those fine grained features.

The history of CNNs goes back to the 19th century. LeCun et al [14]. created the first substantial evolution of the CNN architecture in 1998. The architecture was named as LeNet-5 and it is one of the most basic architectures of all time. It has only 2 convolution layers and 3 fully connected layers. This architecture is one of the earliest CNN networks. It has become the de facto standard template for CNNs, with approximately 60,000 parameters. This is accomplished by stacking convolutions layers on top of each other. This paved the way for the well known CNN layer stacking. LeNet has an activation function, pooling layers after convolution layers and finishing the network with one or more fully connected layers.

Alex Krizhevsky et al [13] constructed AlexNet in 2012 by stacking a couple of more layers onto Lenet-5 architecture. AlexNet was one of the largest convolutional neural networks ever constructed on the imagenet subset. Rectified Linear Units (ReLUs) were used for the first time in this architecture. Also they have contested for the

ImageNet challenge and secured the first place with 37 percent and 17 percent for top-1 and top-5 error rates, respectively, which are obviously superior to the earlier state-of-the-art. Additionally the authors have used dropout as the regularisation method for the deep learning model. This achievement created a new chapter for deep learning based image classification architectures. All the other architectures are inspired by this finding.

CNNs are now becoming increasingly complex. This is due to the fact that increasing the size of a deep neural network is the most basic approach to improve its performance. The influence of convolutional network depth on accuracy in large-scale image recognition was examined by the authors. VGG network [25] primary contribution is a detailed evaluation of networks with increasing depth utilising a convolution filter architecture with extremely small (3x3) convolution filters. These findings formed the basis of their application to the ImageNet Challenge 2014, where they came in first and second place, respectively, in the location and classification tasks.

Building neural networks using blocks or modules rather than convolutional layers was one of the key concepts behind the Inception-v1 [28] design with 5M parameters. These Inception modules are used to implement the Network In Network. An Inception module's architecture is the product of research into sparse structure approximation. The inception module captures different features at 1×1, 3×3 and 5×5 and then clusters them together. Additionally, they employed 1x1 convolutions for dimension reduction in order to mitigate processing bottlenecks.

Inception-v3 is a 24M-parameter successor to Inception-v1. Inception-v2 is an early prototype of Inception-v3, and so is quite similar to Inception-v3 [29] but is not widely used. Inception-v3 is the network that incorporates some tweaks which were tested during the implementation of inception v2, some of the tweaks include modifying the optimiser, loss function, and adding batch normalisation to the auxiliary network's auxiliary layers.These authors were the first designers to use batch normalisation.

Most of the networks try to go deeper with convolution layers. However, as network depth increases, accuracy becomes saturated and subsequently rapidly declines. This problem is exposed as the degradation problem. The authors of Deep Residual Learning for Image Recognition [10] solved the degradation issue by developing a framework for deep residual learning. Rather than assuming that each of the few stacked layers fits the required underlying mapping perfectly, we explicitly allow these layers to fit a residual mapping. This novel implementation eases the training of networks that are substantially deeper than those used previously.

Xception [3] is a modification from Inception. The Inception modules have been replaced by depth wise separable convolutions in this design. Additionally, it has a similar number of parameters to Inception-v1. The authors have also demonstrated that the improvements in performance as compared to the beginning of the study are not related to greater capacity, but rather to a more efficient use of model parameters.

The scientists at Google have achieved it again with Inception-v4 [27], which has 43M parameters. Once again, this is a step forward from Inception-v3. The key distinction is the Stem group, which has undergone some minor modifications in the Inception-C module. Additionally, the same authors introduced Inception-ResNets,

which are a subfamily of Inception-ResNet-v1 and Inception-ResNet-v2 that are based on Inception-ResNets. The main improvement was the conversion of inception modules to residual inception blocks and adding more of those modules. As demonstrated by the authors' extensive empirical evidence, training Inception networks with residual connections considerably accelerates the training process.

Resnext [34] is introduced as an improvement to the existing ResNet architecture. This method reveals a new dimension, called "cardinality." Cardinality is the size of the set of transformations to be aggregated, which expands a new dimension. This is a critical factor in addition to the depth and width dimensions. Moreover, the authors have proven that increasing cardinality is more effective than going deeper or wider with increased model capacity. The authors were able to show that this can get better results than its ResNet counterparts.

CNNs are always developed within a fixed resource budget. Typically the scientists increase these computational resources for a better accurate model. The increase in network depath, width and resolution can lead to better performance. Efficient net paper takes this scaling into consideration and comes up with a simple yet highly effective compound coefficient. This scaling can be used to scale up the network in the most efficient way with respect to the computation budget. Using this scale they have come up with a family of models called EfficentNets. These networks achieve state of the art accuracy for many datasets including imagenet with 9.6x fewer parameters on average.

## 2.3 Object Detection

Object detection combines the tasks of object classification and localization. Classification is the process of identifying what is in the image, where localization answers the question of where the object is. The combination of these will yield to detection of multiple objects within the same image with the location information of each object.

Two decades ago, during the traditional object detection era, most early object detection algorithms were built using handcrafted features due to the lack of effective image representation at the time. There are a couple of good examples for object detection.

Paul Viola and Michael Jones created the Viols Johns Detector in 2001, which is capable of detecting human faces using a sliding window method. This sliding window searches for Haar-like features to detect different objects. N. Dalal and B. Triggs proposed Hog as an improvement to the scale-invariant feature transform in 2005. Here the basic concept is to get the direction and magnitude of gradients of the image. Hog was able to operate over multiple scales by resizing the input image to different sizes. As an improvement to the HOG detector, the Deformable Part-based Model (DPM) was originally proposed by P. Felzenszwalb in 2008. DPM uses the divide and conquers strategy to detect more complex objects by first detecting the smaller parts of the same object. In the training phase, this model learns how to decompose an object and inference involves assembling objects.

All of the above-mentioned techniques are done using hand-crafted features. The performance of handcrafted features saturated after 2010 and there were no new inno-

vations. However, convolutional neural networks improved over time and were capable of learning high-level feature representations. In 2012 CNNs achieved one of the best steps toward object detection. This broke the object detection deadlock with the novel implementation of the Regions with CNN features (RCNN) for object detection.

Ross Girshick et al published a paper named Rich feature hierarchies for accurate object detection and semantic segmentation [8] in 2012 which takes the object detection community to a new way of looking into this problem. They have used a set of object proposals by running an algorithm called selective search. The proposals are then resized and sent to a pre-trained CNN model to extract features. Finally, they have used a linear SVM classifier to predict the presence of an object in each region and recognize the object categories. Although the RCNN does much more than traditional methods there were some drawbacks. There were many redundant object proposals which led to extremely slow detection speeds. Since the selective search is a fixed algorithm there is no learning happening at that stage this could lead to the generation of bad candidate region proposals.

Spatial Pyramid Pooling Networks [9] was proposed by K. He et al with the intention to solve the different scalability problems in images, which means the current network architectures need to have a specific input resolution. But if the same image is rescaled and provided again with a smaller size, then it will reduce the detection accuracy. So the network is not robust to size/scale. They propose a novel pooling strategy called ”spatial pyramid pooling” in this work, which enables the network to generate images with a fixed-length representation regardless of the image’s size. It is also found that this is also robust to object deformations. Adding SPP net concept to any general network boosts the accuracy of the network. Unlike the RCNN method, this will run only once through the CNN layers.

Fast R-CNN’s [7] primary advantage over RCNN is that it uses the entire image as the CNN input for feature extraction, rather than each proposed region. Again the selective search is applied to the image to get the object proposals. ROI pooling is introduced in Fast R-CNN which uses CNN features and region proposals as input and outputs a concatenation of the features extracted. Finally, these features are used to feed to the classification and regression layers to get category prediction and bounding box predictions. While Fast-RCNN successfully combines the advantages of R-CNN and SPPNet, its detection speed is still constrained by proposal detection.

In 2015, S. Ren et al. proposed the Faster RCNN [23] detector shortly after the Fast RCNN. This was the first near-real-time object detector based on deep learning. All the previous methods use selective search for the region proposals which is slow. Here they have removed the selective search and trained a network to predict the region proposals. The region proposal network is trained together with the rest of the model. Finally, the region proposal network can be trained to generate high-quality proposed regions, allowing the number of proposed regions to be reduced while maintaining object detection precision.

T.-Y. Lin et al. proposed the concept of Feature Pyramid Networks [17] in 2017. This addressed one of the known drawbacks of Faster-R-CNN which is the inability to capture the small objects in the image. One of the simplest ways to resolve this issue is to use image pyramids, which scale the image to various sizes before sending it to

the network. With these inputs, the detections are detected at different scales. Finally, all of these predictions can be combined using together. Typically when going into the deeper layers the features are advantageous for categorization but it's not conducive to localising objects. In FPN, authors develop a top-down architecture with lateral connections for the purpose of developing high-level semantics at all scales. As a result, FPN demonstrates significant advancements in detecting objects of varying sizes.

All the previous methods use some kind of region proposal method to localise objects in the image. But here YOLO [22] uses a different strategy, this network looks at the whole image at once and then predicts all the bounding boxes and class probabilities at once. This is also called one stage detector because the network can be inference and optimised as a single step. YOLO is the fastest object detector so far, but this comes will a drop in the object localization accuracy compared to the state of the art two-stage detectors.

Single Shot MultiBox Detector [19] was proposed by W. Liu et al. in 2015, this is also a one stage detector like YOLO. SSD removes the conventional object proposal generation and feature sampling stages and combines everything into a single network. The SSD was able to outperform YOLO's accuracy while maintaining a slightly slower inference speed. This was accomplished by combining multi-scale feature layers, which simplified the prediction process for multiple scales.

# Chapter 3

# IMPLEMENTATION & METHODOLOGY

## 3.1 Introduction

This section comprised a detailed description about the methodology followed to achieve the above stated objective of the project. By gathering background information upon environments using past research works, brainstorming sessions, having continuous discussion with the supervisor and advisor came up with proper approaches, which are defined in each sub-section. The ultimate goal of all these effort is to correctly predict the events and their temporal location in broadcast Rugby Videos.

## 3.2 Datasets

### 3.2.1 Introduction

In any machine learning task, a learning algorithm is chosen with some data to train on it, hence two things can go wrong in a given task. The machine learning task can go wrong because of the bad data or else it could go wrong due to a bad algorithm. So it's very important to choose the right dataset for the right algorithm. Even nowadays with the deepleaning era, the data plays a big role in getting what you want. The quantity and quality of data is important in machine learning. For a child to learn what an elephant looks like, all it takes is for you to show the child an elephant and tell him that it's an "elephant". Then and there onwards the child is able to recognize elephants in place even if it's a baby elephant or a grown elephant. But the machine learning domain is not quite there yet, it takes lots and lots of data to train a ML algorithm. Even for a simple problem like detecting apples vs orange images you need to have at least close to hundred of apples and oranges images in different shapes, colours and sizes. For a more complex problem like speech recognition you may need millions of data points to train a model from scratch.

In 2001 a famous paper was published by Michele Banko and Eric Brill of the Microsoft research team who trained a complex problem of natural language disambiguation and showed that very different algorithms perform very much similar when there is enough data.

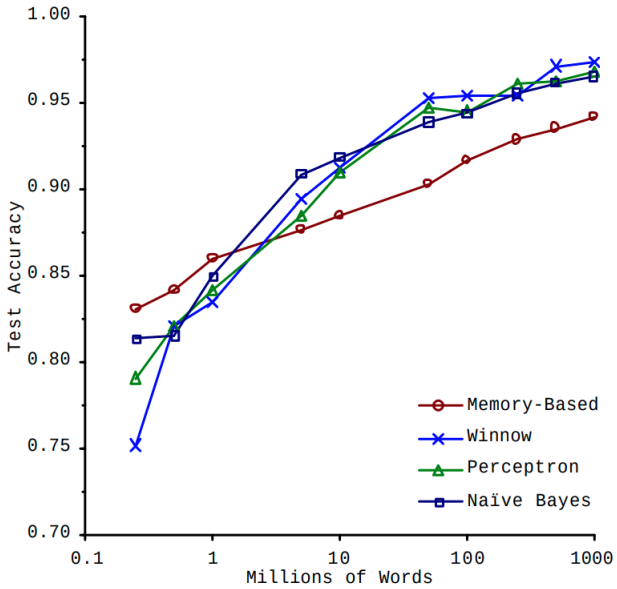



Figure 3.1: Learning Curves for Confusion Set Disambiguation

The authors state that "results suggest that we may want to reconsider the trade-off between spending time and money on algorithm development versus spending it on corpus development". This makes a very important message on machine learning tasks that data plays a key role in this whole process. Still it's very difficult to create lots and lots of annotated data.

As we discussed, data is an essential part of any AI model and it's one of the most contributing factors in the spike of the popularity of machine learning, which is mainly due to ML matching or exceeding human capabilities in a variety of tasks that are hard for a computer to do. One example for this is the ImageNet dataset which has 1000 different classes which contain 1.2 million training images, 150 thousand testing and 50 thousand validation images. This dataset was used by many state of the art models to get the best accuracy exceeding traditional methods.

In this paper for this specific action detection on sports videos, there were no datasets which are currently available for this specific task. Another main contribution of this paper is the event detection dataset for Rugby sports. This dataset consists of two things. The temporal video annotations where the events are and the spatial bounding box annotations for extracted images from the video.

### 3.2.2 Data Extraction

Data extraction can be defined as the process of collecting data from a variety of sources, many of which are unstructured. In this research the Rugby sports video data is extracted properly in order to annotate and then train models. In today's context the broadcast sports videos are within a keypress, the videos are all over the internet. The primary source of raw data for this research would be the broadcast sports videos. There are many Rugby sports videos available in popular video archives such

as YouTube.

Papare.com is one of the pioneers in sports broadcast coverage in Sri Lanka. Their youtube channel has thousands of sports videos. For this research the Dialog Rugby League 2019/20 matches were used for training and testing. There were more than 50 rugby match videos with different team combinations. The best rugby clubs in sri lanka is competing in this tournament and there are 8 teams in this league, namely

| | |
|---|---|
| 1.Air Force SC | 5.Havelock SC |
| 2.Army SC | 6.Kandy SC |
| 3.CH & FC | 7.Navy SC |
| 4.CR & FC | 8.Police SC |

All those match videos were available on YouTube. YouTube is a video sharing platform which is owned by Google. Millions of videos are freely available to watch and download. In YouTube there are different resolution formats available for data download. For this research the resolution was chosen as 480P (480 x 640) which is a low end resolution for the model input. Videos were downloaded using a script and saved to the training machine. When the initial video annotations were done the video was then splitted into different event segments using the starting and end timestamps. Then finally images were extracted from these clips with some equal distances with respect to the events.

| Sports Event | Average time for an event (seconds) | Time between two events (seconds) |
|---|---|---|
| Scrum | 20 | 300 |
| Ruck | 5 | 10 |
| Lineout | 25 | 300 |

Table 3.1: Rugby Sports Event information

The time between two images were decided with the significant changes to the event and the average time taken for an event to be completed. These parameters add up to how well the dataset is generalised to the original problem.

### 3.2.3 Data Cleaning and preprocessing

After the data extraction part then comes the data cleaning and preprocessing part. Data preprocessing tasks mainly focus on the raw data and then transform the data into a more meaningful and understandable format. This is a fundamental stage in any machine learning problem. This directly affects the outcome of the machine learning model, because the machine learning task is highly dependent on its data. So it's very important to give proper attention to this data cleaning and preprocessing part.

Mainly data cleaning is done for the structured tabular data, where we check for missing values and replace them with some strategy, deal with categorical data conversion to values, feature scaling, standardisation, normalisation. But for image data

most of these cleaning and preprocessing steps are irrelevant. When going through the extracted images there were couple of unwanted data is found, such as

- Motion blur images due to the camera movements.
- Images which are not a true representation of actual events.
- Scoreboard overlays in the middle of the event.
- Excessive zooming and the entire event is not visible.
- Poor visibility due to low outdoor lighting levels.

The above mentioned event images need to be filtered from the dataset. This was done with the manual image check and removed and cleaned all the images.

Even if the videos were downloaded as a fixed size there were some videos which the resolution is higher than the expected, the images extracted from those videos needed to be scaled down for the annotation work. Also some images need to be corrected in order to efficiently make use of training. There are a couple of techniques to overcome the underwhelming images to convert it to more useful images.

- Pixel brightness transformations(PBT)

  This transformation modifies the properties of the pixel itself by changing pixel transformation and brightness. This output value of the pixel solely depends on the input pixel value. Some of the examples of this operation are used to change brightness and contrast of the image as well as colour connection and transformations.

- Gamma Correction

  In Gamma correction the individual pixel values are adjusted non-linearly. In this process the relation between input and output pixels are non linear. This it different to the image normalisation which is carried out in such a way that scaler multiplication and addition/subtraction pixel wise.

- Histogram equalisation

  Histogram equalisation is a well-known contrast enhancement technique which is used to improve the contrast on images. This can be applied to almost all types of image which made this technique very popular among people. This checks on the most frequent pixel intensity values and then spreads out those values.

The above mentioned preprocessing methods are used on some of the images to enhance those in-order to create the dataset for the annotation. These techniques are really helpful to improve the quality of the images for training. If the training dataset is better the better the model would be. It would be always better to have a very clean training image set without any noisy images.

### 3.2.4 Annotation

Image annotation plays an important role in computer vision, all of the computer vision models are trained on these images. For our use case we need to annotate images to extract data. The broadcast sports videos are in video format, so first we need to annotate the videos. Video annotation is the process of labelling or tagging video clips. Here in this research, around 50 videos were selected for the annotation. These annotations are carried out in order to train classification and object detection models.

#### Annotation process

There are lots of steps involved in the annotation process. From downloading the videos to annotating events in images the annotations are carried out as steps.

1. Download the videos from YouTube
2. Watch the video and mark the starting and ending positions of each events
3. Extract the sport event clips from the original video
4. Extract images periodically from the event clips
5. Use annotation tool to mark the annotations

The different annotation types were carried out for the different use cases. The use cases for the annotations are as follows.

1. Activity Detection

   In these annotations the main separator is the playing scenes and other scenes. The videos were mainly checked for the play time and none play times. Activity detection was carried out with two different paths.

   - Play and no-play annotations
   - Digital banner annotations

   The first annotation path aim is to identify the playing frames of the video. These annotations consist of the main playground and players. But there are situations where the scoreboard or the player cards are overlaid on top of the play event. To identify those situations digital banner detection needs to be implemented.

2. Events Detection

   The next use case is event detection. As the initial phase three events were identified as the main interest of this research and work on those main events. Those events are

   - Scrum
   - Line-out
   - Ruck

The first two events Scrum and Line-out are considered as the game pausing events. Because those events are started after pausing the game. And the ruck is an ongoing event, because that is happening continuously throughout the play. For the event detection the annotations should contain the location and the type of the event. So the annotation tool is used to annotate those things in each image.

### Annotation Tools

There are many annotation tools available out there. Many annotation tools are freely available. Some of those are as follows

- Labelme

  LabelMe is an annotation tool created by the MIT Computer Science and Artificial Intelligence Laboratory. It provides many useful features to quickly annotate an entire dataset with different annotation formats. This tool needs to be installed in the local machine inorder to use it.

- LabelImg

  LabelImg is also a similar tool to Labelme, this tool is written in Python. But this only offers one annotation type, rectangle shape or a bounding box. This needs to be installed locally to do the annotations and it can export the annotations as the PASCAL VOC format.

- ImgLab

  ImgLab is another image annotation tool which is free to use. This is a freely available web-based online tool which can do various annotations. Even though this tool works in a web browser, there is no online storage for annotations.

- VoTT - Visual Object Tagging Tool

  VoTT is a tool developed by Microsoft. This is free and open-source so this can be used by anyone. This tool provides the support for video and image annotations and also support to attach online storage accounts such as s3 buckets.

- CVAT - Computer Vision Annotation Tool

  CVAT is an open-source, freeware program which is capable of annotating videos and images, developed by many developers all over the world. This is currently supported and maintained by Intel. The online version of the tool is available at https://cvat.org and also it gives the ability to run it locally using docker containers.

- COCO Annotator

  COCO Annotator is a free and open source tool which claims to have additional features where other annotation tools fall short. The objects can be marked as

bounding boxes or masking points. This allows users to export to COCO annotation format.

- Roboflow

  RoboFlow is an online tool which can be used to annotate and store the datasets. The free version is available with limited functionality which is well enough for the annotation purpose.

| Tool Name | Freeware | Remote or Online | Support many export formats | User Friendliness |
|---|---|---|---|---|
| LabelMe | Yes | No | No | Low |
| LabelImg | Yes | No | Yes | Low |
| ImgLab | Yes | Yes | Yes | Low |
| VOTT | Yes | No | Yes | Medium |
| CVAT | Yes | Yes | Yes | Medium |
| COCO | Yes | No | No | Medium |
| Roboflow | Limited | No | Yes | High |

Table 3.2: Different Annotation Tools Comparison

As mentioned above, there are many annotation tools which are freely available for this task. And the comparison is as above. Roboflow was chosen as the annotation tool because it is easy and free to use. This annotation tool was used to label classes and draw bounding boxes around those. The tool itself has some tools to do preprocessing to the images and also create annotations for the same. But those features were limited in the free version hence it was not used for this research.

### Annotation formats

As there are different annotation tools, there are different annotation formats too. Some popular datasets are published with some unique annotation formats, which are named after those datasets. Such as COCO and PASCAL VOC annotation formats. Also some annotation tools provide annotation formats which are unique to those tools, such as CVAT format which is based on their own specific XML based annotation format. Here are some of the popular annotation formats which could be used to annotate bounding boxes.

1. PASCAL VOC

   This format consists of XML files. For each image it generates an XML file even if it consists of a bounding box or not. The annotation pixel coordinates are in absolute values and each of these bounding box annotations contains a tag which stores the class of the object. Some extra information other than the class also can be stored in this format.

2. COCO

Common Objects in Context (COCO) annotation format is represented by a single JSON file which contains all the information. There are separate keys in the json to list down categories, images and annotations. Images and categories are referenced by an id which is used back in the annotations. Bouncing boxes are annotated with their absolute top-left beginning with and explicit values for width and height.

3. YOLO

In this representation text files are used to store annotations and each image has its own annotation file. The annotations are recorded line by line in the text file. A single line of the text file contains the bounding box coordinates and the class ID(c). There is an extra annotation file called class text file which has the mapping of classes to class IDs. These bounding box coordinates are marked as the centre of the bounding box (x,y), the width(w) and the height(h) of the box, all of those values are not absolute; those are normalised by the image height and width. Then the final annotation format looks like [c, x, y, w, h] format. This format is easily used with different resolutions of images due to the normalised annotations.

4. VOTT

Visual Object Tagging Tool produces these types of annotations and each image has a relevant JSON file to store the bounding boxes and the classes. The coordinates are expressed as the upper left pixel coordinate (x,y), width and height in absolute coordinates.

5. CVAT

CVAT annotation format has a unique XML file format which captures all the bounding box information of the dataset. The bounding box coordinates are recorded as the top left corner coordinate and the bottom right corner of the bounding box. All the values are recorded as absolute pixel coordinates. This format is specially used in the CVAT tool to store its own annotated data.

6. TFRecords

This annotation format is specially designed by the tensorflow developers with the intention of the efficient use of the storage and the speed of accessing it. This format combines all the images and the annotation files together and represents it as a serialised format. It can also be partitioned into multiple files. This format is supported by the tensorflow library and many state of the art models which use tensorflow are used in this format to speedup the training.

7. Tensorflow Object Detection

TF object detection annotations are stored in a CSV file which contains all the dataset annotations in one file. It contains filename, image size, class and bounding box information in each line. The bounding box format is represented by the upper-left pixel coordinate and bottomright pixel coordinates. All the pixel coordinates are

recorded in absolute values. This is mainly used in the Tensorflow Object Detection API.

### 3.2.5 Data Augmentation

The accuracy of the supervised Deep Learning model is highly dependent on the training data. If the amount and the diversity of the data is higher means the model is more generalised for the unseen data. Deep Learning models which are trained to achieve high performance on complex tasks generally have lots of neurons. The number of parameters in state of the art computer vision models such as Inception-V3 (24M) and ResNet (60M) is in the order of ten million. YOLO as an object detector also in this same region. These models which are doing some complex tasks such as object detection have a large number of tunable parameters. In simple terms the amount of data is proportional to the number of learnable parameters in the model. And the number of traininable parameters are proportional to how complex the task is.

So it's clear that for our task there should be lots of data available for the training. Since we are generating the annotations from scratch it's a very difficult task to increase the data as much as we want. Another way to deal with this limited data problem is applying different transformations on the available dataset to synthesise new data. This approach of synthesising new data out of existing data is called Data Augmentation. Diversity of the training data and the amount of training data can be improved by the data augmentations. Also this can be used to address the class imbalance problem which is frequently observed in the classification tasks.

It is proven that most of the popular computer vision tasks such as image classification, semantic segmentation and object detection could be very effective with data augmentation. There are mainly two types of augmentations

- Geometric Transformations

There are many spatial transformations that can be applied to an image. Applying these transformations is a geometric transformation of the image coordinate system.

Ex: Flipping, Rotation, Translation, Cropping, Scaling

- Colour Space Transformations

This is the process of changing the pixel values without moving pixels through the image.

Ex: colour casting, Varying brightness, and noise injection.

Geometric transformations work well when positional biases are present in the images such as the dataset used for facial recognition. The colour space transformation can help address the challenges connected to illumination or lighting in the images.

### 3.2.6 Sports Event Datasets

One of the most time consuming task in computer vision field is the dataset creation. In this research, there were no public dataset was available for the Rugby sporting events. The data extraction, preparation was done as a part of this research. Along with this research two datasets have been published to the computer vision community.

1. Event Clip Dataset.

   This dataset is consists of the start time,end time and event type annotated for more than 1600 events spred across 38 videos. The information for each video is stored in a CSV format and the video links are available to download the videos.

| Event Type | No of Annotations |
|---|---|
| Scrum | 532 |
| Ruck | 720 |
| Lineout | 370 |
| Total Annotations | 1622 |

Table 3.3: Rugby Events Clip Dataset Information

   More information about this dataset is available in this location. https://github.com/CodeProcessor/rugby-events-dataset/tree/main/video-annotations

2. Event Object Detection Dataset.

   This dataset consists of images with bounding box information for the three events (scrum, linpout, ruck) which were considered in this research. There are more than 4000 images annotated for different Rugby sporting events. The annotations are in the format of YOLO, which having a text file for each image with the bounding box information.

| Event Type | No of Annotated Images |
|---|---|
| Scrum | 1,301 |
| Lineout | 1,281 |
| Ruck Overlays | 1,646 |
| Total Images | 4,228 |

Table 3.4: Event Object Detection Dataset Information

   More information about this dataset is available in the following link. https://github.com/CodeProcessor/rugby-events-dataset/tree/main/image-annotations

## 3.3 Model Development

### 3.3.1 Model Selection

Model selection is the process of choosing one model from a list of available models for a specific task. This selection process is done by comparing many metrics such

as model performance, model complexity, maintainability and available resources. In this research to solve the primary problems there are many deep learning models available. But its all about selecting the best model for the problem. The main objective of this section is to select the best model for the event detection problem.

### 3.3.2 Activity Detection model

This research is based on mainly two detection methods. Activity detection and event detection. Activity detection is the process of identifying whether the camera view is capturing a playing moment or not. There are a couple of no-play moments in a typical sports video.

- Advertisements are showing in the video.
- Camera is focusing to crowd / out of the ground area.
- Overlaying scoreboards, player cards.
- Focusing to the ground but the game is paused or not started.

These sections in a video are not related to actual gameplay and should be carefully detected and removed from the event detection phase.

In order to detect the advertisements and out of ground areas. The best model would be a classification model. Which uses the entire frame to check whether it's a frame with the play area or not. But what about overlaying scoreboards or player cards. For that we have to use an object detector to detect these digital overlays on the video. So we decided to train two models for the activity detection, one classification model to detect the none play frames and one object detection model to identify the digital overlays.

### 3.3.3 Selecting a classification model

There is an extensive review on classification models under the literature review. The best model so far was the Efficentnet. But in this specific case, we are trying on a simple classification problem. There will be only two classes for this problem. Where the accuracies for those models were tested on image net dataset which consists of 150 thousand test sets with 1000 different classe. These state of the art models work very well for this classification task. Instead of the best model, taking the most efficient model is the best choice.

| Model Name | Number of parameters | Number of Flops |
|---|---|---|
| Densenet 169 | 14 Million | 3500 GFlops |
| EfficientNet-B0 | 5.3 Million | 390 GFlops |
| **Resnet 18** | 11 Million | **233 GFlops** |
| Mobilenet v2 | 3.4 Million | 300 GFlops |

Table 3.5: Classification Model Computational Complexity

The above diagram shows the smallest state of the art image classification models. Here we can see Resnet-18 is the most efficient in computation complexity. This is the main reason for choosing the Resnet-18 model for this project.

### 3.3.4 Human Pose Estimation as a Feature Extractor

In this research the initial approach is to take the human pose estimation as a high level feature extractor for the event detection network. When initial tests were done for the detections there were some complications down the road which made us avoid this human pose estimation entirely. Some of the reasons are listed as follows

- Occlusions in Rugby Videos

  Rugby as we all know is a full contact game. The players are all over the field. Most of the time there are many player occlusions happening inside a frame. This is a major issue for the Human Pose Estimation. There are some pose estimations which are robust to some form of occlusion but in a game like this most of the time there are occlusions in the field.

- Players are too small in some views

  These broadcast videos have different views, Sometimes the video is focused into an area of action and sometimes it takes the entire aerial view of the play area. Especially when the play area of the Rugby match is massive (100m x 70m) the camera most of the time is covering the half of the ground. Which makes the human pose estimation model less favourable for detecting humans at all.

- Distinguish players from the spectators

  In live broadcast videos the video footage is not only capturing the players. It also captures the spectators and other people around the ground. This makes it worse using these kinds of high level feature extractor. The player and non player identification would be another task down the pipeline.

After evaluating carefully and come-up with the above reasons, the human pose estimation is not used in this research. The human pose estimation is discussed as a potential feature extraction method in the literature review, but after the initial results with the pose estimation the direction changed to the object detection methods.

### 3.3.5 Event Detection model

The main task of this research is detecting sports events in a broadcast video. As stated in the previous section the human pose estimation is not a useful method to extract features for this task. We had to choose another method to detect the events in the video. The next best approach would be to test with an object detection model. Object detection is different from classification tasks, this adds another dimension to the classification problem. Object detection is locating the objects and identifying each of those.

Even if the group events cannot be considered as an object, we assumed that object detection models have the potential to detect these types of shapes in the video. As discussed in the literature review there are many models which are used for the object detection domain. The fastest and easiest to train model would be the YOLOv5 which is already implemented using the PyTorch framework with many optimizations. YOLOv5 was explored and used as the first step in this research.

### 3.3.6 Selecting a Object Detection Model

In this research we have been using YOLO heavily for object detection. There is one model for detecting digital overlays on the video as we discussed in the above section and the other model is for the event detection. There are a couple of reasons for using YOLO. Some of the reasons are as follows

- YOLO is well known and improved architecture for object detection

  YOLO Object detection has a rich history with many more additions to the original concept. YOLO v1 was published in 2015 and there are sequences of papers which take the concepts of the previous paper and improve on top of that. Which itself makes the YOLO robust and improve over the time. Most of the improvements are discussed in the previous sections to this post.

- YOLO inference speeds are pretty fast

  The most important advantage of using YOLO is its unparalleled speed in object detection. It can process hundreds of frames per second in a modern GPU which is essential for any application. The application we are developing is highly efficient with the speed of YOLO.

- YOLO class accuracy is comparatively good

  YOLO v1 peper mentioned that their localization accuracy is a bit behind the state of the art models. But later down the line there are many improvements to this end. Even if the localization accuracy is low, in this application the main focus is detection. The tracking will compensate for the spatial localization loss with the IoU threshold and helps to get the temporal localization accurately.

- YOLO is easy to train

  The original YOLO which is trained on Darknet is somewhat difficult to use. But YOLO v5 implementation is in Pytorch which supports many more additions. One of the biggest reasons would be to use this one is how easy it's to use this repository.

Following there is a extended information about the YOLO model and how it has been evolving over the years

### 3.3.7 Introduction to YOLO

YOLO, which stands for "You Only Look Once," is a popular object identification technique that has been around for a while. This is a one-stage object detector that separates images into grids. Where each grid cell is in charge of detecting items within itself. The latest and widely used implementation of the YOLO is YOLOv5 which is written using the Torch framework. This codebase is freely available in github.com and it's easy to implement the training pipeline for this method. Let's scratch the surface of YOLO and how it works.

#### Why YOLO?

YOLO is very simple compared to the RCNN, Fast-RCNN and Faster-RCNN Models. YOLO provides a single convolution neural network capable of predicting class probabilities and bouncing boxes for a given image at the same time. Because it is a one-stage detector, detection performance may be easily optimised using the entire set of annotated photos. According to the authors, this approach has significant advantages over typical object detection algorithms.

According to YOLO v1's publication, the model can run at 45fps on a Titan X GPU, making it the first real-time detector at the time. This speed is achieved because it considers the problem of object detection as a single regression problem. YOLO takes the entire image while training as well as testing. The second main benefit of this model is that it learns through the whole context of the image unlike early sliding window methods or the region proposal based methods. They have compared this fact with the previous state of the art model Faster RCNN and found out that when compared to its predecessor, YOLO creates fewer than half the number of background errors. The third is that the YOLO model is able to learn generalizable representations of objects. This is a huge advantage for this project as well because when YOLO applies to new domains, it is less likely to fail. This means the YOLO model can easily transfer-learn on the other object detection tasks.

The above discussed benefits are one of the reasons behind selecting the YOLO as the object detector for this paper. The main drawback of the YOLO paper stated about the localization accuracy. Our use case is not centred around localising events, this doesn't matter a lot for the event detection. Localization is a part of event detection but for the real use case this component is less important.

#### YOLO Architecture

The name YOLO stands for 'You Only Look Once' which explains its straightforward, simple pipeline where the object boundaries and their respective classes are predicted simultaneously as a single pass given the input image pixels; which is also the main factor for its capability to process videos in real-time. Moreover, it is capable of grasping the global context of the image by looking at the full image, which is the reason for fewer background errors compared to previous object detection implementations.

In terms of the YOLO system, it uses a grid that equally divides the image into (S x S) grid cells where each grid cell is accountable for predicting the object, Pr (Object), if the object centre falls inside it. Moreover, B bounding boxes are forecasted by each grid cell, together with a confidence score for each box which implies the model's confidence of having an object inside the predicted box and how accurate the estimated box is. For instance, not having an object inside a bounding box would result in a confidence score of zero. Each box prediction consists of five elements, x, y, w, h, confidence, where, (x, y) denotes the centre of the box in relation to the grid cell limits and the (w, h) are calculated in relation to the whole image. In addition to B bounding boxes, a set of C conditional class probabilities Pr(Class—Object) are predicted per grid cell by YOLO, which are conditioned on the grid cell that contains an object. Consequently, the ultimate output tensor of YOLO is of size (S x S) x (B x 5 + C), where, S = 7, B = 2 and C = 20 for Pascal VOC which has 20 named classes. The following is a diagram of the YOLO architecture.

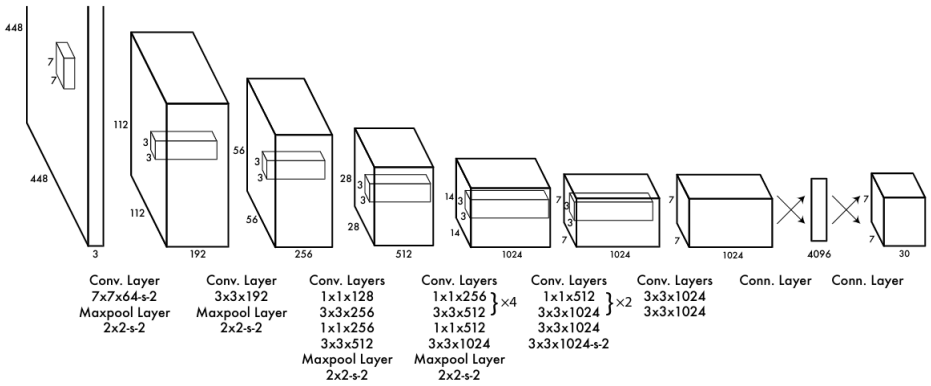



Figure 3.2: YOLO v1 network architecture

### YOLO Training

The last layer of YOLO predicts class probabilities and bounding box coordinates on a scale between 0 and 1. This is because, the (w, h) are normalised to the actual width and height of the image and (x, y) is parameterized as an offset of a certain grid cell position.

The YOLO architecture uses ReLU non-linearity (rectified linear activation) everywhere except the last layer which uses a linear activation function. Due to the simplicity to optimise, YOLO authors used the sum-squared error, even though it is not entirely aligned with the target of maximising average precision. To remedy this YOLO has increased the loss for the bounding boxes which contain objects and decreases which have not. They have used two parameters for that Lambda.

### The family of YOLO models

YOLO v1 has several flaws when compared to state-of-the-art object detection techniques. Compared to Fast R-CNN, YOLO makes a disproportionate number of

errors in localization and it has a low recall when compared to region proposal-based approaches. Therefore, the subsequent YOLO models try to improve the existing architecture even further.

YOLO v2 was released in 2017, and it received an honourable mention at CVPR 2017. On top of YOLOv1, this design made a number of iterative enhancements. YOLO v2 includes batch normalisation, which significantly improves the convergence and also eliminates the requirement of having other forms of regularisation. Adding batch normalisation itself made a 2% improvement on mAP. YOLO v1 was trained on a smaller 224 × 224 classifier network before increasing the detection resolution to 448. This means that the network must adjust for object detection and updated input resolution at the same time, which can be challenging. In YOLO v2 the classification network was directly trained for 10 epochs in full 448 x448 resolution on ImageNet which allows the network to adapt its filters to perform well on high-resolution input images. Moreover, the YOLO model has removed dense layers and gets the assistance of anchor boxes for prediction, which allows the model to predict class and objectness for every anchor box which uncouples the class prediction process from the spatial location. Using anchor boxes YOLO v2 got a small decrease in accuracy but it increased the recall which means the model has the potential to enhance further. Another improvement was that the authors defined a better way to pick priors for anchor boxes instead of choosing priors by hand. These fine priors were found by running the k-means clustering algorithm on the training set bounding boxes. This has given the ability to have the similar performance of having 9 anchor boxes, with just 5 anchor boxes with an average IOU of 61.0 compared to 60.9. As a method of having a range of resolutions, object detectors such as SSD and Faster- R-CNN run their RPN (region proposal network) at various feature levels. YOLO V2 authors have used a different approach to this by having a skip connection (passthrough layer) to pass the lower level features from an earlier layer at 26x26 resolution, which turns the 26×26×512 feature map into a 13×13×2048 feature map, that can be concatenated with the original features. This expansion in the feature map leads to a 1% overall performance increase since there is a way to approach fine-grained features. Since YOLO v2 is a fully convolutional neural network, it can be resized on the fly. Instead of fixing the input image size, authors have changed the network every few iterations. Every ten batches, the network selects a new image dimension size at random. This regime compels the network to learn to predict accurately across a wide range of input dimensions.

| | Type | Filters | Size | Output |
|---|---|---|---|---|
| | Convolutional | 32 | 3 × 3 | 256 × 256 |
| | Convolutional | 64 | 3 × 3 / 2 | 128 × 128 |
| 1× | Convolutional | 32 | 1 × 1 | |
| | Convolutional | 64 | 3 × 3 | |
| | Residual | | | 128 × 128 |
| | Convolutional | 128 | 3 × 3 / 2 | 64 × 64 |
| 2× | Convolutional | 64 | 1 × 1 | |
| | Convolutional | 128 | 3 × 3 | |
| | Residual | | | 64 × 64 |
| | Convolutional | 256 | 3 × 3 / 2 | 32 × 32 |
| 8× | Convolutional | 128 | 1 × 1 | |
| | Convolutional | 256 | 3 × 3 | |
| | Residual | | | 32 × 32 |
| | Convolutional | 512 | 3 × 3 / 2 | 16 × 16 |
| 8× | Convolutional | 256 | 1 × 1 | |
| | Convolutional | 512 | 3 × 3 | |
| | Residual | | | 16 × 16 |
| | Convolutional | 1024 | 3 × 3 / 2 | 8 × 8 |
| 4× | Convolutional | 512 | 1 × 1 | |
| | Convolutional | 1024 | 3 × 3 | |

YOLOv3 was released in 2018 with several improvements on top of the previous version. YOLOv3 uses logistic regression to predict an objectness score for each bounding box. This score is 1 if the box prior overlaps with the ground truth by more than any other bounding box prior. If it overlaps more than a threshold but not the best

overlap then they ignore the prediction which makes sure that only a single box is assigned to the ground truth object. If such bounding box priors to ground-truth object assignment do not occur, no loss is incurred for class predictions or coordinate, only objectness. Furthermore, YOLOv3 has gotten rid of Softmax activation due to the fact that each box has exactly one class which may not be correct all the time. Instead, it uses independent logistic classifiers with binary cross-entropy loss during training for the class predictions. Thus, it supports a multi-label approach that better models data for the object detection task. Moreover, similar to the feature pyramid network concept, YOLOv3 does the box prediction in 3 different feature scales. The last layer of the base convolutional feature extractor stack, predicts a 3D tensor which consists of a bounding box, class predictions and objectness score. The output tensor of the model is of size, N × N × [ 3 * ( 4 + 1 + 80 ) ] since 3 bounding boxes are being predicted in each scale and each box prediction has 4 bounding box offsets, 1 objectness prediction, and 80 class predictions. Similar to YOLOv2, YOLOv3 also uses K-means to cluster and determine bounding box priors but they have chosen 9 clusters and 3 scales arbitrarily and then divide up the clusters evenly across scales. Furthermore, YOLOv3 introduced a new feature extractor network that uses consecutive 3x3 and 1x1 convolutions with some skip connections. Compared to ResNet 101 and 152, this new architecture is more efficient and still more robust than Darknet-19. This network is known as Darknet-53 since it has 53 convolutional layers. It is 2 times faster and has a similar performance as ResNet 152 while it's 1.5 times faster and performs better than ResNet-101. YOLOv3 improves on previous versions by introducing an objectness score to bounding box prediction, connecting to the backbone network layers, and making predictions at three different degrees of granularity to increase performance on smaller objects.

In 2020 YOLO strikes back with a new version called "YOLOv4: Optimal Speed and Accuracy of Object Detection". This paper was not published by the original authors instead it was published by Alexey Bochkovskiy, et al. YOLO v4 brings many concepts into the YOLO v3 architecture, these concepts are divided into two parts the bag of freebies and the bag of specials. Both concepts improve the model performance but the first one is doing this without increasing the computational complexity. Inside the bag of freebies, there are some improvements in data augmentation techniques, regularization methods, bounding box regression loss and normalization. On the bag of specials side, it has added Non-max suppression, Spatial attention modules, Non-linear activation functions and skip connections.

There is no paper associated with YOLO v5, the next member of the YOLO fam-

ily. This was developed by a company called Ultranytics. They have implemented the whole YOLO architecture using PyTorch framework which is easy compared to the DarkNet implementations. This made it popular among the machine learning community. Nevertheless, there are some technical improvements made down the line by the YOLO v5 repository. This implementation contains better data augmentation and loss calculations, auto-learning of anchor boxes, usage of cross-stage partial connections, use of path aggregation and most importantly ease of use and installation.

There are some other YOLO implementations available through some sources. PP-YOLO is one of such implementations which is done by PaddlePaddle(PP), Baidu's neural network. They claim that they surpass YOLOv4 performance on the COCO dataset. Scaled YOLOv4 was released at the end of 2020 by Alexey Bochkovskiy, which takes advantage of Cross Stage Partial networks to scale up the size of the network while maintaining both accuracy and speed of YOLOv4. The original YOLO paper in 2016 continuously grew with the huge community behind it. This made the path to this research to go with YOLO models for sports event detection.

### 3.3.8 Training Dataset

There were no datasets available for Rugby sport, because of that we have created a fresh dataset for group events in rugby. This was created using the YouTube videos captured by papare.com. More information about the creation of the dataset can be seen in section 3.2. The dataset for the training was manually created by hand. Dataset information as follows.

| Annotation type | No of Images | No of annotations |
|---|---|---|
| Play | 10,120 | 10,120 |
| No-Play | 3,166 | 3,166 |
| Digital Overlays | 958 | 1,579 |

Table 3.6: Activity Detection Dataset

| Event | No of Images | No of annotations |
|---|---|---|
| Scrum | 1,301 | 1,293 |
| Lineout | 1,281 | 1,278 |
| Ruck Overlays | 1,646 | 1,654 |

Table 3.7: Event Detection Dataset

These datasets were created for this project and publicly available in this location.

### 3.3.9 Hyperparameter Tuning

Hyperparameters are parameters where the values of those cannot be estimated using the data. Most of those parameters can be considered as configurations that control the learning process of the model. On the other hand Model parameters are the

parameters that a learning algorithm ends up learning. Hyperparameters play a major role determining the model parameters which is also called training the model. This is crucial for a deep learning model because this can have a direct impact on the training of the machine learning algorithm. Thus, in order to achieve maximum performance, it's important to understand how to optimise them.

### Different hyperparameters

When it comes to deep learning there are many hyperparameters involved. Some of the common hyperparameters and the meaning of those are as follows

- Train-Validation-Test Split ratio
- Learning Rate
- Choice of Optimization Algorithm, activation function, loss function
- Number of hidden layers, activation units, iterations
- Batch size, No of epochs

### Activity classification model

Activity classification model is trained with a low learning rate for around 100 epochs. The input resolution was chosen as 224x224 since that is the standard input for resnet 50 network. Initial weights were loaded using imagenet data and hence the mean and standard deviation of imagenet data was used as the preprocessing step.

| Hyperparamter name | Initial value |
|---|---|
| Momentum | 0.995 |
| Initial learning rate | 0.00001 |
| Weight Decay | 0.005 |
| Standard Deviation | 0.229, 0.224, 0.225 |
| Mean | 0.485, 0.456, 0.406 |
| Input resolution | 224 x 244 |
| Batch size | 16 |
| Epochs | 50 |

Table 3.8: Image Classification Model Hyper-parameters

### Activity / Event object Detection model

YOLOv5 was used as the Object detection model and initially the default parameters were used for the training.

| Hyperparamter name | Initial value |
|---|---|
| Momentum | 0.937 |
| Initial learning rate | 0.001 |
| Weight Decay | 0.0005 |
| Standard Deviation | 0.229, 0.224, 0.225 |
| Mean | 0.485, 0.456, 0.406 |
| Input resolution | 640 x 640 |
| Batch size | 8 |
| Epochs | 100 |

Table 3.9: Object Detection Model Hyper-parameters

### 3.3.10 Model Training and Monitoring

Model training or model parameter fine tuning in machine language is the process of finding the best parameters for a machine learning algorithm to minimise the loss over the training dataset. Deep learning based computer vision models are computationally heavy models when compared with traditional machine learning models. To train these models a powerful PC with a discrete graphic card is a must. Some researchers use cloud computing to tackle this problem but those are very expensive compared to the local machines. For this research the following machine was used to train and test all the models,

| Item | Specifications |
|---|---|
| Cores | 6 Cores & 12 Threads |
| Processor Frequency | 3.6GHz Base / 4.0GHz Turbo |
| RAM | DDR4 8 GB x 4 , 2400MHz |
| Graphic Card | Nvidia GTX 1070 |
| GPU Memory | GDDR5 8 GB |
| Hard Disk | 512 GB NVME SSD |
| Operating System | Ubuntu 20.04 LTS |

Table 3.10: Training Machine Specifications

The training model was continuously monitored by an online tool called weights and biases. It works similar to tensorboard but it stores the information in the cloud real time. This looks like a very appealing tool since this is free for the researchers.

Weights & Biases provides support for key steps in MLOps life cycle. This offers performance visualisation tools, dataset versioning and model management for machine learning. This is a handy tool when training models and keeping their historical data. This site enables users to easily compare and share the experiment results within a click of a button.

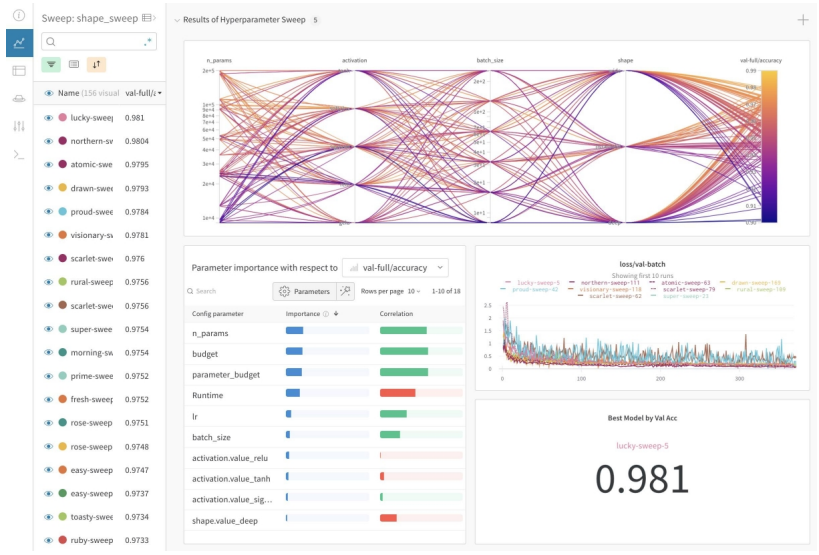



Figure 3.4: Weights and Biases sample dashboard

### 3.3.11 Training and Validation curves

In object detection unlike classification models there are multiple loss curves for different losses. Mainly there are three losses in object detection with YOLO. The final YOLO output layer is useful to identify those losses.

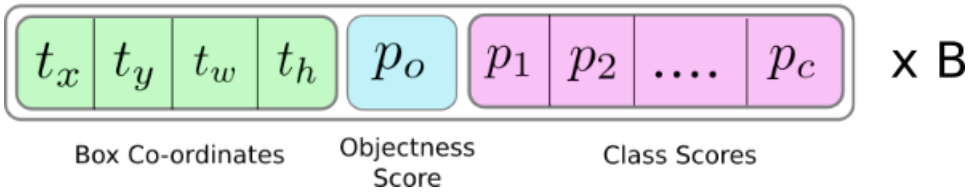



Figure 3.5: Yolo output Tensor Shape

Those losses are as follows

1.Box Loss.

Box loss is the loss which is calculated from the bounding box coordinates. In YOLO the bounding box prediction is output as the centre of the bounding box (tx,ty) and the width(tw) and height(ty) values. This box loss is calculated from these values.

2.Objectness Loss

The objectness loss is the loss where the confidence of the model that there is an object in the given bounding box. For each bounding box tensor this is represented by one neuron.

3.Classification Loss

Classification loss is the loss where the model tries to classify the detected object into one of the predefined classes which YOLO is trained on. This is represented as one-hot encoded neurons which have the length of the number of classes.

With these in mind, let's check the loss curves of the object detection models

- Digital overlays detection model

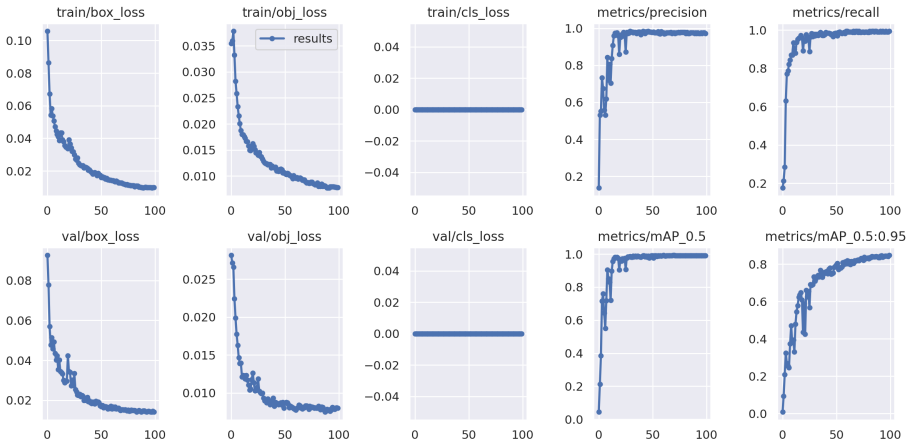



Figure 3.6: Digital Overlay Detection Model Losses

These are the training and validation losses of the digital overlay detection model. The class loss is zero in these plots because there is only one class, so the classification loss is always zero. When looking into the training and validation losses of box and objectness loss. The loss is always getting lower in both graphs which means the model is learning the problem without any issue.

- Event detection model

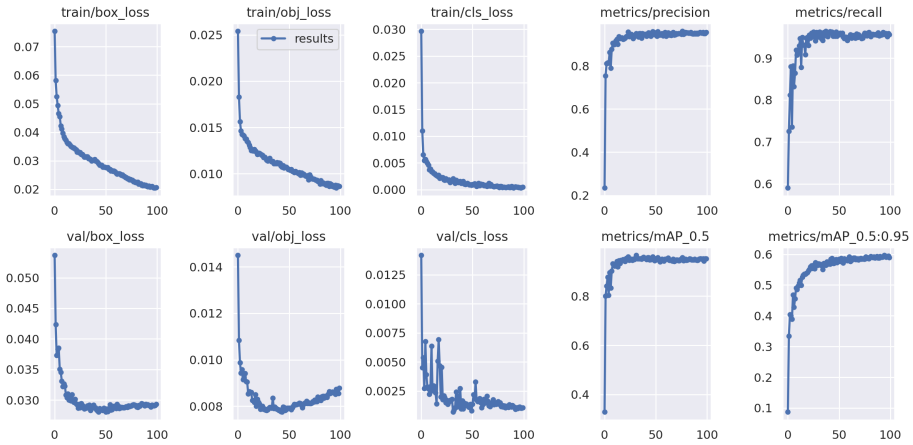



Figure 3.7: Sports Event Detection Model Losses

These are the plots for the event detection model. There are three events in this model, unlike the previous one the class loss is not zero because of multiple events. The training box loss and objectness loss is always going down, but the validation losses for the same is increasing around the 50th epoch. It is suspected that the model is trying to overfit to the dataset. But the best model is chosen with the consideration of the validation losses.

## 3.4 Event Tracking Methods

Detecting events is not just enough. Those events should be tracked within the frames. There can be multiple events happening simultaneously in one frame, how to uniquely identify those events in the next frame? Detecting an event in a frame is not just enough to count that as an event. There can be false positives from the detection model. How to avoid those and get a good accurate prediction? These are some of the questions that need to be addressed in this section. Some of the techniques that have been used for this research are added in the following section

### 3.4.1 Moving Average for Event Tracking

Moving average is not a tracking method in general, and this method won't track the events in the spatial domain. The main assumption of using this method is that, there will be one and only one event happening during a given timestamp. This assumption is not always true, there can be multiple events happening at the same time but relative to the events we are interested in, it's safe to say that those two events won't happen at the same time. So our initial tracking method was the moving average through the video frames.

Before everything else the video FPS is reduced to 5 frames per second, the reason behind this change is the computation overhead. There are a couple of models that need to run on the frames, if we can use minimum frames as possible then we can process the video very fast. Then for every frame we have the model information, whether this frame has an event/events and if there are events what's the confidence and where those events are. First the moving window size is taken as 5 frames to match with a single second. For each frame for each event the confidence scores are checked against a threshold and counted the number of events in each window. If the count is exceeding 60% of the window size and the event type is the highest number of events within that frame window, we consider that window to contain that specific event and mark the timeline where an X event happened in this second. After running this moving window through the entire video we have another event timeline which has the specific event or no event for each second.

$$Event\ Triggered\ in\ n^{th}\ Second = (\sum_{i=n}^{n+fps} C(event)) > 0.6 \times fps, \tag{3.1}$$

$$Where\ C(event) = \{1\ if\ P(event) \geq t$$
$$0\ if\ P(event) \leq t\ \}$$

When we have an events timeline then we can stitch these events together to get a connected events timeline. The event separation can be carefully crafted using some threshold values. The final event start and end times can be taken from this output.

### 3.4.2 Centroid Tracking

Centroid tracking method is another well known simple object tracking method. For this algorithm to work, a set of bounding boxes with class names should be provided. The bounding box centroids will be calculated and taken for this algorithm.

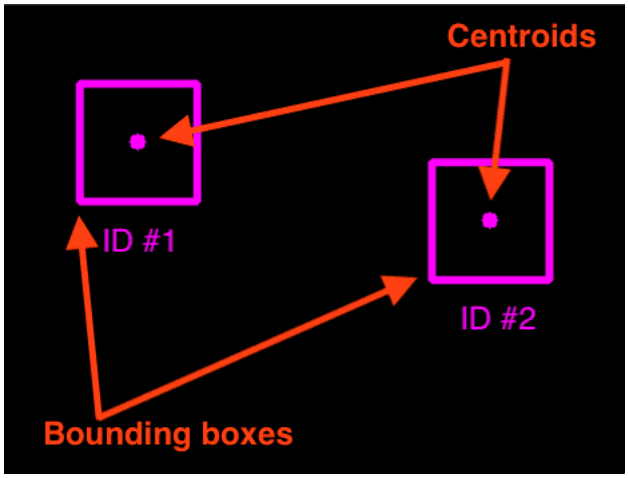



Figure 3.8: Centroid Tracking

When the model predicts the initial bounding boxes some unique IDs will be assigned for those bounding boxes. When the new boxes are predicted with new frames this algorithm tries to match the existing bounding boxes with the predicted ones. In this method the association is done with a distance measurement. Euclidean distance between the centroids of bounding boxes were calculated to get this distance measurement. When an object is moving the distance between the next centroid of the same object should be smaller than the other object centroids. This is the main assumption behind the centroid tracking algorithm. With this assumption the association of the centroids can be done with the minimum distance to the next set of predicted centroids. Whenever there are more predicted objects than the traced objects, then again the initialization with a unique ID would take place.

But what happens if the object did not associate with anything. In that event there could be less predicted objects than the existing objects in the frame, which means the tracked object is lost. In that case there is a separate counter for each object that counts the number of disappeared frames. If this exceeds a predefined threshold then that object is removed from the tracking algorithm.

### 3.4.3 Bounding Box Tracking

This is mostly similar to the above mentioned centroid tracking. The steps are the same as above, the distance measure is slightly different here. Instead of taking a euclidean distance between centroids, in this method we consider IoU between each bounding box. The intuition behind this is that the same object should have a positive IoU between the current frame and neighbourhood frame predictions. Unlike the centroid method this checks for the highest positive IoU. If the IoU is zero it won't assign it to the closest. There are positive things and negative things using this for event detection.

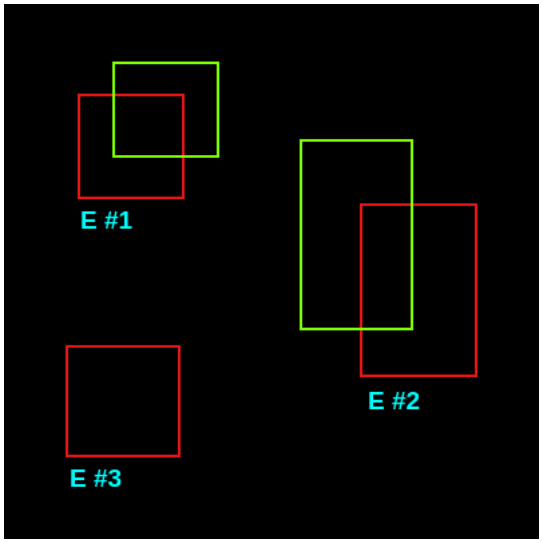



Figure 3.9: Bounding Box Tracking

If there are multiple events happening back to back within a small amount of time this method will correctly identify the events as two events. If there are two events in the same frame the IOU between the predicted and the existing events have a clear separation in the spatial domain, this method won't assign the bounding box unless there is a positive IOU. Whereas in the earlier centroid method will still assign the event to the closest predicted centroid, making it hard to distinguish between two events appearing in the same frame. But one downside of this method is that whenever there is a camera view change even the same event will appear in two different places. Then again this will consider that as two events which are not correct. But the inter-camera view change is not expected in the tracking algorithm.

### 3.4.4 Tracking methods used in the system

As discussed above there are a couple of basic tracking algorithms available. For this project there are two types of events that need to be tracked, activity detection and sports event detection.

- Activity Detection.

The activity detection is done using a classification model and an object detection model. The Classification model won't give the localization information inorder to track the play activity. Also it's pointless to track the digital overlays since it wont move relative to the frame. So the digital overlay object detection is also converted to a classification output by checking a logic.

- Event Detection.

The event detection has only one model which is an object detection model. And for this model the bounding box tracking method is used. The events are tracked across the frames. If an event is detected it is tracked till the event vanishes from the frames.

## 3.5 Event Detection System Flow

### 3.5.1 Introduction

The flow of the system can be divided into two main parts.

1. Event Detection and Localization system
2. Complete Event Detection System

The first one is the core of the system, which does all the heavy duty with the prediction of the events and then combining those frame level information to predict the events. The second part of the system is the overall system which includes the first part. The input of this system is a url which is taken from a web user interface then this will process the video and give a meaningful output to the user. Both systems are extensively discussed in the following section.

### 3.5.2 Event Detection and Localization system

This is the core of the system. Here is where the actual event detection happens. In this research the main focus is mainly on this part. This is the section where the detection and localization of events happens in the video segment. The event detection happens in two stages,

1. Individual frame predictions
2. Combining predictions and get the final output

#### Individual frame predictions

In the first stage (fig1) for every image frame of the video an activity detection and event detection models are predicting on that video and then if there is an activity or event detected in the frame that information is stored in a database. This is done this way because if the user tends to process the same frame twice the system can detect that and skip the precision for already predicted frames. This will be an efficient way to handle the same video predictions multiple times.

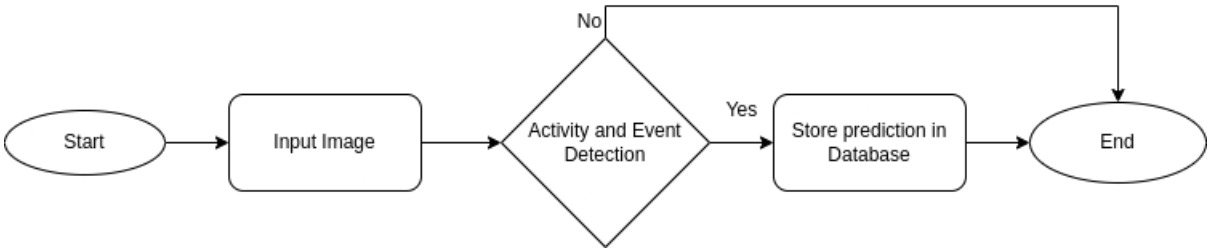



Figure 3.10: Frame Level Prediction Flowchart

**Combining predictions and get the final output**

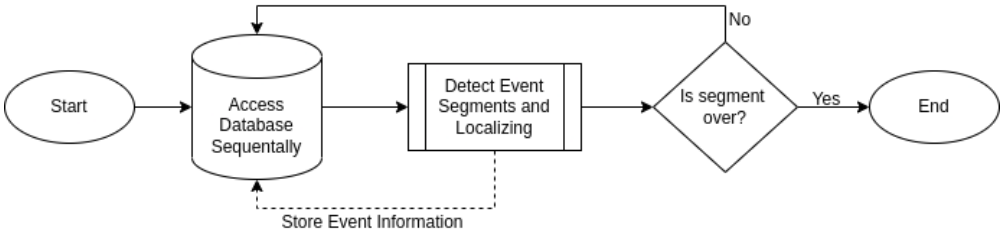



Figure 3.11: Overall Prediction Flowchart

The next phase of this core system is the detection of actual event segments. In the first phase we took the model output directly. But there are lots of noisy events and actions. In this section the main idea is to remove this noise and get the accurate events detected from the predictions. After the phase 1 stage now all the predictions are stored in this database. The database will be accessed from the start of the segment to the end of the segment. The frame data will be accessed sequentially one after the other, then the tracking algorithms will apply on top of the predictions and make sure the noise is removed from the phase1. If the event is available then it is saved back to the database and iterates through the frames till it reaches the segment end time.

### 3.5.3 Complete System Flow Diagram

The whole sports video analysis system can be visualised as in the above diagram. This is a diagram which explains the flow of the system at a high level. Each of these steps can be further broken down into small pieces. This system can be divided into six main parts.

First the input is taken from the user to start processing the video. User needs to give the URL of the video source with the start and end point timestamps. In the second stage (b) the video is downloaded from the URL and then processed in the stage (c). First the video is downsampled to five frames per second. Most of the YouTube videos are in 30 fps, this downsampling helps to increase the efficiency of the prediction. We empirically found that five fps is the minimum fps which needs to process a video. After this video conversion the frames are extracted sequentially as images from the video and all the deepleaning models are applied on those images. All


Figure 3.12: Sports Event Detection Framework Flow Diagram

these predictions are saved into a database. After all the frames are processed, in the analysis part the predictions are accessed again and the tracking is applied to the event detections. Also a moving average is applied for activity detection. In this section the events are generated with a good accuracy.In the final stage all the generated sports events are summarised and presented to the user via a dashboard.

## 3.6 Deployment and User Interface

### 3.6.1 Introduction

In this research the complete system can be hosted in a server with API endpoints. Where the endpoint is capable of downloading the YouTube videos and extract events out of it. But that is not enough and the users can be able to access and effectively engage in this task. An API endpoint is only for the technical users who know how to work with those. We needed to expose this to all of the users without any technical background. This section covers the Hosting of the sports event detection system and its main user interface.

### 3.6.2 Technology stack

The hosting and user interface use different technologies. Docker is used to containerize all the dependencies and libraries. For the user interface we used a popular library called stream.io which is freely available for use.

Docker is an open platform for developing, shipping, and running applications. Docker enables the separation of the application from the infrastructure

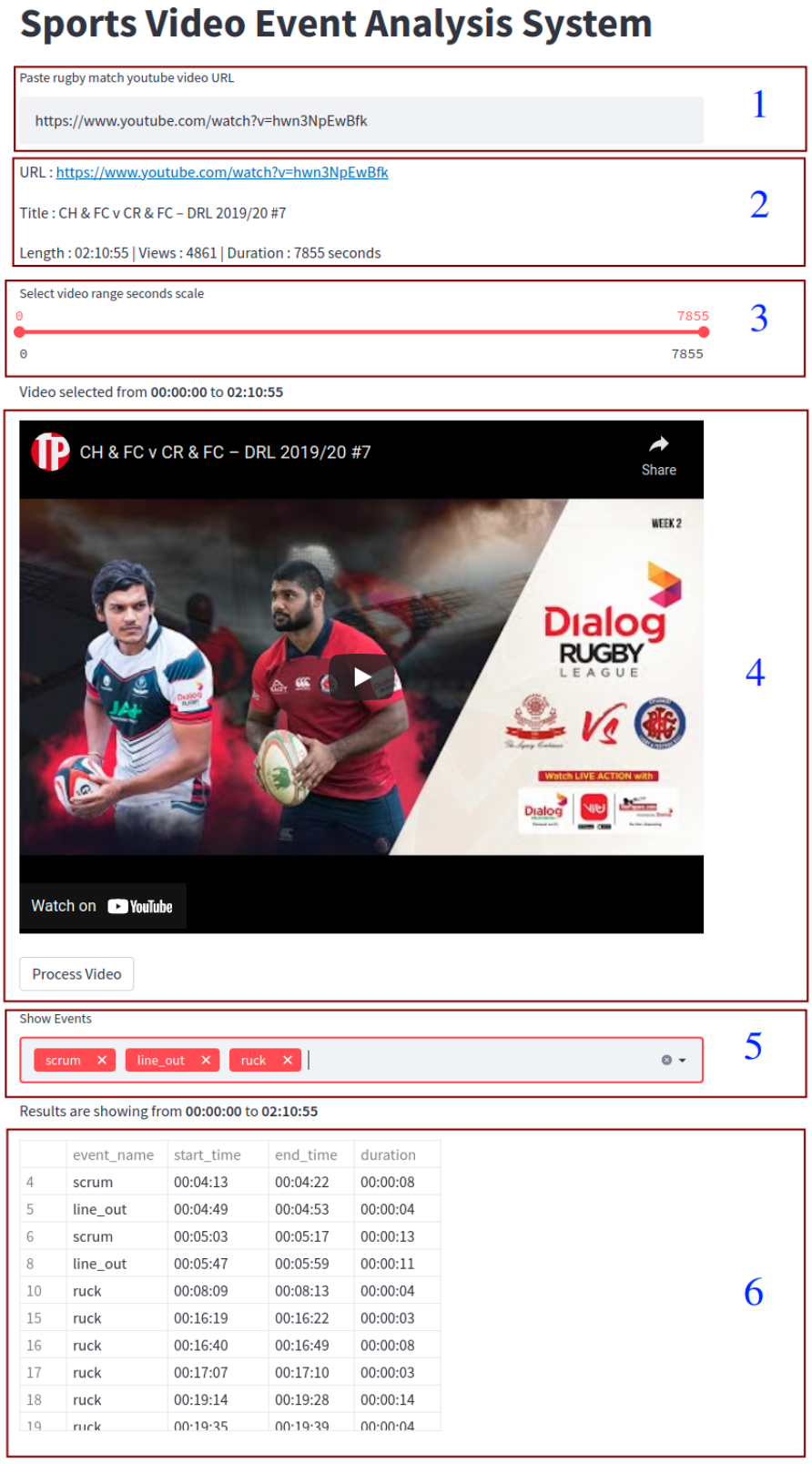
Docker is a very useful tool for devops engineers. Docker is mostly used as a platform which enables the separation of the application from the infrastructure, so anyone can deliver the application very easily. Docker is widely used for production applications, where developers develop applications and deploy thorough docker containers. Docker provides an isolated environment which has the ability to package and run an application, this is called docker container. This docker container contains everything needed to run the application, which makes it less dependent on the host machine. This is very useful when deploying the applications in the cloud where the host machine can change time to time. This project also uses docker when deploying the system in the cloud.

We didn't go with the most popular tools for Web Development that are used worldwide such as React or Angular.JS since those tools are a bit overwhelming for this project. We needed to quickly test and deploy a working solution which can be accessed by anyone. We use a library called Streamlit for this purpose. Streamlit is specially designed for the machine learning and data science community to easily create and share custom web apps using a high level python library. Streamlit was the perfect match for us. The User interface was developed using this tool. More information about the UI is explained in detail in the following section.

### 3.6.3 UI Features and Functionalities

The final UI is shown in the following figure 3.13. The main UI area is divided into different sections in order to identify the importance of each section. These sections are marked with rectangles with the relevant id. Each of these sections is discussed in detail below.

The YouTube video URL for the rugby video can be inserted here at the top of the page (section 1). This is only supported for YouTube sports videos for now. This could be easily expandable to multiple repositories in future.In section 2, the video's details will be displayed, including the title, length, and duration of the video. Using the slider in section 3 someone can select the video start time and end time duration that they are interested in. After selecting the range of the video, the button labelled "process video" (section 4) should be pressed to start the process of event detection. Additionally, there is a preview of the video material, which comes in handy when watching the video while processing. After extracting all the play actions and events the results are shown as a table (section 6). This list consists of the event position, event name, start time of the event, end time of the event and the duration of the event. Just above the results there is a separate event filter (section 5) which can filter with any event or event combinations. This is extremely beneficial if you're simply interested in viewing a single event.




| | event_name | start_time | end_time | duration |
|---|---|---|---|---|
| 4 | scrum | 00:04:13 | 00:04:22 | 00:00:08 |
| 5 | line_out | 00:04:49 | 00:04:53 | 00:00:04 |
| 6 | scrum | 00:05:03 | 00:05:17 | 00:00:13 |
| 8 | line_out | 00:05:47 | 00:05:59 | 00:00:11 |
| 10 | ruck | 00:08:09 | 00:08:13 | 00:00:04 |
| 15 | ruck | 00:16:19 | 00:16:22 | 00:00:03 |
| 16 | ruck | 00:16:40 | 00:16:49 | 00:00:08 |
| 17 | ruck | 00:17:07 | 00:17:10 | 00:00:03 |
| 18 | ruck | 00:19:14 | 00:19:28 | 00:00:14 |
| 19 | ruck | 00:19:35 | 00:19:39 | 00:00:04 |

Figure 3.13: Sports Event Detection Framework User Interface

# Chapter 4

# EVALUATION

## 4.1 Introduction

Evaluation metrics are a key component in machine learning research. There are different matrices for different tasks such as classification, regression and object detection problems. Using those metrics for performance evaluation, one can evaluate or compare the model performance with the other models which tried to do the same task. Even in production models using these evaluation metrics should enable people to improve their model's performance before deploying into the production. Without a proper metric to evaluate will eventually be a huge issue later down the line. In this research a couple of evaluation metrics used for classification models as well as object detection models. In this section, these evaluation metrics will be explained and we will present the results for our models as well.

## 4.2 Different Evaluation Metrics

### 4.2.1 Binary class classification

Binary classification is a machine learning task, it tries to classify the data into two groups or classes. These models predict a given data point into one class out of two. This is the simplest of all classifications in the machine learning domain. There is a concept called one class classification which is training only on one class which is a bit out of scope for this paper. There are many evaluation metrics when it comes to binary classification. Some of the most commonly used ones are as follows.

#### Confusion matrix

There are many different ways to measure classification accuracy. Before coming to that, we need to understand the confusion matrix, the most basic of all the metrics. Confusion matrix is a basic performance measurement technique which is widely used in machine learning problems. This is always used with classification problems with two or more than two classes. The simplest one is being a table with 2 rows and 2 columns with predicted count on one axis and actual or ground truth count on the other axis. Here is a confusion matrix for a binary classification problem.

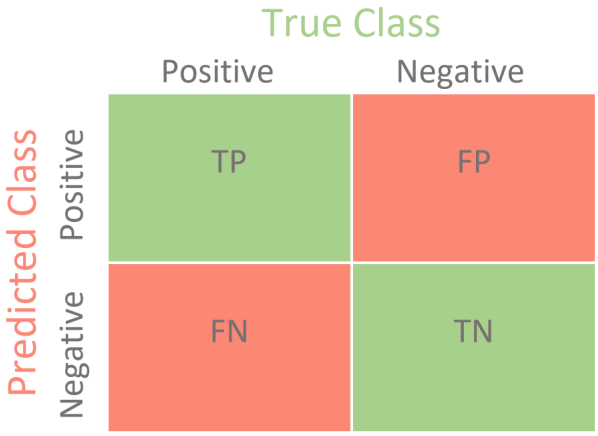



Figure 4.1: Binary Class Confusion Matrix

The interpretation of TP, FP, FN, TN is as follows.

- True Positive (TP) - The model predicted a positive value and it is true.
- True Negative (TN) - The model predicted a negative value and it is true.
- False Positive (FP) - The model predicted a positive value and it isn't true.
- False Negative (FN) - The model predicted a negative value and isn't true.

The following are the test cases executed for to verify whether the model is working properly for different scenarios. Furthermore, as depicted in Figure 4.15 and 4.16; test cases were executed in terms of validating the functionality of the Mood web application for both admin and normal user.

### Accuracy

The most common metric that is used in the machine learning domain is accuracy. Classification accuracy is the simplest among all. The accuracy of a model can be defined as the ratio between correct predictions made on the test set and total prediction.

$$Accuracy = \frac{Number\ of\ correct\ predictions}{Total\ number\ of\ predictions}$$

The error rate is the opposite of accuracy, where it is defined as the total incorrect predictions divided by all predictions which were made on the test set.

The accuracy can be derived from the confusion matrix as

$$Accuracy = \frac{TP}{TP + TN + FP + FN}$$

This metric is widely used because it's easy to interpret and easy to calculate. Also you can represent the whole model accuracy using a single number. But when the data is imbalanced this metric becomes an unreliable measure of model performance.

### Precision

Precision is calculated by total true positives divided by the total number of positive predictions. Also this can be interpreted as the quality of positive predictions made by the model. Mathematically, precision is defined as follows:

$$Precision = \frac{TP}{TP + FP}$$

### Recall

Recall is the measure of the model correctly identifying the positive cases. Recall is calculated by total true positives divided by the total number of positive ground truth cases. Mathematically, recall is defined as follows:

$$Recall = \frac{TP}{TP + FN}$$

### F1-Measure

Precision and recall values have their own definitions and values. But it's important that there is one overall value as the performance measure. So how to compare models with high precision and low recall or vice versa. F1 score comes handy in these cases, this takes the harmonic mean of precision and recall and gives an overall score. This metric uses the harmonic mean instead of arithmetic mean in order to punish the extreme values more.

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall}$$

## 4.2.2 Multiclass classification metrics

While the binary classification confusion matrix was a 2 by 2 confusion matrix which is intuitive and easy to understand. But when there are multiple classes of the confusion matrices , the larger confusion matrices can be truly confusing. Majority classification matrices that discussed before are defined for binary cases by default. Several averaging techniques are used in order to extend these binary metrics to multi-class.

### Metric calculation strategies

There are mainly two strategies to convert a multiclass problem into a series of binary tasks.

- One-vs-One (OVO).

  This method splits the original problem into a single binary classification task for each pair of classes. For example if there are 4 classes, this method uses 6 individual classifiers to binarize the problem. A major downside of this strategy is its computation workload. For this paper this method is not used for this same reason.

- One-vs-Rest (OVR).

  Alternatively, the One vs Rest operates by considering individual classifiers for each class. This is done by choosing a single class and marking it as positive label, encoding as 1. Then the rest of the classes are considered as negative labels, encoding as 0. Even though this solves the computation problem, still since one class is considered positive and the rest as negative makes each binary problem an imbalanced classification.

**Metric averaging techniques**

After converting a multi-class problem into a series of binary class classification problems, there will be multiple classification metrics for each problem. But we need a single score for each of these metrics for this multi-class problem. This is where averaging technique comes into play. There are three averaging techniques available for multi-class classification

- Macro average

  The basic arithmetic mean of all metrics across all classes will be calculated using this method. This offers all of the classes equal weights, making it a suitable choice for a balanced categorization task.

- Weighted average

  This addresses the problem of class imbalance by averaging binary metrics weighted by the number of samples in each class in the target. The weighted average will be calculated by multiplying each metric score by the number of occurrences of each class and dividing by the total number of samples.

- Micro average

  This metric is similar to the accuracy calculation, which is just summing up the diagonal values and dividing it by the total number of samples. Since accuracy is a misleading metric specially with unbalanced data, this technique is rarely used for the averaging.

### 4.2.3 Object Detection metrics

Unlike classification in object detection there are bounding boxes for annotations and predictions. Although the accuracy metrics for these tasks differ slightly from those for the classification problem, most of the concepts, such as definitions of terms including true positive, false negative, true negative and false positive, as well as accuracy, precision, and recall, can be directly applied to these tasks as well. The most commonly used metrics for evaluating object detection models are average precision (AP) and mean average precision (mAP). These metrics are used to assess the performance of well-known detection models such as Faster RCNN, Mask R CNN, YOLO, and RetinaNet. Lets check on these metrics and also fundamental concepts behind evaluating object detection models.

### Intersection over Union - IOU

In object detection, the IOU metric assesses the degree of overlap between the ground truth and prediction.These ground truths or predictions can be any shape, rectangle, box or a circle. But typically object detection is predicting rectangles, so let's consider rectangles for the examples below. The IOU definition can be written as

$$IoU = \frac{area(gt \cap pd)}{area(gt \cup pd)}$$

Diagrammatically, IoU is defined as follows

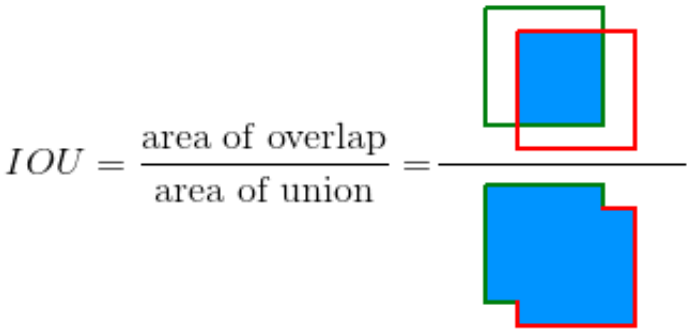



Figure 4.2: Intersection over union diagram

IOU is ranging from 0 and 1, where 0 shows no overlap between rectangles and 1 is where there is a perfect overlap between ground truth and prediction rectangles. IOU is useful through a thresholding mechanism, let's consider the IOU threshold as alpha. True positive detection is where $IoU(gt, pd) >= \alpha$ and false positive detection is where $IoU(gt, pd) < \alpha$. Consider the IoU threshold, $\alpha = 0.5$, and then define the TP, FP, and FNs as shown below.

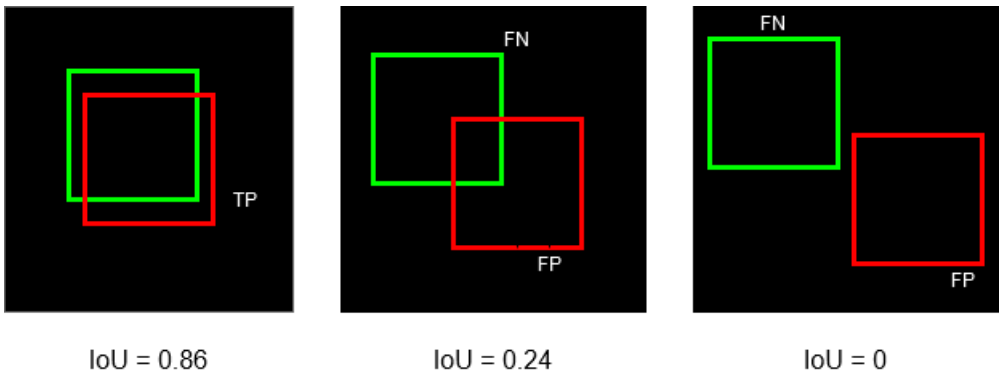



Figure 4.3: How IoU connects to TP, TN and FP in object detection

Note that here, if we increase the IoUI threshold above 0.86 the first one would be False Positive and if we reduce the threshold less than 0.24 the middle one would be a True positive.

### Precision Recall Curve

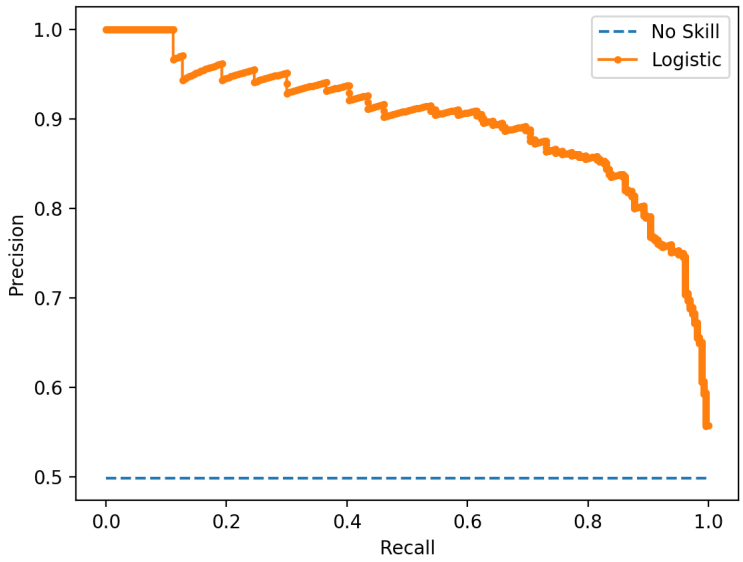



Figure 4.4: Precision Recall Curve for Logistic Classification

Confidence score for each prediction also plays an important role in these metrics. Raising confidence means that the model will miss more objects, resulting in a low recall and high precision, whereas low confidence means that there will be many predictions, resulting in more False Positives and a low precision and high recall. This implies that there is some sort of trade-off between precision and recall. The precision-recall (PR) curve depicts the relationship between precision and recall at various levels of confidence. Precision and recall for a good model remain high even when the confidence score varies.

### Average Precision

Average Precision is the area under the PR curve. This is measured in relation to the alpha IoU threshold. Which can be formally defined as

$$AP@\alpha = \int_0^1 p(r)dr$$

A high value for the Average Precision means it was able to maintain high precision and recall for a given IoU threshold which is obviously a better model overall. Naturally the PR curve is not a monotonically decreasing chart. This property can be removed using interpolation methods, and there are mainly two methods as follows.

1. Eleven(11) point interpolation

A chart that interpolates precision scores for 11 equally spaced standard recall levels is known as an 11-point AP. The recall values are as 0.0, 0.1, 0.2 ... 1.0. The 11AP is defined as

$$AP@\alpha_{11} = \frac{1}{11} \sum_{r \in R} P_{interp}(r),$$

where, R = 0.0, 0.1, 0.2, ... 1.0 and

$$P_{interp}(r) = maxp(r')$$

2. All-point interpolation

In this situation, interpolation is performed for all places (recall values), i.e.

$$AP@\alpha = \sum_{i} (r_{i+1} - r_i) P_{interp}(r_{i+1}),$$

$$where, P_{interp}(r_{i+1}) = maxp(r')$$

**Mean Average Precision (mAP)**

Each class's average precision is calculated separately. This ensures that the dataset has an Average Precision for each and every class. The measurement, mean Average Precision, is calculated by averaging these AP values. To be more specific, mean Average Precision (mAP) is the average of AP values over all classes.

$$AP@\alpha = \sum_{i=1}^{n} AP_i \quad for\ n\ classes,$$

As same as AP, mAp is also calculated at a given IoU threshold alpha. mAp@0.5 is the mean Average Precision at 0.5 IoU threshold. mAP can also be calculated over a variety of thresholds; for example, in certain circumstances, AP is computed for a given category/class at ten different IoU range from 50% to 95% at 5% step-size, and then the mean across classes is taken; this is usually expressed by mAP@[.50:.5:.95].

## 4.3 Model Evaluation

In this part, the main focus is to evaluate trained models using the above discussed model evaluation matrices. Model evaluation is an important step and it is the fundamental key to modelling success. The modele valuation helps us to interpret various evaluation metrics to understand our model's strengths and weaknesses leading us to model improvement hints. Also it will give us a good indication of how well the model is trained and how well it performs for the unseen data.

### 4.3.1 Evaluating Classification Models

In this research there is one classification model involved. Which is related to activity detection. The main duty of that model is to classify a given image to play or no-play classes. This is a relatively easy task for a classification model. The images provided for this task are taken from the same video. The play images are taken where the actual playing moments and the no-play images are taken from the collection of advertisements, out of ground camera views. The play moment overlays are not added to this dataset. So in this dataset there is a clear separation of play and no-play, there is less ambiguity between those two things. Let's look at some of the model's training and validation curves.

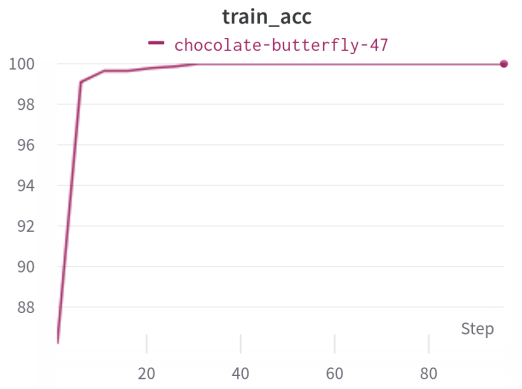



Figure 4.5: Training Accuracy

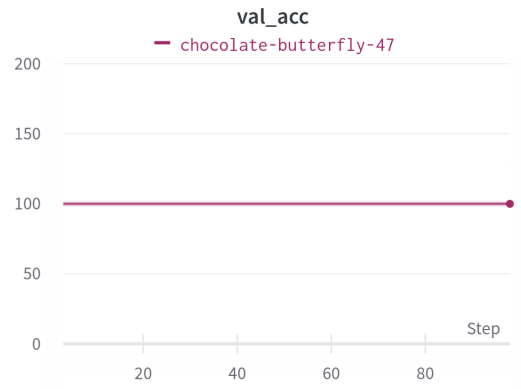



Figure 4.6: Validation Accuracy

The above graph depicts the classification model's training and validation accuracy. Since this problem is very easy compared to the detection problem, within a small number of epochs the model reached almost 100% accuracy. This happened with many types of classification models. Because of this reason the resnet 18 was selected based on the computational efficiency of the model.

### 4.3.2 Evaluating Object Detection Models

There are two object detection modes trained in this research one is for the digital banner detection model and the other is event detection model. Both of the models are trained on the latest YOLOv5 model. The Evaluation metric graphs for those models are as follows.

1. Digital overlay Detection Model

   The above graph shows the recall and precision values of the digital overlay detection model. As you can see in the graphs the accuracy and precision values are not that bad. MAP 0.5 almost hit 100% while mAP0.5:0.95 exceeded 0.8 which is a very good value for the localization ability of an object detector.

   The metrics are over 95% except mAP0.5:0.95, which is really a hard metrix to improve on.

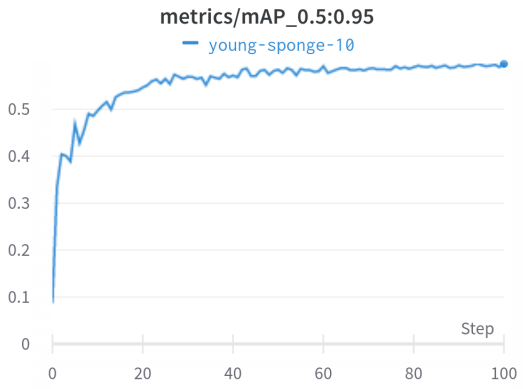



Figure 4.7: AP@0.5

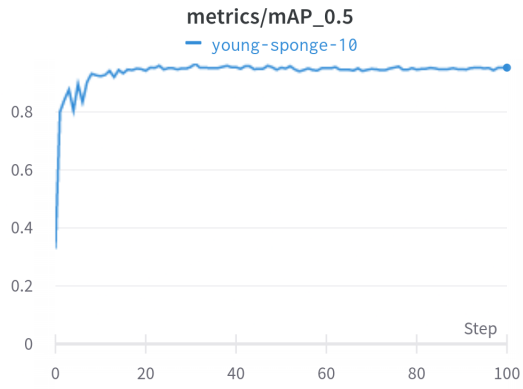



Figure 4.8: AP@0.5:0.95

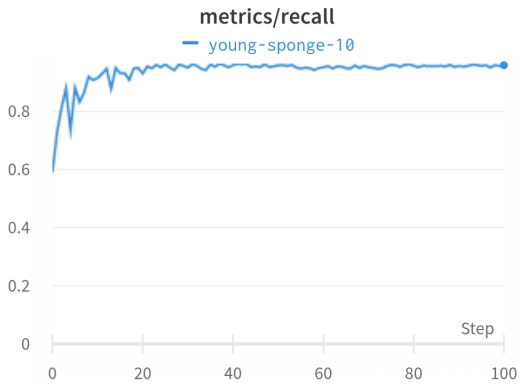



Figure 4.9: Recall Curve

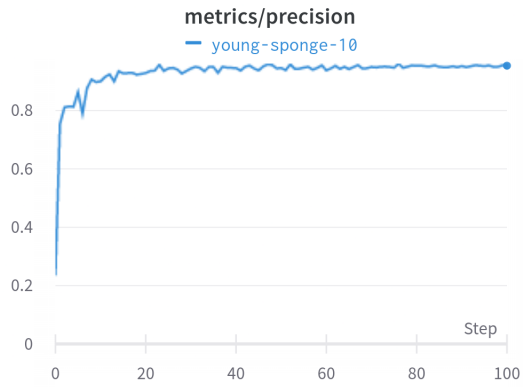



Figure 4.10: Precision Curve

2. Event Detection Model

These graphs are for the event detection model. The values are for the mean of three initial events (ruck, lineout and scrum) which are considered for this project. The final values after 100 epcoches are as follows.

In the end of the training the mAP0.5 was at 0.95 and mAP0.5:0.95 was at 0.59, the important metric here is mAP0.5, which says that it was able to detect the events with at least 0.5 IOU for 95% of the time. So the detection is very good but the downside is that mAp0.5:0.95 is in the 0.59 range. This metric is mostly used to measure the localization accuracy. In this research spatial localization is not a priority. The main objective of the project is to find the temporal localization of the events.

## 4.4 Object Tracking

Object tracking is the method that is used after the object detection for tracking the same object between frames. Here the main reason for an object detection is to uniquely identify the events across frames. The exact location of the event is not a valuable information in this project but it's important that the event is tracked across

| Metric | Epoch 10 | Epoch 20 | Epoch 50 | Epoch 100 |
|---|---|---|---|---|
| Recall | 0.93333 | 0.94822 | 0.98485 | 0.99394 |
| Precision | 0.80624 | 0.96135 | 0.97887 | 0.97342 |
| mAP_0.5 | 0.83117 | 0.95856 | 0.99170 | 0.99132 |
| mAP_0.5:0.95 | 0.39336 | 0.58883 | 0.79742 | 0.84775 |

Table 4.1: Digital overlay detection - Training metrics

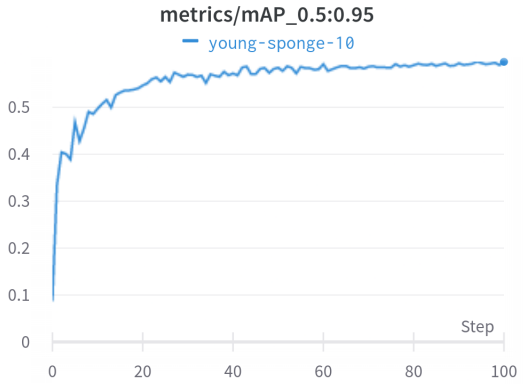



Figure 4.11: mAP@0.5

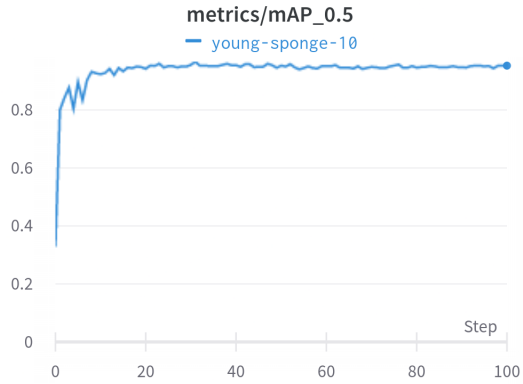



Figure 4.12: mAP@0.5:0.95

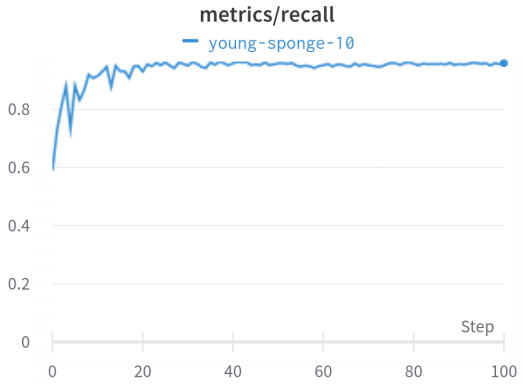



Figure 4.13: Recall Curve

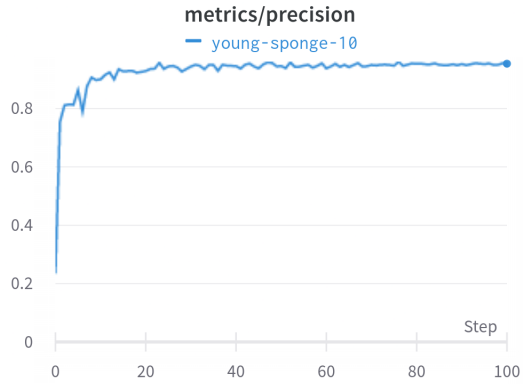



Figure 4.14: Precision Curve

the frames to determine the event's temporal position in the video segment. The object tracking accuracy hugely depends on the object detection accuracy. If the detection fails often the tracking also fails due to lack of tracking information.

There are many types of multiple object tracking evaluation methods available. In this research our main focus will be on two matrices.

1. MOTP (Multiple Object Tracking Precision)

   This mainly measures the localization accuracy of detected bounding boxes. This is mostly dependent on the localization of the object detection models and how that localization is efficiently exploited by the tracking algorithm to achieve even better localization accuracy.

| Metric | Epoch 10 | Epoch 20 | Epoch 50 | Epoch 100 |
|---|---|---|---|---|
| Recall | 0.91966 | 0.94801 | 0.95273 | 0.95577 |
| Precision | 0.90611 | 0.92255 | 0.95592 | 0.95473 |
| mAP_0.5 | 0.93186 | 0.94969 | 0.95421 | 0.95255 |
| mAP_0.5:0.95 | 0.53764 | 0.54621 | 0.57309 | 0.58997 |

Table 4.2: Sports event detection - Training metrics

This is calculated in a fairly easy way. The formula is as follows.

$$MOTP = \frac{\sum_{i,t} d_t^i}{\sum_t c_t}$$

$d_t$ - This is the distance measure (IOU / euclidean distance) between ground truth object and detection output

$c_t$ - This is the total matches between ground truth and detection output

2. MOTA (Multiple Object Tracking Accuracy)

This measures the overall accuracy of both the tracker and the detection. Three errors are taken into consideration when calculating this accuracy.

- **Miss Detection ($FN_t$)** - Object present in the real world but not detected by the detection method.
- **False Positive ($FP_t$)** - Object not existing in the ground truth, but detected as such by the detection model.
- **Mismatch Error ($IDS_t$)** - Due to erroneous tracking, an object in ground truth is incorrectly associated with another object.

The formula is as follows.

$$MOTP = 1 - \frac{\sum_t (FN_t + FP_t + IDS_t)}{\sum_t GT_t}$$

$\sum_t GT_t$ - total ground-truth object count

This project uses two tracking methods. A moving average tracking method for the activity detection and a bounding box tracking method for event tracking. But in both cases the observation is there wont be more than one event in a given situation. These events are occurring one after another. There are some rare instances where there were two rucks on the same frame, where the older ruck was still visible but not that significant. So when comes to the event detection there are no multiple objects in the same frame. Also the bounding box is not going to alter by the tracking method since the spacial location is not important for this research. As a result of these factors, the methods for evaluating tracking described above are ineffective. The tracking accuracy in this project may be a direct reflection of the object detection model's performance.

## 4.5 Overall Event Detection Evaluation

In this section the main objective is to get an overall accuracy value for the entire event detection. At the end of the day the user input is a pointer to a video file and the output is all the events extracted from that. events relevant to the video segment. The overall accuracy in this case is how accurate the system captured the events in the video segment. This is the objective of this system so the overall accuracy.

To get the overall accuracy, a number of separate videos were processed using the full system and compared with the actual events. If the ground truth and predicted events have an overlap with the correct event tag this is considered as a true positive event. False positives is an event which is predicted without any overlap or with a different event tag. False negative where there is a ground truth event but is not predicted by the system. And there is no definition to the True Negative event. Precision, Recall and F1 score was calculated from these values.

The overall accuracy calculated for following videos

1. Army SC v Kandy SC – DRL 2019/20 #38 - YouTube Link

| Event | TP | FP | FN | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| Scrum | 19 | 7 | 2 | 0.73 | 0.90 | 0.81 |
| Lineout | 12 | 23 | 6 | 0.29 | 0.34 | 0.45 |
| Ruck | 76 | 67 | 16 | 0.53 | 0.83 | 0.65 |
| Total | 107 | 97 | 24 | 0.52 | 0.81 | 0.63 |

Table 4.3: Overall Event Detection Accuracy Video 1

2. Navy SC v Kandy SC – DRL 2019/20 #43 - YouTube Link

| Event | TP | FP | FN | Precision | Recall | F1 |
|---|---|---|---|---|---|---|
| Scrum | 10 | 10 | 1 | 0.50 | 0.91 | 0.65 |
| Lineout | 12 | 33 | 3 | 0.27 | 0.80 | 0.40 |
| Ruck | 72 | 52 | 15 | 0.58 | 0.83 | 0.68 |
| Total | 94 | 95 | 19 | 0.50 | 0.84 | 0.63 |

Table 4.4: Overall Event Detection Accuracy Video 2

The accuracy was not taken as an evaluation metric because precision and recall better represent the values. The overall performance was measured using F1 score which is the harmonic mean of precision and recall. The average value of F1 was 0.63 for both the videos. When observing the values, recall has a comparatively higher value. This means that most of the events were captured properly. But the recall values are around 50%. This means that the captured events are not accurate and there are many false positives. Taking these results further, the observation was that these misclassifications are mainly due to the nature of the broadcast video. When the scene is switched between multiple cameras, the tracking fails. Then there are more events

generated under those circumstances. This accounts for the most false positives. Then some events are there for an extended period of time, which could generate multiple events because of the classification error. Furthermore, there were moments where the play was stopped by the referee, but the video contains the ground and players. These clips are very hard to distinguish from the active play. This also accounted for the low accuracy. As a fresh framework implementation, this is an acceptable value, but there is a need for improvement in terms of accuracy.

## 4.6 Comparison with other sports event detection systems

Comparing the Rugby event detection framework with other sports-related research is the main goal of this section. Overall, a substantial number of event detection systems were developed for other sports, but no specific research was conducted for rugby. Most event detection technologies were created for soccer because it is the most popular sport in the world. Rugby and soccer, however, differ significantly from one another. Soccer is not a contact sport, although rugby is. Rugby is a complex sport as players constantly block one another and the ball is often hidden. It is challenging to precisely compare event detection, specifically on Rugby, with other systems due to these significant variation.

The primary rationale for choosing the studies for the comparison was recent research, which is primarily based on deep learning techniques. Four potential research publications examined on the subject of sports event detection. The following paragraphs discuss the comparisons.

In the research Event Detection Based Approach for Soccer Video Summarization Using Machine Learning, Hossam M. Zawbaa et al. presents an event detection and summarization system based on machine learning (ML) for highlighting significant occurrences during soccer matches. The suggested method breaks down the whole video stream into discrete video shots before classifying the resulting shots into various shot-type categories. After that, the system applies two machine learning methods, support vector machine (SVM) and artificial neural network (ANN), to emphasize important segments with logo appearances and detect caption regions that provide game score information. Afterwards, the system recognizes vertical goal posts and goal nets. In the final soccer video summary, the key moments from the match are highlighted.

Three events; Goal event, Attack event, and Other events are presented in this study when compared to the rugby event detection framework. While this presented study employed an end-to-end deep learning network to detect the events, they have used a state machine with information from other events to decide these events. Compared to the goal and attack events in the previous research, the events in this research are significantly more complicated. A state machine cannot decide these sophisticated occurrences. A deep neural network has been chosen to detect these occurrences as a result

"Goal Event Detection in Sports Video" is another framework for event detection presented by Grigorios Tsagkatakis et al. The study's goal was to comprehend the

complex events depicted in the unstructured videos. Through the use of convolutional neural networks and feature fusion via regularized autoencoders, they have independently encoded spatial and temporal data.

The proposed goal detection framework is depicted in the figure C.1. A 20-frame moving window initially chooses the interesting portion of the sequence. Then, to extract spatial and temporal information, raw pixel values and optical flows are encoded using pre-trained deep CNNs. Then those features can be fused with higher level networks for classification problems.

The developed event detection framework is directly related to this study. They included spatial and temporal features for the mode, however my research used only special features for the model and then combined these features via the time domain using a tracking algorithm. The most widely used and effective solution for most detection problems is object detection. This is the major justification behind choosing object detection over a custom network for this issue.

In 2018, Abdullah Khan et al's study on Soccer event detection, he suggested a framework that first detects objects from each individual video frame and then presents a set of candidate objects with corresponding confidence scores. The event detection system then uses rules based on temporal and logical combinations of the detected items and their relative distances to identify events. The table C.3 provides the event detection accuracy.

Although the object detection employed in this research is identical to the work I provided here, it is used different method to detect the humans and the ball than it was in my work. I have directly identified the event using object detection. In contrast to this approach, where event detection occurs in two steps; first detecting the objects and then generating the event based on those objects and their proximity. However, the event detection in my framework is one unified network. It's always preferable to integrate everything into a single network as then training can be completed in a single run without adding a rule based system between different predictions.

Ali Karimi et al, proposed an event detection pipeline that uses both machine learning and deep learning in their study "Soccer Event Detection Using Deep Learning." In order to train the specified networks, they also constructed a dataset containing 10 UEFA Champions League games. The proposed algorithm's general block diagram is shown below.

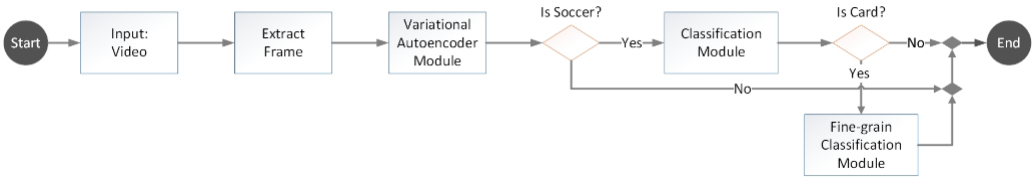



Figure 4.15: Generic block diagram of the proposed algorithm - Ali Karimi et al

To determine if the scene contains an event or not, they have used Variational Auto Encoders. Then, the classification module will divide the image into nine classes. The image is once again sent to the Fine-grained classification module to discover the color

of the card if the event is a card event.

In this study, event detection was primarily performed by classification, and events were then tracked through frames utilizing only event frames. In my method, I used object detection to locate the events and monitored them using spatial data as well. The approach I mentioned could capture two simultaneous occurrences without any problems, but in this research, there is no way to observe such events. With the current implementation, the event detection system is capable of multiple object detection and tracking.

Comparing to related research, Rugby Event Detection proposed new method of detecting events. Sports event detection using object detection is an area which is never explored before. Combining this with activity detection using both state of the art classification network and object detection network is novel to this domain. Also rugby is a relatively complex sport when compared with the soccer which most of the researches are focused on. These methods can easily apply for other sports like soccer with the hold of the proper dataset. There were no datasets available for event detection which is annotated for object detection, which makes hard to get a accuracy comparison with other methods.

# Chapter 5

# DISCUSSION & CONCLUSION

## 5.1 Discussion

Sports event detection is an interesting task in the field of sports video analysis, and solving it will pave the way for many future improvement. This use case can be utilized for a variety of tasks, including coaching new players, examining the teams' game plans, summarizing the entire match, and analyzing the teams' strengths and weaknesses. These technological innovations will benefit players, coaches, and spectators.

This project presents a novel approach to event detection in a sports video with a fully functional end-to-end framework that has been successfully deployed for everyone to use. Rugby event detection is a relatively new area in the event detection domain. Hence a dataset for a rugby event was produced as part of this research. The temporal positions of more than 2000 occurrences from 40 videos are contained in this dataset. Additionally, this provides annotations for event bounding boxes for more than 5000 images. The "Rugby Event Dataset" was made accessible along with this study for those who are interested.

The primary research area was generally divided into two sections. The first step is identifying sporting events in the video, and the second is identifying sporting activities in the clip. The accuracy of the action detection is increased up to 0.9 f1 score using a simple implementation strategy that combines a classification model with an object recognition model to identify digital overlays. This research has demonstrated that object detection models that easily utilized to identify events in team sports such as rugby, which is important for the identification of sporting events. With mAP0.5 above 0.95, which represents detection accuracy, outstanding results for Sports Event identification were obtained using the state of the art object detection models. Furthermore, the localization accuracy is reasonable at 0.59 mAP0.5:0.95. It has been demonstrated that the accuracy of event localization and detection can be increased when this detection and tracking are combined.

Finally, the sports event detection framework was implemented with end to end pipeline. Non-technical individuals can easily access this pipeline and host this on any cloud service. When the entire system was merged and evaluated for the broadcast videos, the f1 score was about 0.63, which is on the lower end, but given that this is one of the first event detection systems for rugby sports, there will undoubtedly be

more improvements in the future.

This project's entire source code is accessible to everyone at: https://github.com/CodeProcessor/sports-events-detection.

## 5.2 Future Work

The major goal of this study is to develop a technique for identifying important moments in a sports video. The primary goal was successfully met with reasonable accuracy. Currently this project is mainly aimed for the Sri Lanka Dialog Rugby League, although it is easily extendable to international rugby with the appropriate dataset, and further events may also be recorded and trained using the same architecture. Finally, this concept can be expanded to can be applicable for the sports such as cricket, football, baseball, and basketball.

## 5.3 Conclusion

In the world of sports, the detection of sporting events is an important yet largely unexplored topic. There aren't many publicly accessible datasets in this field for the same reason. With this study, I have released a brand-new dataset called the Rugby Event Dataset. This report outlines an innovative technique for locating sporting events and activity in a rugby broadcast video using only visual elements. A framework for locating sporting events was also developed and released as part of this study. I anticipate that this study would help and inspire other scholars to carry out more studies in the area of sports event detection.

# REFERENCES


[1] Mark Baillie and Joemon M Jose. Audio-based event detection for sports video. In *International Conference on Image and Video Retrieval*, pages 300–309. Springer, 2003.

[2] Marc-André Carbonneau, Alexandre J Raymond, Eric Granger, and Ghyslain Gagnon. Real-time visual play-break detection in sport events using a context descriptor. In *2015 IEEE International Symposium on Circuits and Systems (ISCAS)*, pages 2808–2811. IEEE, 2015.

[3] François Chollet. Xception: Deep learning with depthwise separable convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1251–1258, 2017.

[4] Samuel F de Sousa, Arnaldo de A Araújo, and David Menotti. An overview of automatic event detection in soccer matches. In *2011 IEEE Workshop on Applications of Computer Vision (WACV)*, pages 31–38. IEEE, 2011.

[5] Carl Doersch. Tutorial on variational autoencoders. *arXiv preprint arXiv:1606.05908*, 2016.

[6] Ahmet Ekin and M Tekalp. Generic play-break event detection for summarization and hierarchical sports video analysis. In *2003 International Conference on Multimedia and Expo. ICME'03. Proceedings (Cat. No. 03TH8698)*, volume 1, pages I–169. IEEE, 2003.

[7] Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015.

[8] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014.

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE transactions on pattern analysis and machine intelligence*, 37(9):1904–1916, 2015.

[10] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[11] Ali Karimi, Ramin Toosi, and Mohammad Ali Akhaee. Soccer event detection using deep learning. *arXiv preprint arXiv:2102.04331*, 2021.

[12] Abdullah Khan, Beatrice Lazzerini, Gaetano Calabrese, and Luciano Serafini. Soccer event detection. In *4th international conference on image processing and pattern recognition (IPPR 2018)*, pages 119–129. AIRCC Publishing Corporation, 2018.

[13] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, 25:1097–1105, 2012.

[14] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

[15] Baoxin Li and M Ibrahim Sezan. Event detection and summarization in sports video. In *Proceedings IEEE Workshop on Content-Based Access of Image and Video Libraries (CBAIVL 2001)*, pages 132–138. IEEE, 2001.

[16] Yongqiang Li, S Mohammad Mavadati, Mohammad H Mahoor, Yongping Zhao, and Qiang Ji. Measuring the intensity of spontaneous facial action units with dynamic bayesian network. *Pattern Recognition*, 48(11):3417–3427, 2015.

[17] Tsung-Yi Lin, Piotr Dollár, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2117–2125, 2017.

[18] Li Liu, Ling Shao, Xuelong Li, and Ke Lu. Learning spatio-temporal representations for action recognition: A genetic programming approach. *IEEE transactions on cybernetics*, 46(1):158–170, 2015.

[19] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. Ssd: Single shot multibox detector. In *European conference on computer vision*, pages 21–37. Springer, 2016.

[20] Li Lu, Fengpei Ge, Qingwei Zhao, and Yonghong Yan. A svm-based audio event detection system. In *2010 International Conference on Electrical and Control Engineering*, pages 292–295. IEEE, 2010.

[21] Mahdyar Ravanbakhsh, Moin Nabi, Enver Sangineto, Lucio Marcenaro, Carlo Regazzoni, and Nicu Sebe. Abnormal event detection in videos using generative adversarial nets. In *2017 IEEE International Conference on Image Processing (ICIP)*, pages 1577–1581. IEEE, 2017.

[22] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016.

[23] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015.

[24] A Salunke, Sonal Deshmukh, and Ratnadeep Deshmukh. Event detection in sports video game -a review. 01 2010.

[25] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[26] Nitish Srivastava, Elman Mansimov, and Ruslan Salakhudinov. Unsupervised learning of video representations using lstms. In *International conference on machine learning*, pages 843–852. PMLR, 2015.

[27] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-first AAAI conference on artificial intelligence*, 2017.

[28] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[29] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2818–2826, 2016.

[30] Grigorios Tsagkatakis, Mustafa Jaber, and Panagiotis Tsakalides. Goal!! event detection in sports video. *Electronic Imaging*, 2017(16):15–20, 2017.

[31] Gül Varol, Ivan Laptev, and Cordelia Schmid. Long-term temporal convolutions for action recognition. *IEEE transactions on pattern analysis and machine intelligence*, 40(6):1510–1517, 2017.

[32] Jinjun Wang, Changsheng Xu, Engsiong Chng, Xinguo Yu, and Qi Tian. Event detection based on non-broadcast sports video. In *2004 International Conference on Image Processing, 2004. ICIP'04.*, volume 3, pages 1637–1640. IEEE, 2004.

[33] Lexing Xie, Shih-Fu Chang, Ajay Divakaran, and Huifang Sun. Structure analysis of soccer video with hidden markov models. In *2002 IEEE International Conference on Acoustics, Speech, and Signal Processing*, volume 4, pages IV–4096. IEEE, 2002.

[34] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1492–1500, 2017.

[35] Changsheng Xu, Jinjun Wang, Kongwah Wan, Yiqun Li, and Lingyu Duan. Live sports event detection based on broadcast video and web-casting text. In *Proceedings of the 14th ACM international conference on Multimedia*, pages 221–230, 2006.

[36] Gu Xu, Yu-Fei Ma, Hong-Jiang Zhang, and Shiqiang Yang. A hmm based semantic analysis framework for sports game event detection. In *Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429)*, volume 1, pages I–25. IEEE, 2003.

[37] Min Xu, Namunu C Maddage, Changsheng Xu, Mohan Kankanhalli, and Qi Tian. Creating audio keywords for event detection in soccer video. In *2003 International Conference on Multimedia and Expo. ICME'03. Proceedings (Cat. No. 03TH8698)*, volume 2, pages II–281. IEEE, 2003.

[38] Hossam M Zawbaa, Nashwa El-Bendary, Aboul Ella Hassanien, and Tai-hoon Kim. Event detection based approach for soccer video summarization using machine learning. *International Journal of Multimedia and Ubiquitous Engineering*, 7(2):63–80, 2012.

# APPENDICES

# Appendix A

# Rugby Events

Below figures shows what are the rugby events


Figure A.1: Rugby Lineout

Figure A.2: Rugby Scrum

Figure A.3: Rugby Ruck

# Appendix B

# Training and Validation Results

## B.1 Digital Overlay Detection Model

Figure B.1: Training Batch 1


Figure B.2: Validation Batch 1 - Labels


Figure B.3: Validation Batch 1 - Predictions


Figure B.4: Validation Batch 2 - Labels


Figure B.5: Validation Batch 2 - Predictions


Figure B.6: Validation Batch 3 - Labels


Figure B.7: Validation Batch 3 - Predictions

## B.2 Event Detection Detection Model

Figure B.8: Training Batch 1


Figure B.9: Validation Batch 1 - Labels

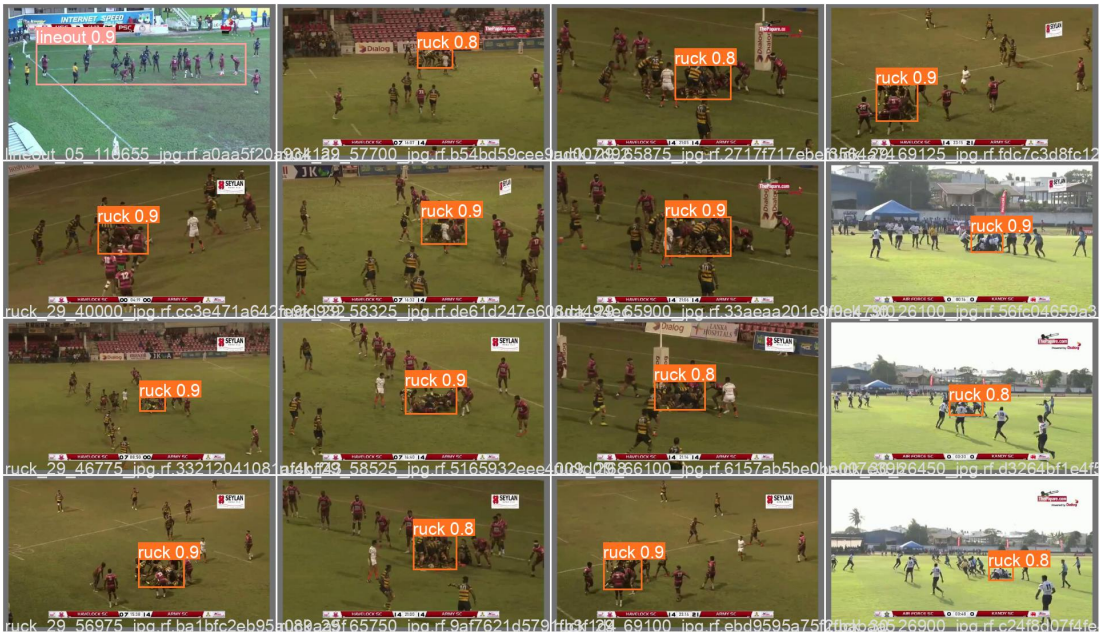



Figure B.10: Validation Batch 1 - Predictions

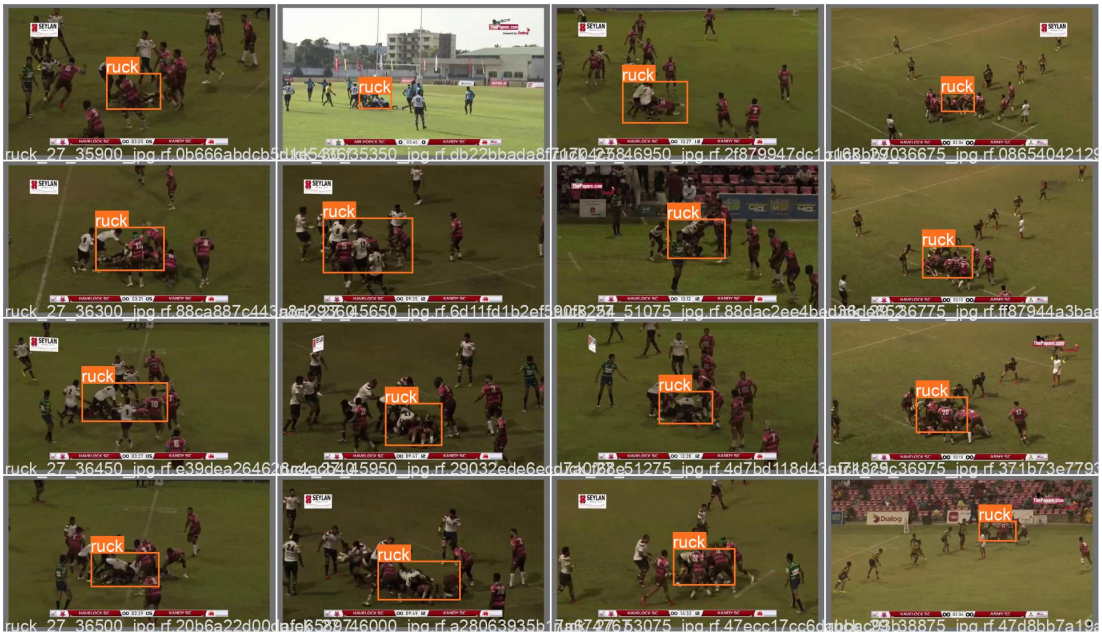



Figure B.11: Validation Batch 2 - Labels


Figure B.12: Validation Batch 2 - Predictions


Figure B.13: Validation Batch 3 - Labels


Figure B.14: Validation Batch 3 - Predictions

# Appendix C

# Results Comparison

| Event Detection | Goal | Attack | Other Events |
|---|---|---|---|
| Goal | 57 | 3 | 0 |
| Attack | 6 | 176 | 8 |
| Other Events | 0 | 18 | 283 |
| Recall | 95.0% | 92.6% | 94.0% |
| Precision | 90.5% | 89.0% | 97.3% |

Table C.1: Confusion matrix for event detection and summarization - Hossam M. Zawbaa et al

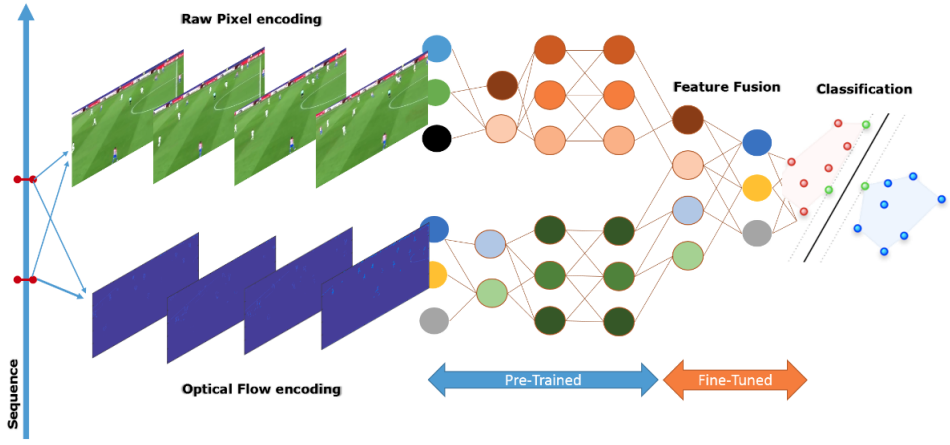



Figure C.1: Block diagram of the proposed Goal detection framework.

| Method | Linear | Gaussian | Polynomial |
|---|---|---|---|
| Spacial | 0.16 | 0.12 | 0.11 |
| Spat. & Temp. | 0.14 | 0.10 | 0.09 |
| Fused | 0.45 | 0.02 | 0.03 |

Table C.2: Classification error for different combination of classification kernel and features - Grigorios Tsagkatakis et al

| Detected Events | Total | Detected | Miss | Accuracy |
|---|---|---|---|---|
| Ball Possession | 14 | 13 | 1 | 92% |
| Kicking the ball | 19 | 16 | 3 | 84% |

Table C.3: Event Detection Results - Abdullah Khan et al