UNIVERSITY OF OSLO

Master's thesis

TACDEC: Developing Automatic AI-Based Tackle Detection for Soccer

Evan Jåsund Kassab

Informatics: Robotics and Intelligent Systems
60 ECTS study points

Department of Informatics
Faculty of Mathematics and Natural Sciences

Spring 2024

Evan Jåsund Kassab

TACDEC: Developing Automatic AI-Based Tackle Detection for Soccer

Supervisors:

Kyrre Glette, Pål Halvorsen

Abstract

This thesis presents the development and evaluation of automated AI-based models for detecting soccer tackles from broadcast video. The study introduces a novel dataset curated specifically for this task, consisting of annotated soccer videos from the Norwegian Eliteserien, focusing on differentiating between tackles occurring live and for replay scenarios. Two primary approaches were explored using spatial features extracted with DINOv2: one using stand-alone spatial features with ensemble learning and a temporal analysis through the TempTAC model, which incorporates LSTM networks to leverage temporal dynamics. Extensive experiments demonstrated that manipulating frame presentation (e.g., stretching, padding) before feature extraction can significantly impact model accuracy. The models achieved high performance in identifying the different types of tackles, with particular improvements noted when temporal features were considered. This research not only advances the automatic detection of soccer tackles but also contributes a valuable dataset to the sports analytics community, potentially aiding further developments in broadcast technology and tactical sports analysis.

Acknowledgments

I would like to extend a special thanks to Cise Midoglu and Pål Halvorsen for their invaluable guidance and support. With their thoughtful feedback and advice, this thesis was shaped with me at the helm.

I also wish to express my gratitude to Michael Riegler for sharing his insights and providing consistent feedback throughout this journey.

Additionally, a special thanks goes out to Chelsea FC and Cole Palmer for offering me some much-needed distraction during their somewhat turbulent season.

AI Declaration

In this thesis, Generative AI has been used through tools like GPT UiO, which is a GPT model by OpenAI but specialized to the University of Oslo to ensure that the privacy set by the university is complied with [23]. Grammarly, which also offers a premium service that uses Generative AI has been used in this work [54].

These tools have been used exclusively for:

- Rephrasing
- Debugging LaTeX
- Suggesting synonyms
- Grammatical questions

Neither of these nor any other tools have been used for any form of content generation or creation involving text or code.

The only thing that has been created and used using Generative AI in this thesis is the cover image on the front page. This image is used using Midjourney, a text-to-image generator [43].

The image was generated using the following prompt:

Create an arbitrary drawing of a Chelsea soccer player slide tackling another Manchester United soccer player to obtain the ball –ar 2:1

Contents

List of Figures

List of Tables

Acronyms

3D-CNN 3D Convolutional Neural Network.

API Application Programming Interface.

CNN Convolutional Neural Network.

CV Computer Vision.

EPL English Premier League.

FN False Negative.

FNN Feedforward Neural Network.

FP False Positive.

FPS Frames Per Second.

IFAB International Football Association Board.

LSTM Long Short-Term Memory.

MAC Multiply-accumulate.

ML Machine Learning.

NN Neural Network.

RNN Recurrent Neural Network.

SOTA State of the art.

TP True Positive.

UK United Kingdom.

VAR Video Assistant Referee.

ViT Vision Transformer.

Chapter 1

Introduction

In this chapter, central questions and objectives of this thesis will be presented. The motivation behind the research will be discussed, and the problem statement defined. Additionally, the scope of the study will be delineated, and the research methodology described. Ethical considerations surrounding the work will also be addressed, along with highlights of the contributions of this thesis. Finally, an outline of the subsequent chapters will be provided.

1.1 Motivation

Association football, also known as soccer or just football, is the biggest sport in the world [66, 73]. With more than 1.5 billion watching the World Cup 2022 on live television, it is still surprising that the aftermath of digital content reached a staggering 5 billion engagements [48]. With content creation and sharing easier than ever, combined with the overuse of smartphones [8] and the prediction of AI generating better and more personalized highlights [55], diversity is crucial.

Videos have now been a part of our lives for over a century [34], capturing the essence of memorable moments and becoming an integral part of our entertainment. In sports coverage and analysis, the demand for highlight content has soared to new heights [58]. Soccer fans and supporters yearn for more than traditional events like goals, penalties, and cards that have already been extensively researched [6, 19, 46, 47, 75]. They crave a deeper connection with the game, seeking to uncover the hidden gems, like the tackles, the mistakes, and the big chances that often go unnoticed [67].

Imagine a world where, within seconds of a tackle being made on the field, fans can relive the moment through instant replays, gaining insights and appreciating the details of the game. This is the essence of the commercial potential within automated event detection in sports videos.

The current landscape of sports content poses a considerable challenge. While key events such as goals and penalties are carefully annotated and easily accessible to viewers, many moments that could greatly enhance fan engagement are largely overlooked. These moments include heated moments, errors leading to goals, skilful dribbles, and the complex dynamics of tackles, all of which have the potential to generate substantial fan interest and debate. The exclusion of these events from highlight reels and social media content reduces the richness of the sports viewing experience and misses valuable opportunities for deeper fan interaction and engagement. Therefore, the challenge involves the technical ability to automatically identify and catalogue these events from raw footage and ensure such diverse and engaging moments are easily accessible to fans, enriching their connection to the game and each other.

1.2 Problem Statement

As stated above, traditional event annotations in soccer videos, such as goals, penalties, and cards, have received considerable attention in manual and automated approaches. However, there exists additional content that is perceived as valuable, including big chances missed, last-minute tackles, and game-altering mistakes. These unofficial events contribute significantly to the game's dynamics, but have not been extensively studied in terms of automated annotation. This thesis investigates methods for automatically annotating these unofficial events in untrimmed soccer videos, enhancing the understanding and analysis of gameplay dynamics. More specifically, the automated detection of tackles in soccer was appealing. **The aim is therefore to answer the following question:**

How can a framework be developed for automatically detecting tackle events in soccer videos using Machine Learning (ML)?

Furthermore, subquestions are organized into three categories: model-, feature-extraction-, and dataset-related questions:

1. **Model**
   (a) *RQ1. Does developing a model that combines simpler models, each specifically targeting a distinct class, lead to performance improvements?*
   (b) *RQ2. What is the benefit of employing recurrent neural networks with memory capabilities compared to models lacking such recurrent structures?*
   (c) *RQ3. What are the trade-offs with using a sequence-based model using a sliding vs. a shifting window over an input sequence?*
2. **Feature Extraction**
   (a) *RQ4. How well can classification be performed using features extracted from a pretrained feature extractor?*
   (b) *RQ5. How can different aspect ratios impact learning and is there a way that is more effective than the regular centre cropping?*

3. **Dataset**

   (a) *RQ6. Can a dataset that enables training a model to recognize different types of tackle events in soccer broadcast video be created?*

In addition to these research questions, this study is guided by several objectives, which are listed below.

1. Investigate current research on tackles in soccer and if there are any datasets that offer this.

2. Investigate if the variations in lighting conditions, camera angles, and player appearances commonly found in sports broadcasts prevent learning.

1.3 Scope

Event detection can be done for several sports, but in this thesis, the focus is exclusively on men's soccer. Videos, which feature several modalities including audio, are limited to the visual component for this research; audio features are not used. All videos used originate from televised broadcasts of the 2021 and 2022 seasons. Additionally, the content is confined to Norwegian soccer. The computational power available is constrained by the hardware available.

The primary constraints of this work include computational resources, licenses to use data and time.

1.4 Research Methods

For this research on unofficial event detection, the methodology outlined by the Association for Computing Machinery (ACM) has been adopted. The ACM report, *Computing as a Discipline* [7], published in 1989, provides insights into three paradigms: Theory, Abstraction, and Design. These paradigms will be described and their influence on the approach in this thesis will be illustrated.

The Theory paradigm within ACM's methodology focuses on establishing coherent and valid mathematical theories. It involves four stages: defining the objects of study, hypothesizing possible relationships among them, validating them through proofs, and interpreting the results. In this work, existing theories and models in the field of ML are drawn upon to formulate hypotheses about detecting unofficial events and explore the underlying relationships between features and anomalies.

The Abstraction paradigm, also outlined by ACM, is rooted in the experimental scientific model. It encompasses four stages: forming a hypothesis, constructing a model and making predictions, designing experiments to collect data, and analyzing the results. The principles of the Abstraction paradigm are followed by formulating hypotheses about the distinguishing characteristics of these unofficial events, developing models to capture these patterns, conducting experiments using appropriate datasets, and analyzing the results to validate the effectiveness of the proposed approach.

The third paradigm, the Design paradigm, is closely related to engineering practices. It involves four stages: stating requirements, specifying the system, designing and implementing it, and testing it. In this research, alignment with the Design paradigm is achieved by clearly defining the requirements and objectives of the detection system, designing and implementing algorithms or Neural Network (NN) architectures tailored to the task, and rigorously testing the system's performance using appropriate evaluation metrics and datasets.

By integrating these three paradigms — Theory, Abstraction, and Design — a comprehensive research methodology is formed. This methodology allows for exploring and developing novel approaches for detecting unofficial events, conducting experiments to evaluate their performance and draw insights from both theoretical and practical perspectives. By adhering to ACM's research methodology, a systematic and rigorous approach is ensured to advance the field of event detection in the domain of sports events.

Following this methodology, prototyping will be used to implement a solution that best answers the problem statement through the process and the final result.

1.5 Ethical Considerations

Using broadcast footage from the Norwegian Eliteserien, carries significant ethical responsibilities, especially since the videos feature visible faces and names of players. With the approval of Norsk Toppfotball (NTF), there is a commitment to ensuring that the research adheres to ethical guidelines regarding the use of personal data. Given that the videos used were already publicly available [1], anonymising the videos to protect player privacies was not needed, as these were already in the public domain. Players present in videos are taking part in a professional sports context and have given their implicit consent for broadcast video through the leagues' existing media agreements and regulations.

However, precautions were still taken to ensure that the usage and dataset comply with ethical standards concerning the use of these, focusing on minimizing potential misuse.

1.6 Main Contributions

This thesis introduces unexplored methods for automatically using AI to detect tackles during soccer broadcasts. This will be accomplished through an objective analysis, with the aim of identifying the current state of research involving tackle event detection in soccer. Automatic detection of tackles could enable content producers to provide fans

[1]https://highlights.eliteserien.no/

with highlights of events that, despite their central role in soccer game dynamics, often go unnoticed in terms of their impact on the overall match. This research contributes substantially to sports science, specifically in the field of soccer event detection, which has traditionally focused on goals, corners, substitutions, and cards. Additionally, these methods could aid in performance analysis for coaches and players by offering new insights into defensive manoeuvres, especially in amateur soccer, where financial constraints often limit such analysis. Implementing such systems in amateur soccer would contribute to Computer Vision (CV) research within event detection and the enhancement of player analysis, potentially elevating the sport's levels and development.

1.7 Thesis Outline

The remainder of this thesis is organized into the following chapters:

Chapter 2: Background and Related Work. This chapter provides information on topics including ML, content creation using AI Producer, and current research on tackle detection. It also offers an overview of open datasets related to soccer. Additionally, the structured data and features within video content are detailed.

Chapter 3: Dataset Curation - TACDEC. This chapter outlines the process of creating the TACDEC dataset.

Chapter 4: Methodology and Implementation. This chapter outlines the methods employed in the experiments, including the extraction of features and the preprocessing of the dataset described in Chapter 3. It provides reasoning for the methodological choices and offers detailed descriptions of the architectures and implementation of the models used.

Chapter 5: Experiments and Results: Spatial Approach. This chapter presents and analyzes the results following the experiments conducted using what is referred to as the spatial approach.

Chapter 6: Experiments and Results: Temporal Approach. This chapter presents and analyzes the results following the experiments conducted using what is referred to as the temporal approach.

Chapter 7: Discussion. This chapter discusses the findings and presents potential use cases. Further, limitations of the work and concerns related to the approach are presented.

Chapter 8: Conclusion. This chapter concludes the thesis by exploring insights gained from the research. It revisits and addresses the research questions and objectives, and suggests areas for future investigation.

Chapter 2

Background and Related Work

Based on the motivation and aim of this thesis, this chapter provides background information and work that can be related to it. The following chapter will present a list of terminology, followed by an introduction to AI Producer and an address of its limitations, which are central to this work. Subsequently, the exploration of video features, crucial to this thesis, details the fundamental information inherent within video content. This chapter also presents ML paradigms, NNs, and essential ML concepts that are either utilized in this thesis or necessary for a comprehensive understanding of the material. Before introducing available soccer datasets, there is a focus on tackles in soccer event detection and an introduction to the camera basics of broadcast videos.

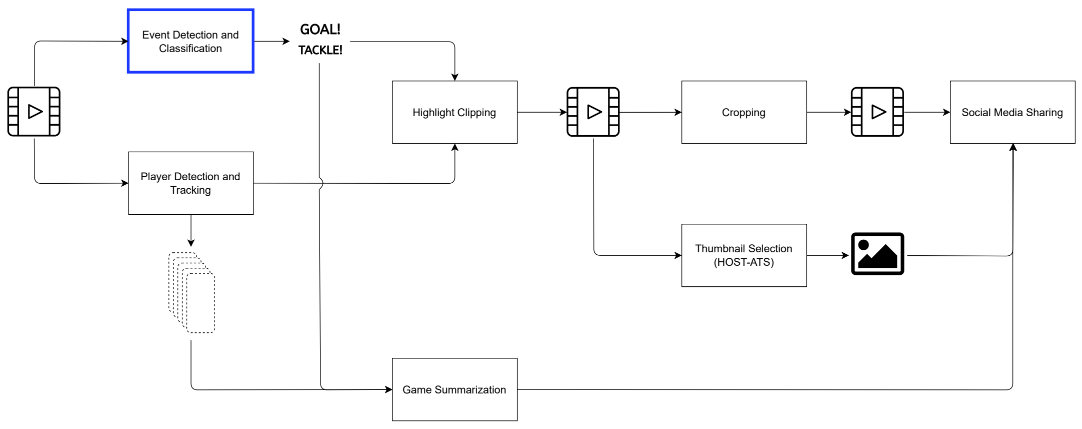
2.1 Terminology

To best understand video processing, it is necessary to define some terminology. The following chapter will discuss the different terminology and methods used in event detection.

- **Frame** - a single image.
- **Frame sequence / Video** - multiple consecutive frames grouped or classified together, thereby displaying some sort of motion. Often recognized as a video.
- **Tackle sequence** - following the former description of a frame sequence, a tackling sequence is, therefore, a frame sequence displaying a tackle event.
- **Action** - is something that is done, often with a distinct aim.
- **Detection** - is identifying someone or something.
- **Highlight clip** - a frame sequence that captures key moments or actions.
- **Soccer** - is also called associated football, a ball sport with two teams with 11 players each and referees.

- **Foul** - foul is awarded when there is an infringement to the *Laws of the Game* by IFAB [72].
- **Latency** - describes how much delay there is when transferring data from the time the transfer is instructed.
- **CLS-token** - a special token added to the input sequence that aggregates information across the entire image to serve as a representative feature.
- **Optimizers** - are algorithms or methods used to adjust the parameters of a neural network model to minimize a loss function.
- **API** - Application Programming Interface, a collection of protocols, routines, and tools used in software and applications to construct and communicate.
- **Labelled data** - samples that have been tagged with labels
- **Unlabelled data** - samples that do not have labels

2.2 AI Producer



Event Detection and Classification
GOAL!
TACKLE!
Highlight Clipping
Cropping
Social Media Sharing
Player Detection and Tracking
Thumbnail Selection (HOST-ATS)
Game Summarization

Figure 2.1: Illustration of the Forzify/AI-producer pipeline.

Forzify is a system created by ForzaSys to provide and engage fans with sports content [13]. Within this system, a sub-system using AI to assist video production called AI Producer [44] is under development. The basic idea of this sub-system is depicted in Figure 2.1. The AI Producer is an entirely autonomous system crafted to create social media content directly from raw broadcast videos of soccer and hockey games. Utilizing broadcast video as the initial input, this pipeline employs multiple modules to process and modify the data, facilitating the creation of various highlight materials. This fully automated system covers all stages of content production, including event detection, video clipping and cropping, thumbnail creation, summarization, and the distribution of the content, all without the need for human intervention.

To understand how the pipeline operates, a brief introduction to the set of modules is provided below:

- **Event Detection and Classification** - Event detection in broadcast videos often faces high latency and low accuracy, making real-time implementation difficult. This module introduces a pipeline that employs 3D Convolutional Neural Network (3D-CNN) to accurately detect official events, such as goals, cards, and substitutions [47, 59]. The pipeline's effectiveness is demonstrated across three datasets, including SoccerNet [19], enabling letting of the standard manual annotation process.

- **Player Detection and Tracking** - While event detection and classifications are performed on a clip basis, player detection and tracking require robustness as occlusions, rapid player movements, and lightning variations occur throughout a clip. Here, a player detection using Ultralytics YOLOv8 [36] fine-tuned on Forzasys's dataset [44].

- **Highlight Clipping** - In today's multimedia content production, creating highlights is a labour-intensive task that involves manually defining the start and end points of clips. This process becomes particularly time-consuming when handling large volumes of content. By detecting scene boundaries, logos, and replays and allowing for optional fan-cheering removal, highlights of goals are effectively captured and clipped by combining a VGG-inspired model, a ResNet model, and a TransNetV2-based model [74, 75].
- **Game Summarization** - Game summarization uses a multimodal scene-understanding pipeline, which includes videos, structured metadata, and captions. By reorganizing and modifying metadata, converting game commentaries into audio, and integrating a language model, it can generate detailed summaries of soccer games [15–17].
- **Cropping - SmartCrop** - When posting content on social media, it is important to note that aspect ratios can differ between platforms and screens. SmartCrop utilizes a Point of Interest (POI), like a specific player or the match ball, and tracks it using the TransNetV2 and YOLOv8-medium models [9, 63]. These models are refined with Forzasys' unique dataset, demonstrating effective results. In the case of multiple detections, false POI detections are eliminated using interpolation.
- **Thumbnail Selection - HOST-ATS** - Selecting the ideal thumbnail is crucial for accurately representing content with a single image. Factors such as relevance, context, and image quality are paramount. Accordingly, the HOST-ATS pipeline performs various detections, including face, logo, close-up shot, and blur detection, also conducting image quality analysis on frames [31, 32]. It accomplishes this by capturing features using CNNs.
- **Social Media Sharing** - The system currently supports TikTok and Instagram, featuring robust integration with authentication mechanisms. It handles requests and efficiently redirects access tokens. Once authenticated through the respective APIs, a preview UI popup appears, enabling text and post-type selection. This procedure simplifies user authentication and content customization, fostering seamless online content sharing [62].

2.3 TACDEC within AI Producer

The Forzify pipeline, undoubtedly, showcases an extensive array of features that make it a valuable tool in its domain. However, like any other system, it lacks limitations that must be addressed. The event detection and classification module of the pipeline, as it stands, is capable only of recognizing official events within a game [47, 59] such as goals or cards. This leaves out a significant aspect of the game – the unofficial events. Unofficial events such as tackles or dribbles often form the heart of the gameplay, providing exciting moments that engage fans and contribute to the dynamism of the sport. Tackles especially, are often involved in several game-decisive moments throughout a match as they play a central role in the defensive aspect of soccer.

2.4 Video Features

To understand how to detect events like tackles in a video sequence, it is essential to explain some basics about videos. A video is put together by several modalities, such as several images, often with audio, text and structured data describing the content or categorical features of the video. This thesis, will look into some of these, elaborated in the subsections of this section.

2.4.1 Spatial Features

Spatial features are found within the space of each frame. Curvature, edges, and shapes often outline the content of any image, like, for instance, an object. These can be used to illustrate a more abstract description of the object. Algorithms can extract and highlight these features to intensify and focus on whatever is important. Works such as [22, 57] show that enhanced spatial feature extraction yields better results.

2.4.2 Temporal Features

In contrast to spatial features, which remain static within a single frame, temporal features span the time dimension and highlight the dynamics unfolding within a scene over a period. By exhibiting variation over time, these features have been demonstrated to perform better in contexts where previously only spatial features were employed at time-series tasks, as evidenced by [3]. For example, in predicting when a golf ball will contact the ground, it becomes critical to understand its direction and velocity. An isolated frame merely shows the ball in a fixed position; however, analyzing a series of frames allows for the computation of both speed and trajectory, illustrating the temporal evolution of the scene.

2.4.3 Spatio-temporal Features

Spatio-temporal features integrate spatial and temporal characteristics and are fundamental enablers for scene analysis, particularly in video contexts. By facilitating space and time, these features provide a comprehensive understanding of dynamic scenes by merging the static aspects of spatial features with the motion-oriented insights of temporal features. For instance, in weather forecasting, utilizing spatio-temporal features has led to significant improvements [70]. Leveraging these features in weather prediction models enhances forecasting accuracy by offering detailed insights into the development and movement of weather patterns over time.

2.4.4 Structured Data

Listing 2.1: Example of a JSON-file

```
{
  "name": "Ola Nordmann",
  "siblings": {
    "sister": "Sister Nordmann",
    "brother": "Brother Nordmann"
  },
  "favourite": {
    "soccer-team": "Chelsea",
    "number": 10
  }
}
```

There are numerous methods to convey information. If one expects a continuous stream of data, standardizing the format in which this information is presented simplifies its use by ML models. Structured, highly organized data necessitates minimal preprocessing before being employed in ML applications. An example of how easily structured data can be understood can be found in Listing 2.1.

2.5 Introduction to Machine Learning

ML has through recent years gained a lot of popularity with tools like OpenAI's Chat-GPT [4] (an AI-language model designed to generate human-like text based on input) reaching one million users in only 5 days, way quicker than other popular services like Instagram (2.5 months), Twitter (24 months) and Netflix (41 months) [41].

To be able to develop intelligence, such as seen in Chat-GPT, the domain of ML contains many methods and algorithms, each with strengths and applications. Within video processing, specific methods and algorithms have emerged as key players in the pursuit of event detection. These techniques leverage the power of ML to analyze video data, extract meaningful features, and make accurate predictions. The following subsections delve into some important methods and algorithms used in video processing for event detection.

As several models are used in this thesis, developed upon different learning schemes, a brief introduction to the basics of these will be given below.

2.5.1 Machine Learning Basics

Different mechanisms are used to improve models before and throughout a training process to foster learning. In this section, some of the most important steps will be presented.

2.5.1.1 Preprocessing

Working with videos and images can be computationally time-consuming due to the abundance of information they contain. Preprocessing these media files, such as resizing or cropping, are often done to reduce this computational burden. This initial stage of preparing the data involves intricate algorithms, including techniques like Principle Component Analysis (PCA), that analyze and manipulate each frame or pixels. PCA helps reduce the data's dimensionality by identifying and emphasizing the most informative components, enabling more efficient processing and analysis.

2.5.1.2 Pooling Operations

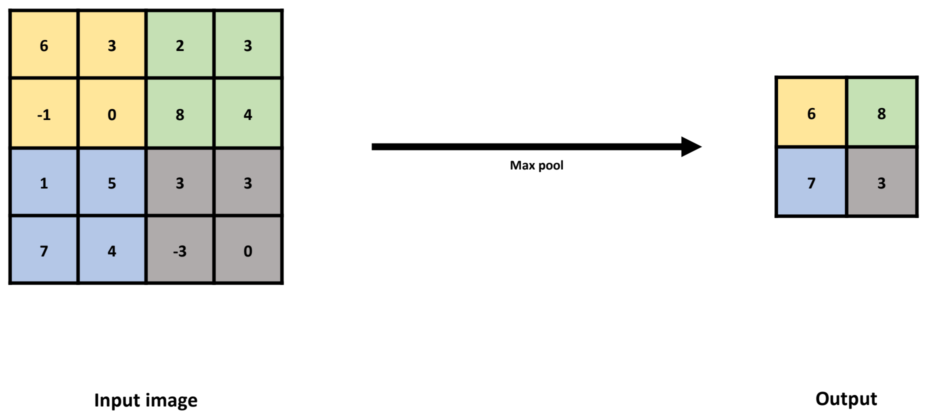


Figure 2.2: Pooling illustation.

Convolutional layers generate feature maps that capture various aspects of the input. These maps are highly sensitive, with minor input variations potentially altering them significantly. Pooling layers mitigate this by downsizing the feature maps, enhancing their invariance to small changes. Among the pooling techniques, average-pooling and max-pooling are common. Max-pooling, for example, applies a filter to select the maximum value within a specific window, as depicted in Fig 2.2. These play a large part in CNNs, more detailed in Section 2.5.3.2.

2.5.1.3 Local and Global Optima

In optimizing an objective function $f(x)$, where x represents controllable decision variables influencing its performance, various solutions can be identified. These solutions vary in effectiveness, with some being more optimal than others. The concept of Global Optima represents the absolute best solution across all possible values of x, characterized as the minimum or maximum value that $f(x)$ can achieve, dependent on whether the goal is to minimize or maximize the function. For a given objective function $f(x)$, the Global Optima is defined by:

$$\begin{aligned} f(x) > f(x_{global}) &\quad \text{for minimization} \\ f(x) < f(x_{global}) &\quad \text{for maximization} \end{aligned} \tag{2.1}$$

On the other hand, Local Optima pertains to solutions that are optimal within a specific subset of the input space rather than across the entire domain of x. This means there can be multiple local optima, each being the best solution within a certain vicinity. A Local Optima is defined such that:

$$\begin{aligned} f(x) \geq f(x_{local}) &\quad \text{for minimization within}[x_{local} - \epsilon, x_{local} + \epsilon] \\ f(x) \leq f(x_{local}) &\quad \text{for maximization within}[x_{local} - \epsilon, x_{local} + \epsilon] \end{aligned} \tag{2.2}$$

These global or local optima are commonly referred to based on the context as the local/global minimum or maximum, highlighting their role in optimization tasks.

2.5.1.4 Gradient Descent

Gradient descent is a fundamental optimization algorithm used to minimize a certain objective function, often a loss or cost function. It seeks the best parameter values that narrow the discrepancy between predicted and actual video classifications. This involves iteratively adjusting the model's parameters based on the objective function, which in video classification, quantifies the difference between predicted labels and actual labels for video clips. The algorithm computes the function's gradients with respect to the model parameters and modifies the parameters in the direction of the negative gradient. Conceptually, this process is similar to navigating a landscape with valleys and peaks, where the goal is to reach the lowest valley. This iterative process persists until convergence, as seen in Figure 2.3, resulting in optimized model parameters that improve the accuracy of video classification. It can be worth noting that there are different ways of computing these gradients, depending on how much data is used [60].

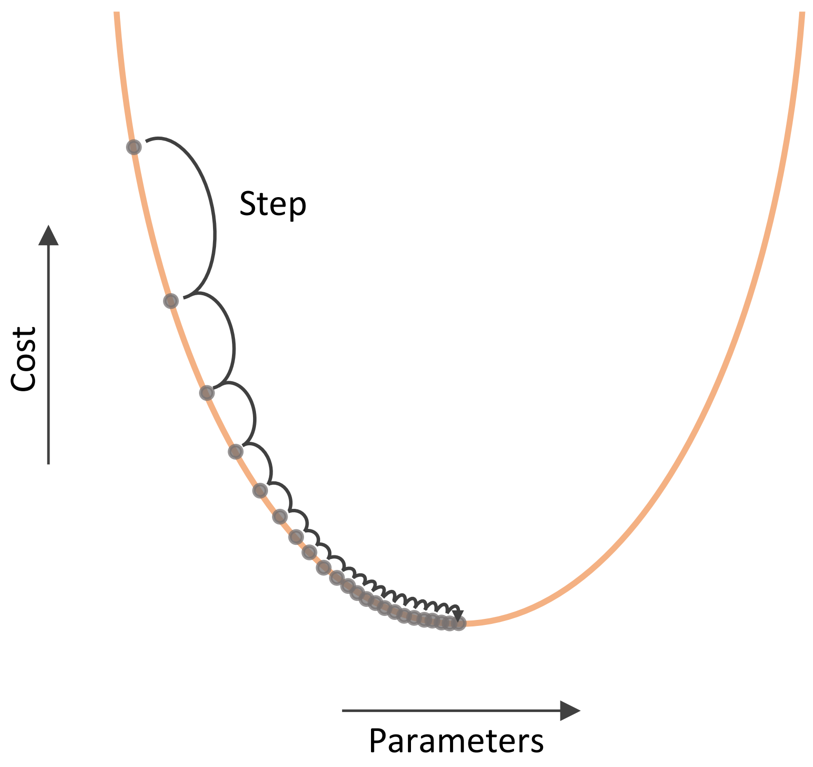
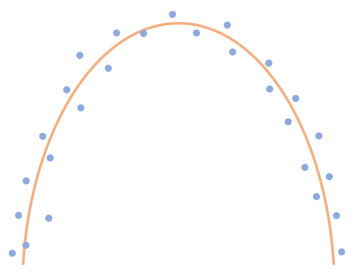


Figure 2.3: Illustration of the incremental steps to reach extrema.

2.5.1.5 Overfitting


(a) good fit

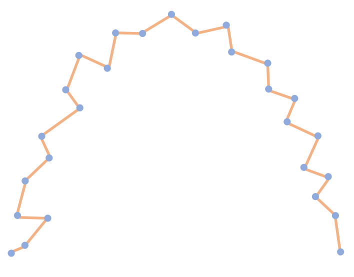

(b) overfitted

Figure 2.4: Example of a model's predictions (orange) to training data (blue).

Training an NN aims to create a model that generalizes well to unseen data. However, during training, the network may encounter complex data or be repeatedly exposed to the same data, causing it to learn and memorize noise, adapting too well to the training data. This phenomenon is called overfitting, see Figure 2.4. This is evident when the model demonstrates high accuracy on the training data but significantly lower accuracy on new, unseen data, indicating overfitting.

2.5.1.6 Batch Normalization

When training NNs, achieving high accuracy and stability is desired. Batching input data introduces beneficial noise that stabilizes and generalizes the network. The number of I/O operations and overall memory usage are reduced, accelerating the training process, especially when executed in parallel. Normalizing the input data ensures that all features are on a comparable scale, providing a uniform landscape for optimizers to navigate towards the local optima efficiently.

2.5.1.7 Regularization

Regularization methods are methods that add a penalty term to a loss function to prevent overfitting by regularising or shrinking weights in an NN. L1- and L2-regularizations are some of the most popular ones.

2.5.1.8 Backpropagation

Backpropagation is a technique used to calculate the gradient of the loss function with respect to each weight in the network, facilitating optimization in the direction of the steepest descent [61]. More specifically, backpropagation is used when training NNs. It leverages the concept of gradient descent to efficiently update the weights of an NN based on inputs. During training, backpropagation involves a forward pass, where the input data is processed through the network, and activations are computed. Then, by comparing the network's output with the ground truth labels, the gradients of the network's weights are computed using the chain rule. These gradients are used to update the weights, allowing the network to learn from its errors and improve its ability to recognize actions or detect video events. The network gradually optimizes its parameters through repeated forward and backward passes, enabling more accurate predictions.

2.5.1.9 Self-attention

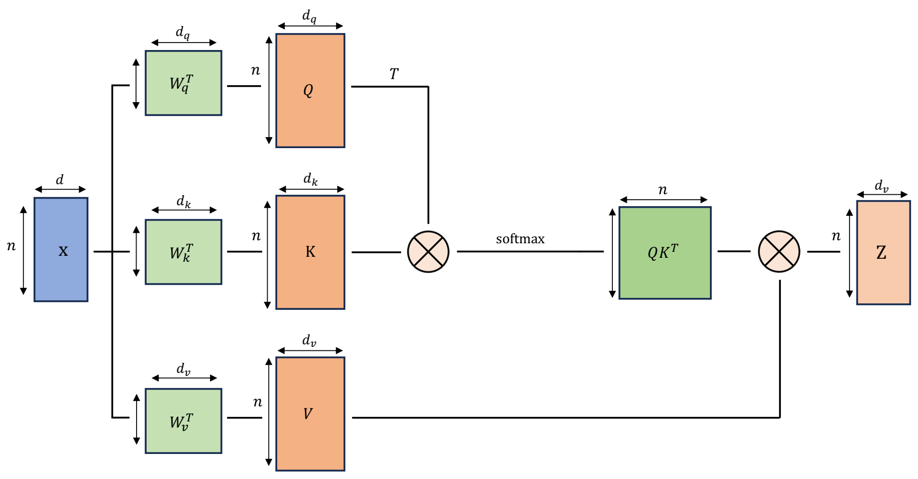


Figure 2.5: Illustration of how self-attention is calculated.

Self-attention is mechanism that is mostly used for sequence problems where some parts in the sequence are of more importance than other. For instance, if the task is to classify a sentence as either negative or positive, a sentence like **I love you** is easier to classify by paying attention to ***love*** than to ***I*** or ***you***. The same goes for the sentence **I hate you**, where ***hate*** can more easily help determine the mood of the sentence. Self-attention is calculated by transforming an input sequence X into three separate vectors: a query, a key, and a value. Upon these, further calculations and transformations are done, which separate weights for each vector, as seen in Figure 2.5 to finally obtain an attention output Z. This attention output shares the same dimensions as the input but is reweighted according to the computed attention weights. Essentially, this means that there is now attention included in the sequence, where values of lower importance score lower, and more important features are highlighted. The powerful benefits that self-attention offers have even proved to be beneficial enough to do image recognition solely based on different forms of self-attention and providing matching or outperforming results of those of a convolutional baselines [81]. Self-attention plays a pivotal role in Vision Transformers (ViTs), described more in detail later in Section 2.5.3.3

2.5.2 Machine Learning Paradigms

In ML there are several paradigms or ways to train models. This section will describe some of them.

2.5.2.1 Supervised Learning

Supervised learning is a fundamental concept in ML, where a model is trained using labelled data to make accurate predictions. This is central to how models presented in Chapter 4 are trained. In supervised learning, models begin with some initial weights or parameters. Using these weights, the model generates predictions for given input data. These predictions are then compared to the corresponding labels in the labelled dataset, allowing the model to update its weights based on the observed differences between predictions and true labels. By iteratively adjusting the weights through a process known as optimization, the model aims to improve its predictive performance over time. It should be noted that there are several methods for initializing weights, including assigning random numerical values to the weights or employing the widely recognized Kaiming initialization [26].

Given the assumption of having enough labelled data, supervised learning can, for example, use medical images to predict diseases like Alzheimer's accurately [80], or near-perfect classify numbers with an accuracy of 99.79% [78].

Classification Classification is a fundamental task in supervised learning to assign predefined labels or categories to input data. This process involves a binary problem that aims to distinguish between two distinct classes. Given a labelled training dataset, a classification model learns to recognize patterns and predict the observed features. The model's objective is to classify new, unseen instances into the appropriate classes accurately—either one class or the other—based on the learned patterns, such as classifying an email into spam or not.

Multi-class Classification Multi-class classification is about learning to predict classes when the problem is no longer binary. When distinguishing between two or more classes is desired, the methods used for classification will, therefore, not work as that is a linear problem. Determining what breed dogs are can be given as a multi-class problem. The problem faced in this thesis is a multi-class classification problem.

2.5.2.2 Unsupervised Learning

Unlike supervised learning, unsupervised learning uses unlabelled data to discover underlying patterns and structures. Unsupervised learning algorithms extract meaningful representations or features from the data without explicit annotations or labels. These methods identify commonalities, anomalies, or latent factors within the dataset through techniques like clustering and dimensionality reduction methods like PCA [42].

2.5.2.3 Self-supervised Learning

Self-supervised learning represents a paradigm shift in which the principles of supervised learning are applied without directly using labelled data. This methodology primarily utilizes unsupervised learning techniques to create supervisory signals directly from the dataset. In domains where the application of labelled data is crucial, such as CV, the labour-intensive process of manual labelling to establish ground truths presents challenges. The ViT DINOv2 [51], described later in Section 2.5.3.4, has achieved State of the art (SOTA) results within computer vision tasks by exploiting self-supervised learning.

Since most data remains unlabelled, self-supervised learning is a potent alternative when enough data is present. By setting up tasks brilliantly, this method automatically figures out the labels from data that doesn't have any, removing the need for manual labelling.

To simplify this, a self-supervised learning system learns to pair input data by creating and pairing it with a label, a 'ground truth'.

2.5.2.4 Transfer Learning

Transfer learning revolves around leveraging pre-existing knowledge and applying it to a new, yet related, domain to ease the learning process. Consider the analogy of cycling: if you have experience riding a bicycle in urban environments, transitioning to gravel riding does not start from scratch. Despite differences such as the riding posture, the shift from asphalt to gravel, and varied terrains, the fundamental skills of cycling, like pedalling, braking, and gear shifting, remain applicable. Similarly, in ML, transfer learning implies that a model trained on one task is repurposed for another, utilizing the foundational knowledge it has already acquired. This approach simplifies the learning process compared to building a model from the ground up without prior knowledge.

This process is typically achieved by freezing most weights within an NN. These foundational weights are crucial for extracting coarse details, such as edges and shapes, whereas the subsequent layers focus on finer details. Utilizing pre-trained weights, which have already been proven effective, transfer learning involves retraining the latter layers, often referred to as the head, to specialize in the desired domain.

2.5.2.5 Ensemble Learning

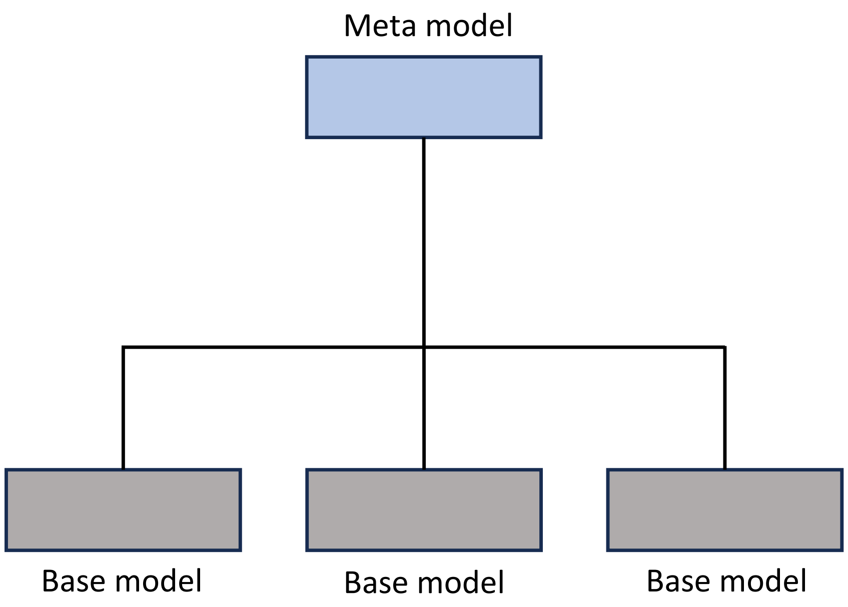


Figure 2.6: Illustration of the meta-learner structure.

Particularly when addressing problems characterized by noise and complexity or when dealing with imbalanced datasets, employing specialized techniques to enhance performance becomes crucial. These techniques include bagging, bootstrapping, or, as adopted in this thesis, the use of a meta-learner. The meta-learner is designed to discern the strengths and weaknesses of simpler models, see hierarchy in Figure 2.6. Understanding these characteristics aims to cooperatively integrate their predictive capabilities, thereby refining the overall performance. Ensemble learning can remind a bit of how knowledge can be increased within a group.

2.5.3 Neural Network Fundamentals

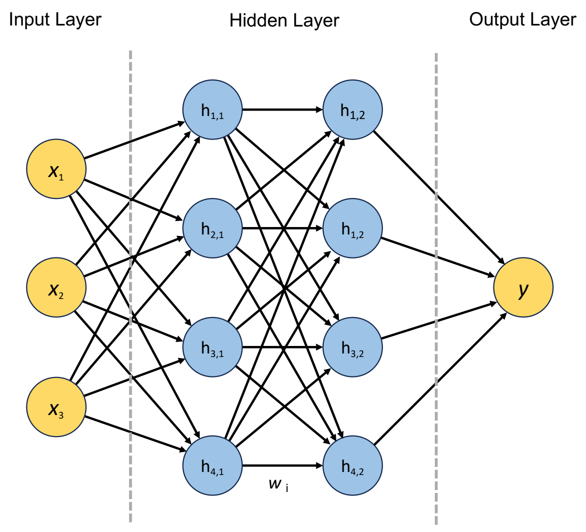


Figure 2.7: A simple neural network with input layer variables **x**, hidden layer variables **h**, weight variables **w**, and output variable **y**.

An NN, as seen in Figure 2.7, is a computational model that processes input data and generates predictions. It is trained using techniques like supervised learning with backpropagation, explained in Section 2.5.1.8, to learn the relationship between input data, such as pixels in a picture and labels. By adjusting internal parameters called weights, the network optimizes its ability to understand complex patterns in the data.

NNs plays a crucial role in video processing tasks such as event detection and classification. They consist of connected layers of neurons, forming a hierarchical structure to capture features in video data. Different types of NNs, such as Feedforward Neural Networks (FNNs), CNNs, and Recurrent Neural Networks (RNNs), are used depending on the task and data characteristics. These will be described in the following sections.

In an NN, the calculation in a neuron can be represented as follows: Each neuron performs a weighted sum of its inputs, considering the corresponding weights and a bias term. This calculation, often called neuron activation, allows the network to process information and make predictions. Different activation functions can be applied to introduce non-linearity and capture complex patterns in the data. This is calculated as seen in Equation 2.5.3.

$$z = \sum_i w_i x_i + b$$

where z is the weighted sum of the inputs x_i with corresponding weights w_i, and b is the bias term.

The activation function is applied to the weighted sum to introduce non-linearity and determine the neurons' output. The most commonly used activation function in CNNs is the ReLU function [56], which can be written as:

$$a = \sigma(z) = max(0, x)$$

where σ denotes the ReLU function.

While also the sigmoid function is also heavily used when output has to be probabilistic or binary, displayed in Equation 2.5.3 [39]:

$$a = \sigma(z) = \frac{1}{1 + e^{-z}}$$

where σ denotes the sigmoid function.

Complex relationships in the data are modelled through calculations and activations performed in each neuron throughout the network, enabling more accurate predictions. During training, the network adjusts its weights based on labelled data, optimizing its ability to generalize and make accurate predictions even on unseen data.

2.5.3.1 Fully Connected Neural Networks

A FNN, often referred to as a dense layer, consists of interconnected neurons across successive layers. Each neuron in a layer is connected to every neuron in the preceding and following layers, with specific weights assigned to these connections. This architecture ensures that every neuron contributes to the network's output through linear transformations and non-linear activations.

2.5.3.2 Convolutional Neural Networks

Feature extraction is a critical aspect of video processing, and CNNs has revolutionized this field. Often thought to be inspired by Fukushima's Neocognitron [14] and work on receptive fields and the hierarchical scheme of processing in the visual cortex by Hubel et al. [30], the mimicking of these visual processing mechanisms in the visual cortex enables models to learn and recognize patterns robust enough, allowing it to accurately identify objects regardless of their spatial orientation or shifts in position.

More technically, CNNs are designed to automatically learn and extract meaningful features from input data, making them particularly powerful in capturing spatial patterns like curvatures and edges. By leveraging convolutional layers, pooling operations, and non-linear activations, CNNs excels at feature extraction and hierarchical representation of visual information.

In video processing, CNNs have become highly effective tools for action recognition, object detection, and video segmentation. By applying convolutional operations, CNNs can capture important spatial patterns within video frames, enabling the network to understand and differentiate visual objects and structures. This capability of CNNs to extract discriminative features directly from raw video data enhances the accuracy and efficiency of video analysis, facilitating a deeper understanding of the visual content.

2.5.3.3 Vision Transformers

ViT has, in recent years, gained popularity since it first was presented [10]. A ViT is based on the transformers, which are models that are used in natural language processing, first presented in *Attention is all you need* [76]. These are capable of handling input sequences of text, like sentences, and processing these in either translating the sentences to another language, determining the mood, and more. By first splitting the images into different patches, the ViT uses an encoder network to process these patches. The encoder does not reduce the dimensionality of these patches but transforms them to enrich their representations with contextual information from the entire image. This transformation is facilitated by self-attention mechanisms (previously described in see Section 2.5.1.9), which allow elements of the input sequence to dynamically interact with each other, retaining the most relevant context. In addition to splitting the images into patches, some ViT models introduce a [CLS] token that serves as an aggregate representation of different image patches processed by the transformer. In the next section, a key component of this research, DINOv2 which utilizes ViTs, will be described.

2.5.3.4 DINOv2

DINOv2 is a self-supervised model using two distinct ViTs: a student ViT and a teacher ViT [51]. This architecture is designed to leverage the benefits of self-supervised learning, operating independently of labelled data. Instead of relying on predefined labels, DINOv2 generates supervisory signals based on the differences in data representations captured by the student and teacher models. Throughout the training process, the student model aims to mimic the behaviour and output of the teacher model.

Both the student and the teacher models share the same underlying architecture, but their weights are updated differently. The student's weights are adjusted through standard backpropagation techniques during training, whereas the teacher's weights are updated using an exponential moving average (EMA [25]) of both its own previous weights and the student's weights. This method helps stabilize the learning process and gradually improves the teacher model's performance by integrating insights gained from the student model's evolving capabilities.

The training regimen for DINOv2 was extensive, involving the model learning from a dataset of 142 million unlabeled images. This scale of data helped the model robustly capture and generalize across a wide range of visual features without the need for manual annotation. By utilizing self-attention mechanisms, DINOv2 efficiently identifies and focuses on the most relevant spatial features within each image, enabling it to understand and process complex visual inputs effectively, and has proved to be highly competitive for several visual tasks, achieving several SOTA performances as mentioned in Section 2.5.2.3.

2.5.3.5 LSTM

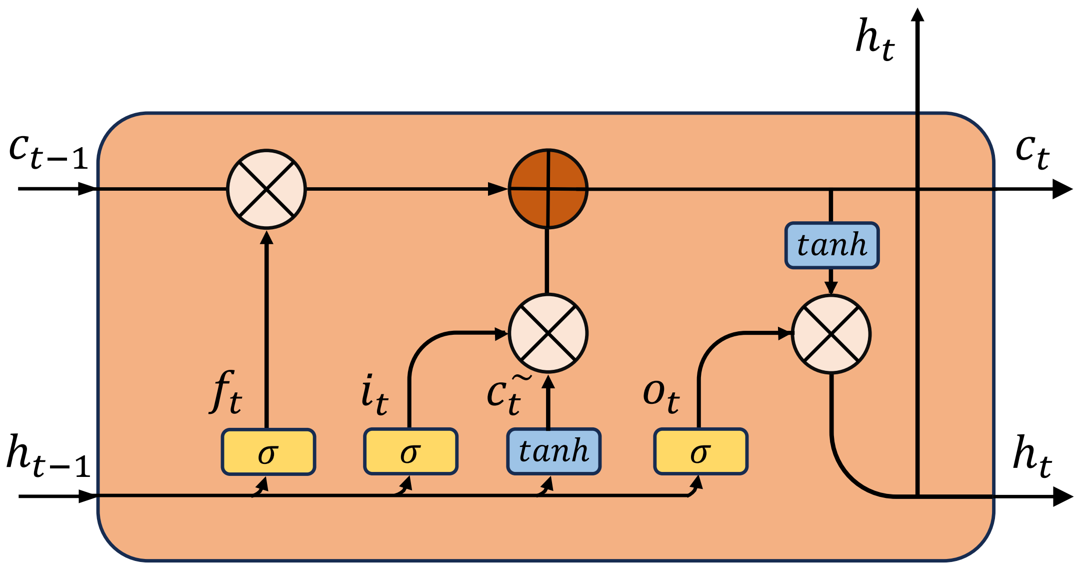


Figure 2.8: Illustration of the internals on an LSTM-cell.

First presented in 1997 by Hochreiter et al., Long Short-Term Memorys (LSTMs) was proposed as a modification of the RNN to address the challenges faced by the vanilla RNNs, particularly their struggle to retain information for long periods [27]. This shortcoming is significant in scenarios where the context is essential for understanding subsequent data. For example, if a person says, *I grew up in Norway*, a vanilla RNN might after a few sentences fail to recognize that the word ***Norwegian*** naturally follows after the sentence *My daily tongue is therefore ...*. Being able to understand sequences and remember the context, LSTMs has proven a solid enabler within action detection for soccer [1, 33].

Unlike RNNs, which generate outputs based solely on the current input and a hidden state (an intermediate representation of the input akin to a hidden layer in a FNN) acting as a memory, LSTMs incorporates a cell state (c) that acts as a memory unit instead. The difference is that the cell state uses three distinct gates: an input gate (i_t), a forget gate (f_t), and an output gate (o_t). These gates determine which information is retained, discarded, or passed to the next iteration, thus enhancing the model's ability to manage information flow more effectively than vanilla RNNs. For a visualization of how these work in an LSTM-cell, see Figure 2.8. Vanilla RNNs calculates the hidden state in the same way at every timestep without determining if some information is more important, causing a vanishing gradient (essentially means that it has no long-term memory).

The ability of LSTMs to remember and forget information is crucial when dealing with temporal data that requires a contextual understanding across time. This is in contrast to many other NN architectures that focus primarily on the features within individual samples, independently of each other.

Backpropagation in LSTMs is performed through these elements of the cell state, considering not just the current input but also the retained historical context. This process, known as Backpropagation Through Time (BPTT), is essential for optimizing the network's weights across sequences of data, thereby improving the ability to learn from and make predictions based on temporal data.

Bi-LSTM

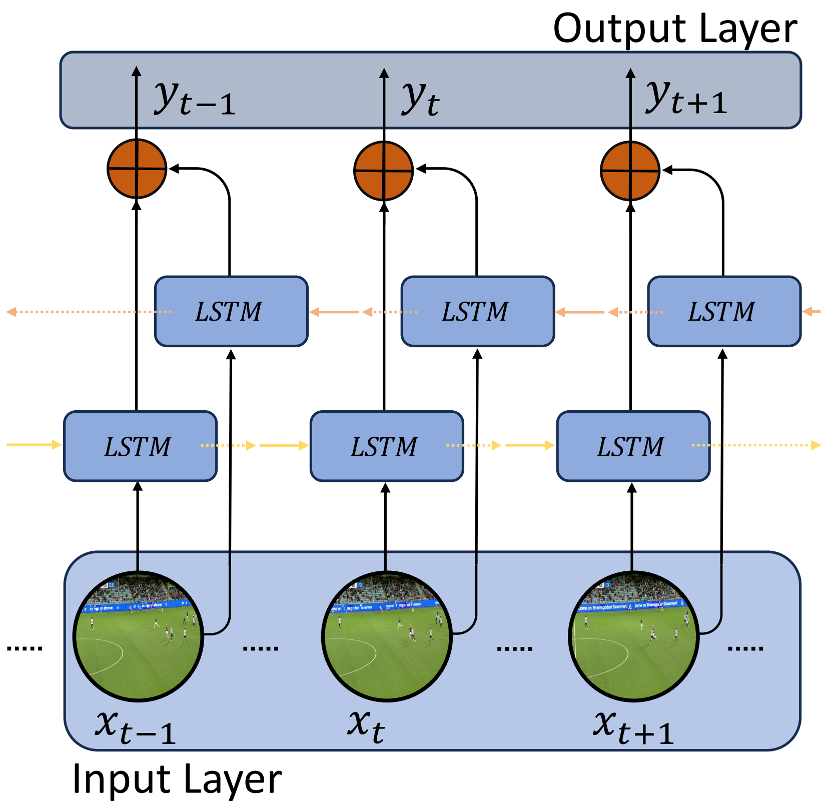


Figure 2.9: Illustration displaying how an input sequence is processed in both forward and backward directions to learn both past, present and future context.

Bi-LSTM, or Bidirectional Long Short-Term Memory, enhances the traditional LSTM model by incorporating an additional LSTM layer that computes sequences in both forward and backward directions as seen in Figure 2.9. This dual-direction processing allows the model to capture temporal relationships from both past and future contexts, effectively improving performance. The outputs from each direction are then combined using various methods such as summarization, multiplication, and concatenation, thereby enriching the model's ability to understand sequential data more comprehensively [24].

2.5.4 Evaluation Metrics

For performance evaluation, the metrics outlined in this section will be employed. When working with these evaluation metrics and providing analysis, closely related terms will also be introduced. These terms will therefore also be presented:

1. **True Positive (TP)** - a prediction is evaluated as Class X and truly belongs to Class X
2. **True Negative (TN)** - a prediction is evaluated to **not** belong to Class X and does **not** belong to Class X
3. **False Positive (FP)** - a prediction is evaluated as Class X and does **not** belong to Class X
4. **False Negative (FN)** - a prediction is evaluated to **not** belong to Class X but truly belongs to Class X

Note that all these metrics range from 0 to 1.

2.5.4.1 Accuracy

Accuracy is a measurement of how many of all predictions made were correct out of the total number of predictions made. The formula for accuracy can be seen in Equation 2.3.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.3}$$

2.5.4.2 Precision

Precision is a measurement of how many of the instances identified as positive are actually true positives. The formula for precision can be seen in Equation 2.4.

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

2.5.4.3 Recall

Recall is a measurement of how many of the total instances of a class were captured. The formula for recall can be seen in Equation 2.5.

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

2.5.4.4 F1-score

F1-score is the harmonic mean of precision and recall. It provides a balance between precision and recall, effectively accounting for both false negatives (False Negative (FN)) and false positives (False Positive (FP)), offering a single metric to evaluate a model's performance. The formula for recall can be seen in Equation 2.6.

$$F_1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.6}$$

2.5.4.5 ROC Curve

Short for the Receiving-Operating-Characteristics curve, ROC curves present a plot of the true positive rate on the y-axis against the false positive rate on the x-axis. The further shifted towards the top left corner, the better. AUC scores are a measurement of the area under the curve to better evaluate the graph.

2.6 Event Detection in Soccer

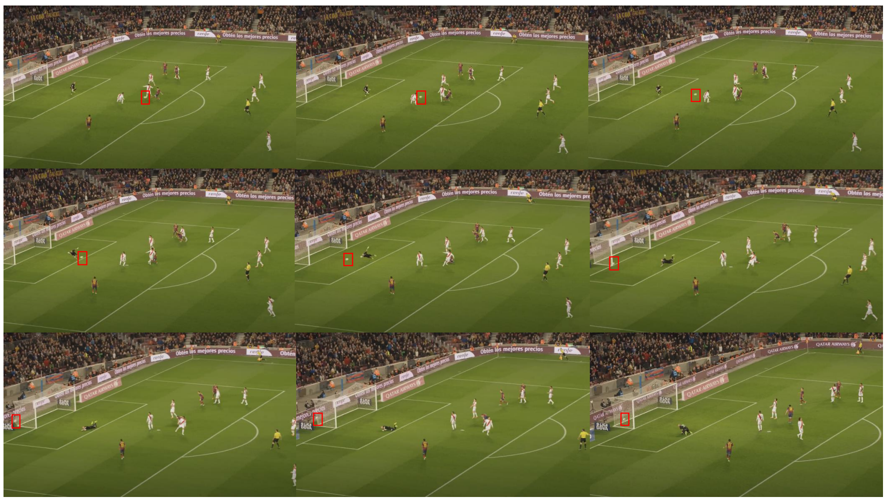


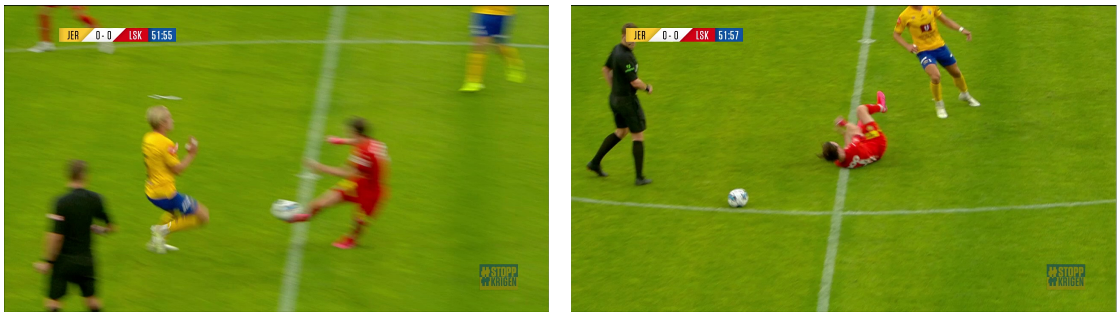
Figure 2.10: Action detection of a goal in a soccer match.

Event detection refers to the process of identifying and recognizing specific occurrences or actions within a given dataset or stream of data. This can, for instance, be done by training an NN to recognize specific patterns in videos, such as when a ball enters the net to score a goal, as illustrated in Figure 2.10.

But goals are not the only events that occur in a soccer match. There are several events occurring through a 90-minute soccer match, some more decisive than others. A huge dataset containing highly detailed data unveiled that across 1941 matches, the average game contained closer to 1700 events [52]. These events can range from traditional events like goals, penalties, or cards to more specific events such as tackles, mistakes, near misses, or significant player movements. By automatically detecting and categorizing these events, valuable insights such as key moments or actions can be extracted, and the analysis and understanding of the game can be enhanced.

While traditional events like corners, goals, yellow cards, substitutions, throw-ins, and kick-offs have been studied in recent years [6, 19, 46, 47, 59], there is limited research in the less frequent domain of tackle detection, and not using CV [37, 45]. This is despite tackles playing a crucial role in the game, both in decisive moments and in regaining possession and is also why the research is based upon it.

2.7 Tackle Definition



(a) Start Frame. (b) End Frame.

Figure 2.11: Display of where a tackle sequence starts and ends.

As this research evolves around tackles, defining it is crucial. International Football Association Board (IFAB), which sets the 'Laws of the Game' for soccer worldwide [72], defines a tackle as:

A challenge for the ball with the foot (on the ground or in the air) [21]

Opta Sports, which is a subsidiary of Stats Perform [49], on the other hand, defines it otherwise. Being the official statistics partner of several United Kingdom (UK) domestic leagues, including the English Premier League (EPL) [68], and providing stats for several bookies [29], they separate between missed tackles and tackles. A tackle is defined as the successful recovery of the ball by connecting with the ball '*in a legal, ground-level challenge*' [50]. A tackle can then be categorized as won or lost, depending on which team receives the ball after the end. A missed tackle is a failed attempt to challenge for the ball, leading to fouls.

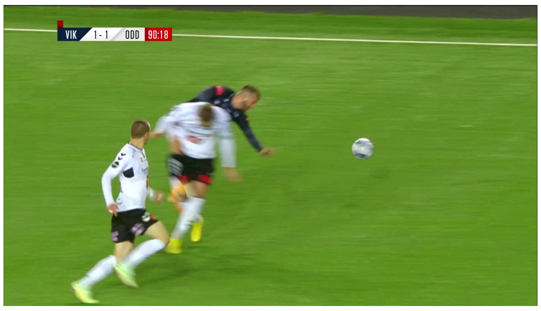
In this work, the definition of a 'tackle' is refined to encompass the entire sequence of a tackle, as follows:

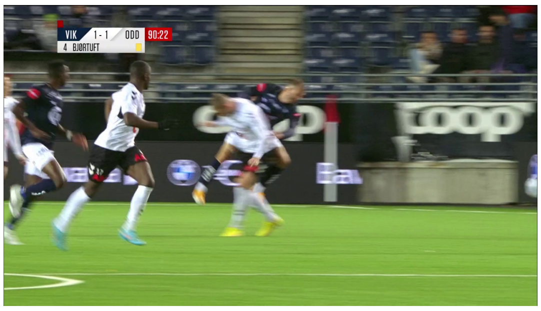
A tackle is initiated when the tackler applies pressure onto their supporting foot, distinct from the foot used for tackling, see Figure 2.11a (in yellow).

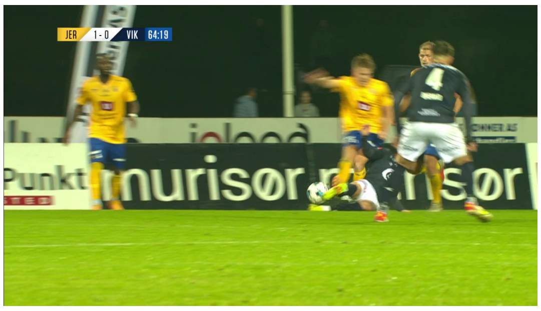
This action involves a forward motion directed into the opponent, propelled by the force generated from the supporting foot. The tackle is considered complete either when the tackled player (if airborne) makes contact with the ground or when the forward momentum of the tackle ceases, see Figure 2.11b (in red).

It is independent of whether the ball has been recovered and whether a foul is given.

2.8 Shot (Scene) Type


(a) Game X - Close-up from touchline-view 1


(b) Game X - Close-up from goal-end


(c) Game Y - Close-up from touchline-view 2

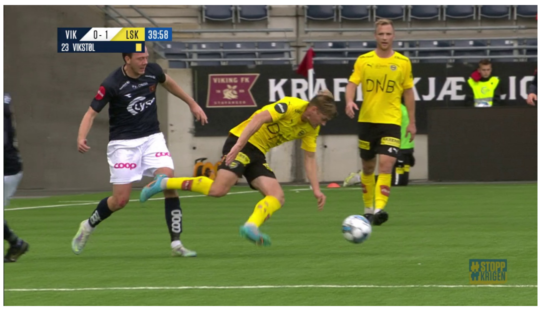

(d) Game Z - Close-up from corner

Figure 2.12: Some of the different shot types seen in Norwegian Eliteserien.

In video analysis, especially in scenarios where clips are combined, such as in broadcast footage, various shot types emerge. These shot types represent different perspectives or views of the object of interest. Shot types refer to the point of view. To see some of the different shot types that appear in broadcast video, see Figure 2.12. As we can see, Figure 2.12a and Figure 2.12b display the same tackle but from two different perspectives. Figure 2.12c and Figure 2.12d are close-ups from games played at different stadiums where the cameras are in different positions.

2.9 Open Soccer Datasets

Although there has been an increase in action spotting and event detection for soccer, with public datasets being released [6, 19, 35, 59, 79], there is still lack of tackles being annotated.

Open Datasets						
Events	SN	SN-v2	Yu	SDB	Papp	Rog
Videos	✓	✓	✓	✓	-	✓
Goals	✓	✓	✓	✓	✓	✓
Cards	✓	✓	✓	✓	✓	✓
Shots	-	✓	✓	✓	✓	-
Subs	✓	✓	-	✓	✓	✓
FKs	-	✓	✓	✓	✓	-
PKs	-	✓	✓	✓	✓	-
Corners	-	✓	✓	✓	✓	-
Fouls	-	✓	✓	✓	✓	-
Offsides	-	✓	✓	✓	✓	-
(other)	-	✓	✓	✓	✓	✓
Tackles	-	-	-	-	✓	-

Table 2.1: Comparison of Open Datasets on Soccer Events.

From Table 2.1, we can see that the dataset of Pappalardo et al. [52] is the only one available with tackles. Unfortunately, this dataset only contains structured data without videos or clips.

Four of the six datasets listed in the table include instances of fouls, which can easily be confused with tackles. It is crucial to understand that while tackles may result in fouls, executing tackles without infringing the rules is entirely feasible. Therefore, a tackle dataset enriched with video content would significantly contribute to filling the existing research void in soccer event detection.

2.10 Chapter Summary

Through this chapter, background information essential for a comprehensive understanding of this thesis is presented. This chapter highlighted the significant role of video features in computer vision and reviewed the basics of machine learning, illustrating how these can help stabilize and optimize learning processes across various learning paradigms. It discussed different neural networks, with a particular focus on DINOv2, alongside the metrics used for evaluating such models. The chapter also explored current research on event detection in soccer, specifically how tackles are defined and their relevance to this field. Additionally, related research is included to complement the provided information. Concepts and methods from this chapter form the methodology that will be employed to conduct experiments in the following chapters.

Chapter 3

Dataset Curation - TACDEC

In this chapter, the motivation behind and the process of creating this unique dataset are outlined. Given that existing datasets did not meet specific needs, the decision was made to construct TACDEC to explore an area that has received limited attention in previous research. The comprehensive steps involved in this creation, from data collection and annotation to the analysis and presentation of the dataset's statistics, are described. The dataset is accessible on the HuggingFace platform [1] and has featured in ACM MMSys 2024 [38]. For paper and conference poster, see Chapter A of the appendix.

3.1 Motivation for Curating Own Dataset

The comprehensive review of existing datasets, as highlighted in Table 2.1 from Section 2.9, underscores a notable gap in resources available for this specific type of research. Tackles, a fundamental aspect of soccer's defensive play, hold a unique place, captivating fans with a keen appreciation for the blend of technical skill, genuine passion, and the sheer adrenaline that fuels players and spectators alike. The essence of tackles transcends mere physical engagement; it embodies the strategic depth and emotional intensity that define the sport.

The motivation to curate the TACDEC dataset stems from recognizing this significant void. The existing datasets predominantly concentrate on more quantifiable aspects of soccer, such as goal counts or card incidents, overlooking the nuanced complexities and dynamic nature of tackles. These moments are not only pivotal in shaping the outcome of matches but also in enriching the viewing experience, offering insights into the players' skills, tactical decisions, and the overarching strategies employed by teams.

With resources enabling us to gather data across several seasons directly from the Norwegian domestic league, Eliteserien, a collection of 425 videos has been carefully labelled with four tackle classes. Throughout these 425 clips, a staggering 836 tackle events take place, unlocking the opportunity to do the research that has been missing.

[1] https://huggingface.co/datasets/SimulaMet-HOST/TACDEC

3.2 Sources and Data Collection

To obtain 425 clips, a considerable number of videos had to be secured and reviewed to ensure a tackle event was present and ready to be labelled. Through a collaboration with the Norwegian Professional Football League (Norsk Toppfotball) [2], the access to Application Programming Interfaces (APIs) from their partner company ForzaSys [3] afforded the opportunity to retrieve video clips that were already tagged with the different events.

Using these APIs, clips tagged as yellow card events were acquired, as these were seen with a much higher likelihood of featuring tackle occurrences among the available events. This was given that yellow cards are frequently awarded following tackles that foul the opposition player.

The data collection process, therefore, involved analyzing yellow card events from 246 soccer matches, focusing on the 2022 and 2021 seasons. Specifically, events from 176 games in the 2022 season and 70 games in the 2021 season were collected. The initial objective was to compile all yellow card incidents involving all 16 teams during the 2022 season. This scope was later broadened to include yellow card incidents from six randomly chosen teams (Brann, Odd, Rosenborg, Sarpsborg, Viking, and Vålerenga) during the 2021 season, in addition to the complete 2022 season.

The data extraction process returned 1015 individual video clips, each of which required manual review to confirm the presence of a tackle and ensure audio availability for potential multi-modal analysis. The review process also involved differentiating between tackles and instances of pushing or pulling, a task that presented challenges due to the frequent overlap of these actions.

The exclusion of clips with no tackles, no audio, or that proved hard to categorize reduced the number of eligible clips from 1015 to the final dataset, compromising 425 clips that accurately represented tackle events. Although the size was significantly reduced, this refined dataset contains 836 tackle events annotated across all 16 teams for each surveyed season.

Given the dynamic composition of teams in the Eliteserien, with at least two teams relegated (and a third potentially involved in a relegation play-off) and at least two promoted annually, the team lineup changes yearly. The dataset is meticulously broken down in Table 3.1, which details each team's total annotated events by class across the two seasons. Including games with all teams represented ensures a robust dataset for analysis, reflecting the nuances and variations of gameplay across different teams and seasons.

[2] https://toppfotball.no/
[3] https://forzasys.com/

3.3 Annotation

Because annotating a large amount of data can be overwhelming, utilizing an annotation tool not only simplifies the labelling process but also functions as a platform to monitor progress and provide a more detailed overview.

3.3.1 Labelbox

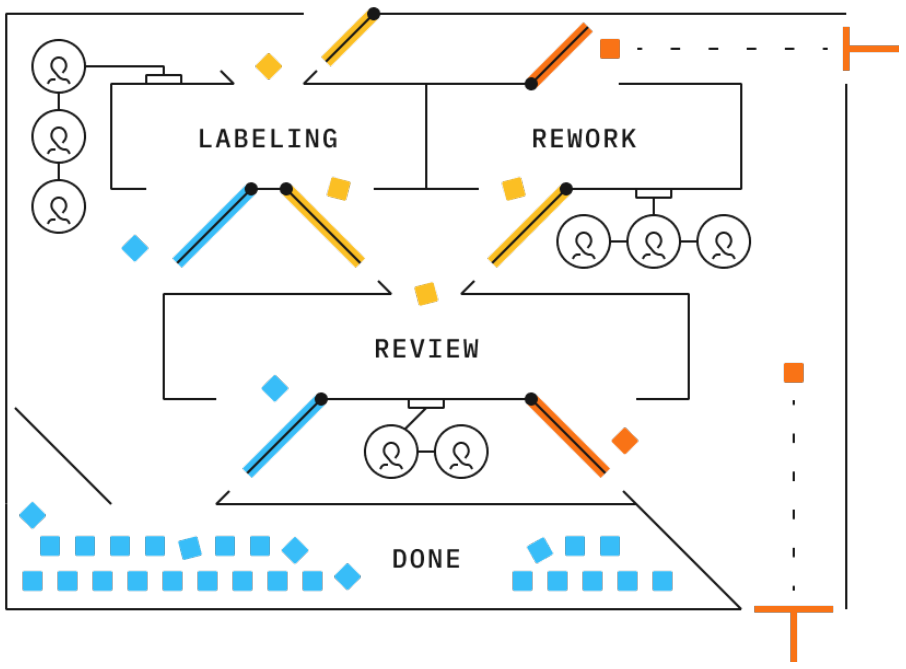


Figure 3.1: Illustration of the Labelbox workflow.

Video annotation can be tedious and labour-intensive, requiring constant attention and precision to secure quality. With several annotation tools at disposal, including Supervisely [69], CVAT [5], and VGG [77], the decision was made to employ Labelbox [40] for the dataset.

With a user-friendly interface easy-to-navigate, as seen in Figure 3.3, Labelbox came out on top, offering hotkeys for labelling the current frame, as well as navigating onto the following frames. Offering a larger annotation limit for free, the prices needed to complete the full annotation were competitive. Utilizing Labelbox was easy, requiring no extensive training.

To successfully label an item in Labelbox, a review upon the initial annotation is required; see Figure 3.1, extending the duration of the process but done to ensure data quality. This choice was also further reinforced by the knowledge that Labelbox is trusted and utilized by prestigious organizations such as NASA [28], underscoring its reliability and effectiveness in handling complex annotation tasks.

3.3.2 Annotation Process

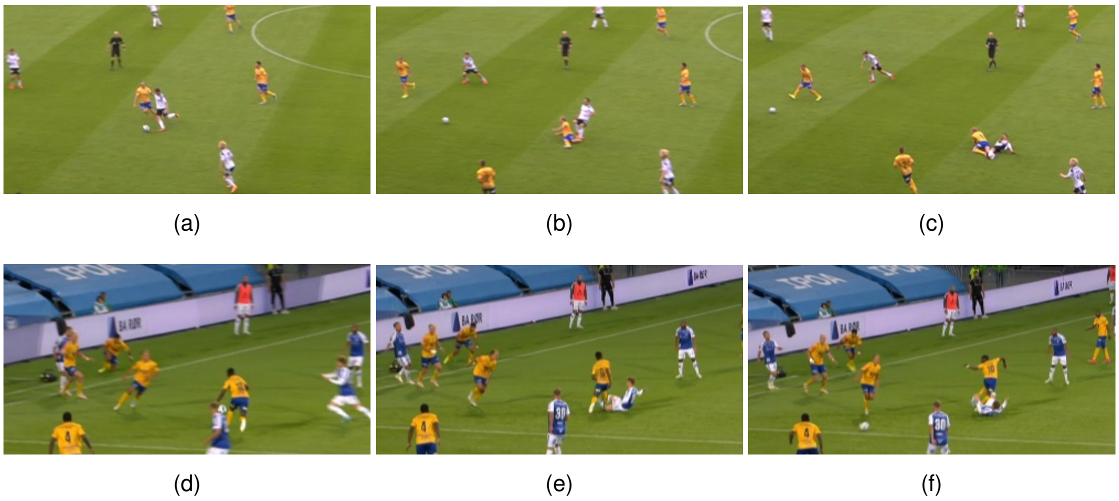


Figure 3.2: Sample frames from a `tackle-live` event with a side block tackle (top), and sample frames from a `tackle-replay` event with a sliding tackle (bottom). Starting frames in (a) and (d), ending frames in (c) and (f).

To maintain uniformity in the class definitions during the annotation process, each clip selected was subjected to an initial visual inspection. This step was critical to establishing definitions and categories that were comprehensive enough to encompass all events chosen for the dataset. By doing so, four instances or classes to best represent tackle events was created.

Given that entire tackle sequences were labelled rather than just the frames of impact, it is crucial to define precisely the start and end frames for what a tackle is. Below are the descriptions of the four classes used to annotate, plus a background class:

1. **background:** For when none of the four other class labels below are applicable.
2. **tackle-live:** A 'real-time' tackle in a broadcast video. A specific tackle can only be a live tackle once, even though there could be multiple unique live tackles in one clip.
3. **tackle-replay:** When a tackle is replayed during the same clip. It could be in slow-motion or zoomed, and there could be multiple replays for each live tackle.
4. **tackle-live-incomplete:** A live tackle, for which the start or end frame is not visible according to the definition of a tackle. This could be due to clipping artefacts or players being off-screen.
5. **tackle-replay-incomplete:** A tackle replay, for which the start or end frame is not visible according to the definition of a tackle. This could be due to clipping artefacts or players being off-screen.

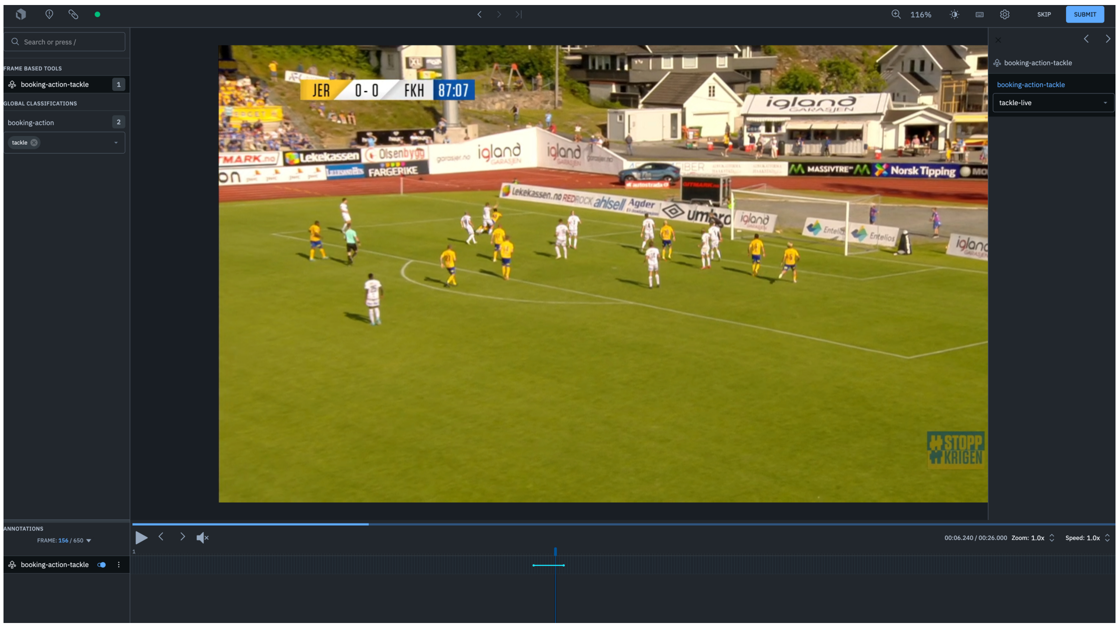


Figure 3.3: A display of the Labelbox editor.

As mentioned, a clear way of defining where a sequence starts and ends was needed. The criteria for determining the start and end of a sequence are defined as follows:

1. **Start frame:** The moment the defender redistributes their weight onto the leg that is used to launch into the tackle, initiating the movement.

2. **End frame:** The tackle is considered complete when one of the two following conditions are met:

   (a) The forward motion of the player tackling has stopped.

   (b) If airborne, the tackled player makes contact with the ground.

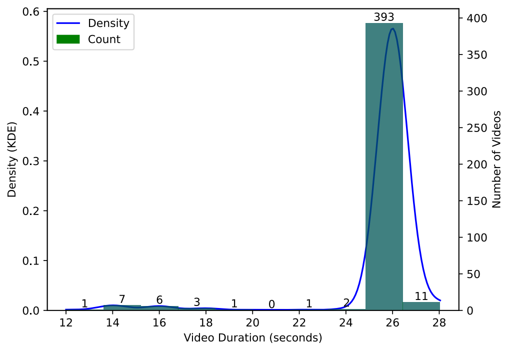
Figure 3.2 depicts a cut out of two different tackle sequences to illustrate the visuals of frames annotated.

To initiate the annotation process in Labelbox, videos were first organized into a batch which was required to create a task. The annotation process was then carried out using the Labelbox Editor, as illustrated in Figure 3.3. This involved conducting a frame-by-frame analysis to classify each frame according to its corresponding class. When a batch was finished, the respective videos were sent for review and, from there, revised as needed.

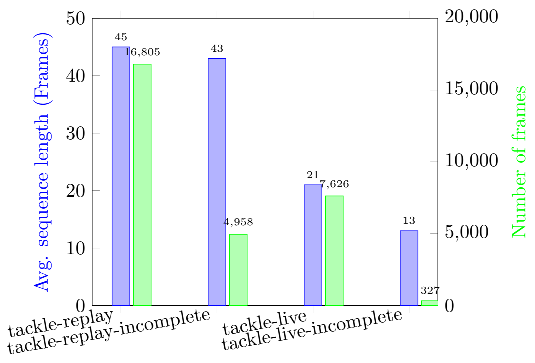
Through this process, a careful review of every frame of the 425 videos was completed, and 271,902 frames were reviewed twice.

After the annotation process was completed, labels were exported as JSON files from Labelbox and given a filename corresponding to the external ID of the video labelled.

3.4 Dataset Statistics



(a) Histogram of Video Duration.



(b) Frame count and average sequence length per class.

Figure 3.4: Comparison of dataset statistics.

The refined dataset consists of 425 labelled video clips, with an average duration of approximately 0.43 minutes, which equals 25.8 seconds per clip, see Figure 3.4a. The total frame count and average sequence length are displayed in Figure 3.4b.

Team	**Season**	**tackle-live**	**tackle-replay**	**tackle-live-incomplete**	**tackle-replay-incomplete**	**Total**
Aalesund	2021	-	-	-	-	-
	2022	10	18	1	5	34
Bodø/Glimt	2021	3	2	1	2	8
	2022	20	16	-	4	40
Brann	2021	8	2	1	2	13
	2022	-	-	-	-	-
HamKam	2021	-	-	-	-	-
	2022	22	29	2	13	66
Haugesund	2021	2	2	-	-	4
	2022	35	29	2	5	71
Jerv	2021	-	-	-	-	-
	2022	28	16	2	6	52
Mjøndalen	2021	1	-	-	-	1
	2022	-	-	-	-	-
Kristiansund	2021	1	2	-	2	5
	2022	9	20	-	1	30
Lillestrøm	2021	6	8	-	3	17
	2022	20	12	1	10	43
Molde	2021	1	1	-	-	2
	2022	19	17	-	2	38
Odd	2021	4	10	-	2	16
	2022	20	18	1	11	50
Rosenborg	2021	1	9	-	2	12
	2022	11	13	2	7	33
Sandefjord	2021	2	1	-	1	4
	2022	18	14	2	7	41
Sarpsborg 08	2021	9	10	2	1	22
	2022	7	17	-	2	26
Stabæk	2021	6	4	1	-	11
	2022	-	-	-	-	-
Strømsgodset	2021	4	4	1	2	11
	2022	13	22	2	2	39
Tromsø	2021	4	4	-	3	11
	2022	10	14	0	5	29
Viking	2021	9	7	-	-	16
	2022	16	21	0	6	43
Vålerenga	2021	3	2	1	1	7
	2022	19	17	0	5	41
All	**All**	**341**	**361**	**22**	**112**	**836**

Table 3.1: Number of annotated events in the dataset, per home team and season.

The distribution of each class can be seen in the pie chart in Figure 3.5, with the distribution of these samples as events across different teams and seasons detailed in Table 3.1. To ensure clarity and avoid double counting, each sample is uniquely attributed to the home team involved in the event. Accompanying each video clip is a distinct JSON file, as outlined in Listing 3.1, which provides comprehensive details on the labelled sequences, including class labels and each sequence's start and end frame. The size of this dataset is regarded as sufficient, considering the results and dataset sizes observed in similar research [12, 71].

Listing 3.1: Template for the JSON files in the dataset.

```
{
  "id": "1234_qwerty",
  "metadata": {
    "game-id": 1234,
    "date": 2023-02-01,
    "team-home": {"name": "TeamH, "id": 12}
    "team-away": {"name": "TeamA", "id": 34}
    "clip-id": "qwerty"},
  "media_attributes": {
    "height": 720,
    "width": 1280,
    "mime_type": "video/mp4",
    "frame_rate": 25,
    "frame_count": 650},
  "events": [
        {"start_frame": 200,
         "end_frame": 300,
         "type": "tackle_live"},
        {"start_frame": 325,
         "end_frame": 450,
         "type": "tackle_live"},
        {"start_frame": 520,
         "end_frame": 600,
         "type": "tackle_replay"}]
}
```

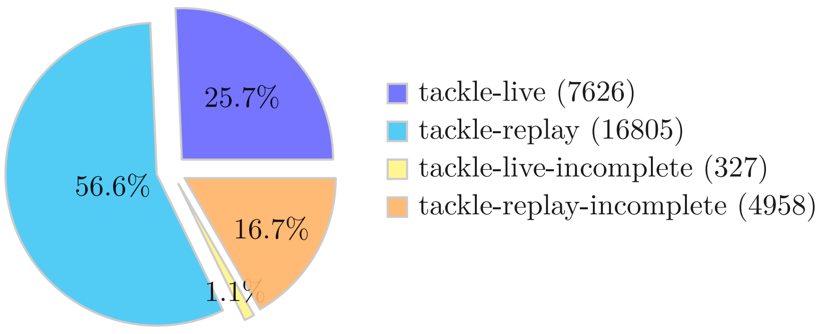


Figure 3.5: Distribution of event classes.

3.5 Chapter Summary

In this chapter, the TACDEC dataset was introduced, which was developed through collaboration with the Norwegian Professional Football League and Forzasys. This dataset, compiled from the 2021 and 2022 Eliteserien league seasons, includes 425 video clips totalling 836 tackle events, offering a robust resource for soccer tackle research.

The dataset's creation was discussed, highlighting the efficient use of ForzaSys APIs and Labelbox for data collection and annotation. The methodology ensured consistent classification of tackles, setting a benchmark for future sports analytics studies.

In the next chapter, a proposed methodology for using this dataset to investigate whether ML can detect tackles in soccer videos automatically will be presented.

Chapter 4

Methodology and Implementation

In this chapter, the methods employed in the research are detailed. Utilizing the dataset curated in Chapter 3, the aim is to develop two distinct frameworks designed to facilitate the automatic detection of tackles in soccer videos.

4.1 TACDEC

For all experiments, the TACDEC dataset created will be used [38]. The dataset itself has a designated chapter in Chapter 3, containing a detailed description of the entire process.

Given the structured data in the labels, processing prior to use is needed. In the TACDEC dataset, tackle events within the videos are identified in the JSON labels by a start and end frame for each sequence. Given the objective of frame-by-frame classification, it is necessary to alter these labels to create a continuous span for each sequence, encompassing all frames from start to end.

From Chapter 3, we saw that the 'background' class is for the frames where none of the four labelled classes applies. Originally, TACDEC did not include explicit labels for these background frames. Therefore, a decision for these to be labelled explicitly was made.

By processing and concatenating the labels for all frames across the videos, and aligning them in sorted order using a unique ID, a list representing the labels for every individual frame, encompassing the total number of frames present in the dataset will be made.

In the spatial approach described in Section 4.4, labelled frames from three classes will be used: 'background,' 'tackle-live,' and 'tackle-replay.' Frames labelled as 'tackle-live-incomplete' and 'tackle-replay-incomplete' will be mapped to their respective complete classes, since it is nearly impossible to determine from a standalone frame whether it belongs to a complete or incomplete sequence without having any temporal context.

For the temporal approach outlined in Section 4.5, a similar strategy of mapping instances to their corresponding full tackle classes will be adopted. This is due to the incomplete sequences being represented by really low counts.

4.2 Spatial Feature Extraction with DINOv2

The data contained within the pixels of each frame is of significant importance in processing videos and images. These are the so-called spatial features, described more in detail in Section 2.4.1. However, processing these frames demands substantial computational resources; thus, feature extraction becomes crucial in representing the images more efficiently. By capturing the most essential information and patterns in frames, a solid foundation for effective learning is layed.

The self-supervised ViT, DINOv2 [51], will be employed to perform spatial feature extraction.

4.2.1 Reasons for Selection

Although DINOv2 is highly competitive in the domain of visuals and can be used in tasks like image classification, instance retrieval, and more fine-grained pixel visual tasks like depth estimation, it has the capabilities to be a strong model for spatial feature extraction as proven as the backbone for a simple classifier [51]. Since it has been trained by self-supervised learning, there is no need for fine-tuning any parameters, and the good performance is ready to be exploited right out of the box. Additionally, investigating how well classification can be done using spatial features extracted by a ViT is intriguing, particularly considering the success of transformers with text data.

4.2.2 Architecture

In this thesis, through the use of transfer learning described in Section 2.5.2.4, a pre-trained model provided by Hugging Face [11] will be utilized. This model, known for its extensive scale, comprises 304 million parameters. It is structured starting with an initial embedding layer, followed by 23 encoder layers.

Every frame is processed through the model to extract meaningful feature embeddings, such as pattern recognition, curvatures, or shapes. Before this, it is crucial to ensure compatibility between the frames and the model. This compatibility is achieved using an image processor integrated with the pre-trained model. The processor prepares the input images by resizing them to match the model's input size, 224x224, using scaling and centre cropping, but also normalizing the colour channels and converting to PyTorch.

After completing the preprocessing steps, the frames are ready to be fed into DINOv2. Upon input, DINOv2 divides each image into 256 equal-sized patches, effectively tokenizing the image into smaller, more manageable pieces for analysis. This process allows for a more detailed examination of the image content. Following this, the model generates two outputs: a list of the last hidden states and another of the pooler outputs for all the processed images. The last hidden states provide the final layer representations for all the image tokens.

A 1024-dimensional vector representing each patch's local features is calculated, resulting in a 256x1024 output matrix for each frame. Moreover, an additional token plays a crucial role in this research, adding another dimension of analysis.

Analogous to using transformers in text analysis, a CLS (classification) token is generated and appended to the sequence of image patch tokens. This token acts as a comprehensive representation of the image, encapsulating its global features. The CLS token occupies the foremost position in the list of last hidden states.

The pooler output, obtained from the CLS token and subsequently processed with layer normalization, is available for further analysis. However, in this context, this will not be utilized.

4.2.3 Implementation

DINOv2 and an image processor with their functionalities were used through Hugging Face's Transformers library, renowned for offering SOTA ML solutions. The methodology involved systematically processing video data, organized in a sorted order based on unique clip identifiers (clip IDs) as mentioned in 4.1.

For the extraction, a structured approach was adopted and video frames was grouped into batches with a batch size of 32 frames. These batches were then processed using DINOv2, which calculates and extracts the last hidden states and the pooler outputs. A pivotal part of the analysis was extracting the CLS token from the last hidden states for each frame. This process generated a tensor that encapsulated the CLS tokens derived for all the frames in the TACDEC dataset serving as a representation for the frames in the experiments. From this point forward, frames and CLS tokens may be interchangeably referred to. However, it should be noted that the CLS tokens are utilized as representations for the frames.

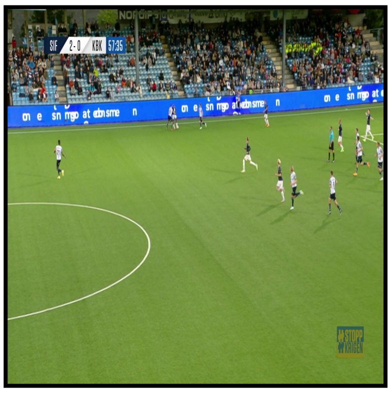
The entire feature extraction workflow was executed within Jupyter Notebooks, leveraging the PyTorch framework [53]. This choice not only facilitated a smooth execution process but also ensured that the methods would be easily replicable by other researchers. The code and processes we developed are publicly accessible for review and use [1].

To maximize the efficiency of the feature extraction, NVIDIA DGX-1 equipped with Tesla V100 GPUs will be used. This high-performance computing environment significantly accelerates data processing by offering 512 Gigabytes of memory, 40,960 NVIDIA CUDA cores, and 5,120 Tensor Cores.

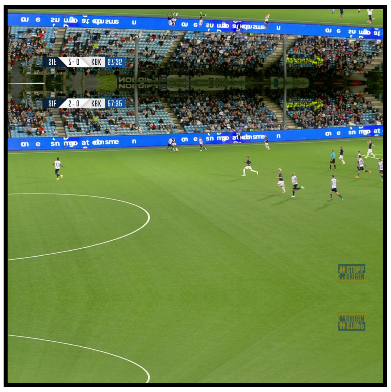
These resources will be available and used throughout the research, with all the code to take place in Jupyter Notebooks and using PyTorch.

[1] https://huggingface.co/SimulaMet-HOST/TACDEC-model/blob/main/feature_extraction.ipynb

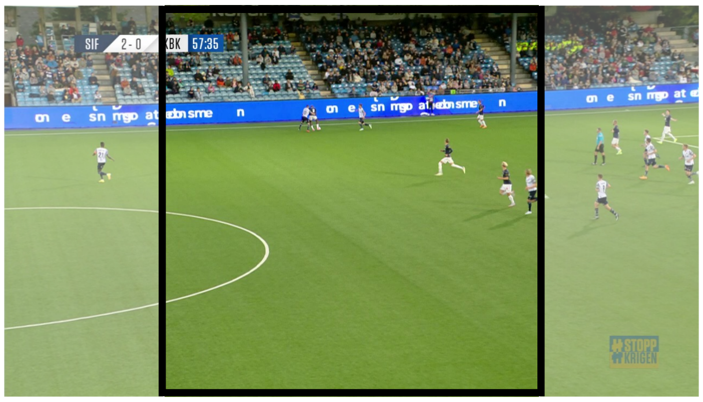
4.3 Cropping Done with the Image Processor



(a)



(b)



(c)

Figure 4.1: Illustratrion of how the three crops are executed with stretched crop (a), reflected crop (b) and the default centre crop (c).

Section 4.2.3 noted that DINOv2 is equipped with an image processor that applies a centre-crop to ensure the input data conforms to DINOv2 requirements. However, when this centre-crop is used on images that are not squared, information is inevitably lost. Since TACDEC contains broadcast videos with a 4:3 aspect ratio, valuable data might be excluded. Therefore, all experiments will be conducted using three different methods to modify the frames before feeding them to the image processor.

Figure 4.1 illustrates the various cropping methods, which are described in greater detail below:

(a) **Stretched Crop:** - the frames fed are stretched vertically to get a 1:1 aspect ratio

(b) **Reflected Crop:** - the frames fed are with horizontal borders vertically reflected to get a 1:1 aspect ratio

(c) **Centre Crop:** - the frames fed are raw and they are centre-cropped

The desire to conduct this research was further backed up by seeing that leading researchers within computer vision for soccer use centre-cropping [64, 65].

4.4 Spatial Approach

Two dissecting methods will be presented for capturing the features of tackles, which will described in separate sections. The following section will present a spatial approach, focusing on the features present within the frames themselves.

4.4.1 Meta-learner

This specific model will classify individual video frames and determine whether they are part of a tackle sequence and which one. An ensemble learning approach, see Section 2.5.2.5, will be used to enhance its accuracy and reliability. This technique involves combining multiple base models, each with its unique strengths and weaknesses, into a cohesive system. A meta-model, central to this approach, will analyze and leverage the collective insights from these base models to make more refined and accurate classifications. This strategy aims to harness the complementary capabilities of the base models, thereby improving the overall performance and robustness of the frame-based model in identifying tackle events within video frames.

4.4.1.1 Reasons for Selection

The concept of 'learning to learn' underpins our decision to utilize a meta-learner in our research. This approach allows us to assign a specific base model to each class, capitalizing on the shared characteristics among different classes while also addressing their unique aspects. Given that some classes, such as 'background' and 'tackle-live,' bear close resemblance to each other, it becomes crucial to tailor models that can effectively differentiate between them. By deploying distinct models for individual classes, the precision of the classifiers will hopefully be enhanced, as each base model is fine-tuned to recognize the subtle distinctions and commonalities within its designated class. This strategy aims to optimize the overall performance of the meta-learner, ensuring it benefits from the specialized insights and capabilities of each base model to achieve a more accurate and nuanced classification across the dataset.

4.4.1.2 Architecture

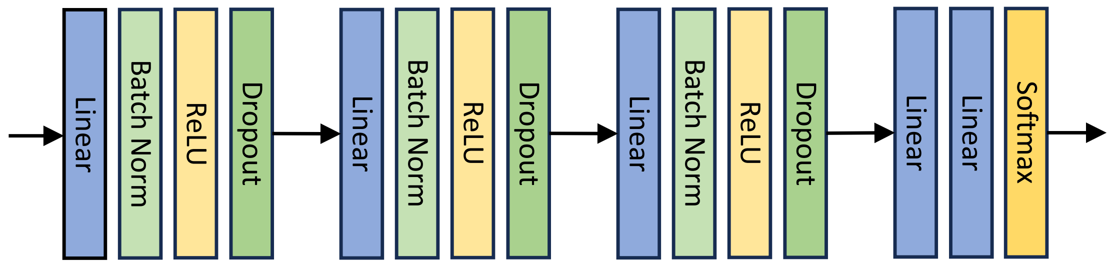


Figure 4.2: Meta learner architecture.

The meta-learner will be designed as a sequential NN that includes several layers aimed at understanding the strengths of the base models' algorithms. It will feature three linear layers, each succeeded by batch normalization and ReLU activation. The model then ends with a fully connected output layer, which utilizes softmax activation for its output. For a visual representation, see Figure 4.2.

4.4.1.3 Implementation

The idea for this model is to learn and understand when to trust each of the three models. As 'experts' in each class, the aim was to use their predictions and train meta-learners based on these alongside the ground truth. This was done by first running predictions from all the models on a validation set with data kept completely separate from the training data. These predictions were then stored as probabilities to ensure that each base model's output was kept on the same scale. Then, for each sample in the validation set, the predictions were concatenated together. Once the predictions of the three models and their ground truth labels were obtained, they could be split into training, validation, and test sets, respectively 70

The training was then conducted in an environment with an early stopping mechanism. The idea behind this is to halt the training process when validation loss increases over a predetermined number of epochs. Early stopping is one of the simplest yet most effective methods of regularization to prevent overfitting by ceasing training when the generalization error, as indicated by validation loss, begins to rise.

To better equip the model during training, Kaiming initialization was opted for setting the initial weights in every linear layer, as this method is particularly suitable for networks employing ReLU activations [26]. This choice was informed by the proven effectiveness of Kaiming initialization in accounting for the characteristics of ReLU activations, unlike the Xavier/Glorot method [20], which does not specifically cater to these activations. Employing Kaiming initialization ensured more stable gradients and facilitated faster and more reliable convergence of the models.

Furthermore, the models were optimized for three hyperparameters, which include the learning rate (η), the hyperparameter for L1-regularization (λ_1) deciding the L1-regularization strength, and the weight decay hyperparameter of the optimizer (λ_2) regulating L2-regularization strength.

The learning rate was decreased when the validation loss started to fluctuate or see no improvement to try to exploit the model learning capabilities to the fullest using a learning rate scheduler.

The use of a batch size of 64 was consistent for all experiments involving the meta-learner as this gave the most stable learning process. It is important to note that this batch size configuration was specific to the meta-learner and did not extend to the training of the base models.

4.4.2 Base Models

Since the strategy involves using multiple base models for the meta-learner to combine and enhance performance, it was decided that these models should be similar in structure. However, to capitalize on their individual strengths, the training process for each will be adjusted to focus on different aspects, with each model targeting a specific class. This approach aims to diversify the capabilities within the ensemble, allowing the meta-learner to leverage distinct advantages from each base model.

4.4.2.1 Reasons for Selection

Different base models were chosen, each trained to become 'experts' in specific classes, with the hope that this approach enhances the overall system's accuracy and robustness by harnessing specialized knowledge in each area. The decision to have these models share a common architecture supports consistency and simplicity, facilitating maintenance and promoting a clearer understanding of the overall model behaviour. This shared architecture also streamlines the integration process within the meta-learner framework, enabling more straightforward interpretation and adjustment of individual model contributions to the collective decision-making process.

4.4.2.2 Architecture

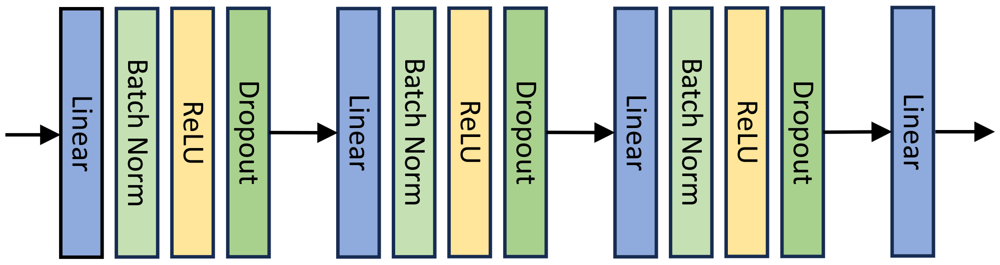


Figure 4.3: Base model architecture.

The base model is structured to include three linear layers, each followed by a dropout layer and batch normalization to enhance generalization and prevent overfitting. Initially, the model will receive input from the CLS-token of DINOv2, which has a size of 1024, as detailed in Section 4.2. This input size will first be reduced to 64 through the initial linear layer. Subsequently, batch normalization will be applied, where the output from the activations is normalized by subtracting the batch mean and then dividing it by the batch standard deviation to stabilize and accelerate the learning process. A ReLU activation will then be used as the activation function. Before the output passes through a dropout layer with a 40% dropout probability, meaning there's a 40% chance of dropping each node during training,

This configuration will be mirrored in the second layer, where the node size is further reduced from 64 to 32, followed by batch normalization and a ReLU activation, and then the dropout layer, where the dropout probability will be slightly reduced to 30%. The third layer continues this pattern, reducing the node size from 32 to 16, with a further reducing dropout probability of 20%.

The final part of the model will consist of a linear layer that transforms the 16 features into the raw scores (logits), corresponding to the number of classes in the model. The reason to not use a Softmax-activation on this layer, is due to the desired use of the cross-entropy loss function, which applies this internally. This architecture design ensures a gradual reduction in dimensionality, promoting efficient learning and robust feature extraction specific to each class.

Figure 4.3 shows a visualisation of the proposed architecture.

4.4.2.3 Implementation

To implement these base models and create three separate experts, through the training process, a skewed dataset was offered that was skewed towards the models' target class. That means that there were three different training, validation and test sets, where the three base models had their respective ones. To put this more into perspective, the datasets were structured to have double the number of instances for their target class. The validation set and test set contained instances completely separate from the training data, and they were also balanced. Respectively, the dataset was divided into splits of 70%/15%/15%.

To train the base models, the models were optimized for three hyperparameters. These hyperparameters were the learning rate (η), the hyperparameter for L1-regularization (λ_1), and the weight decay hyperparameter of the optimizer (λ_2) with the same purposes as for the meta-learner but with the target of improving the class-wise F1-score for the target class. The weights were also initialized using Kaiming initialization, as for the meta-learners.

Early-stopping was also implemented to prevent overfitting and a learning rate scheduler was used to decrease the learning rate, hoping that some smaller steps would help move a bit closer toward a local optimum.

4.4.3 Proposed Evaluation

To evaluate both the meta-learners and the base models, a baseline will be established by training a base model on a balanced dataset. After optimization using the same hyperparameters as presented for both models, this model will serve as a benchmark for comparing. Confusion matrices will be created from inference on a test set kept completely separate from training and validation. In addition to this, metrics such as accuracy, precision, recall, and F1-score, described in Section 2.5.4, will be provided to evaluate and compare the performances. ROC curves will also be utilized for comparing the meta-learners as these are up for evaluation.

4.5 Temporal Approach

The second approach for detecting tackles involves focusing on temporal aspects. This method is similar to using spatio-temporal features, as detailed in Section 2.4.2. Unlike the previous in Section 4.4, which was based solely on spatial features represented by the CLS-tokens, the aim is here to identify temporal features among these CLS-tokens without further processing the spatial data.

4.5.1 TempTAC

The second model proposed in this thesis, TempTAC, will be created to capture features that hide between frames, such as movement or camera shifts, the temporal features. These are features that are not represented when analysing individual frames. To do so, an LSTM-based model will be implemented as it is capable of memorizing earlier inputs and using these to make predictions. For instance, if the previous frame seen was classified as a tackle, the model might learn that there is a higher probability of the next one also belonging to that class if there are features implying so. By already having spatial features extracted in the CLS-tokens from DINOv2, these will be used in sequences in order to mimic a spatio-temporal feature extraction will be done.

4.5.2 Reasons for Selection

In analyzing sequences such as tackle events, which inherently unfold over time, the temporal characteristics embedded within these sequences might prove crucial for accurate classification. RNNs, and especially LSTMs models excel in capturing these temporal dynamics due to their ability to remember information over extended periods. The distinction in motion speed between replays and live events can serve as a significant factor in this context. Specifically, replays often feature slower motion than live events, increasing similarity among consecutive frames within a sequence. This characteristic potentially enhances the LSTMs' learning process by providing it with more granular yet coherent temporal information to analyze. By effectively leveraging these temporal features, the LSTM-based approach is more suited to distinguish and accurately classify frames within tackle sequences, taking into account both the spatial features extracted with DINOv2 and the temporal features.

4.5.3 Architecture

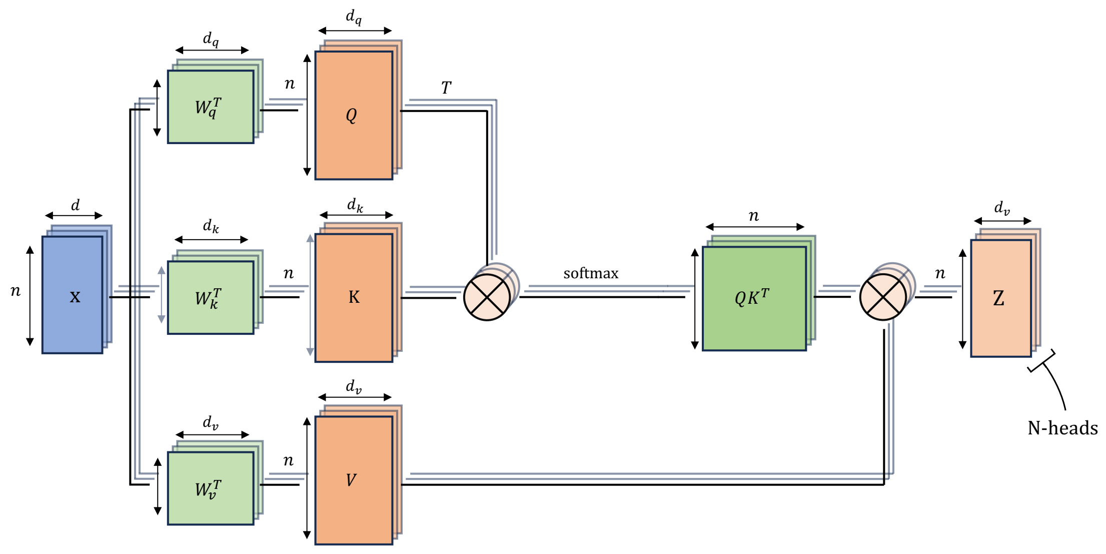


Figure 4.4: Illustration of several (N)-heads with self-attention is calculated.

The architecture of the TempTAC consists of several mechanisms trying to enhance the temporal understanding of tackles. The model consists of a bi-directional LSTM module with two layers equipped with 512 hidden nodes each. The first LSTM layer has a dropout probability of 0.5 to prevent overfitting by preventing the nodes from adapting too much. Self-attention is applied to the whole sequence along the time dimension in order to manipulate the sequence of CLS-tokens in a way that highlights certain timesteps over others based on the values that are represented in the CLS-tokens.

To apply self-attention, linear transformations are done to create query, key and value pairs from the input sequence. The attention is calculated in every head that is available in the multihead attention layer. Deciding to use 8 heads, experiments will be conducted with window sizes of 25, 50, and 75. With 8 heads, the aim is to capture different temporal dynamics, where shorter windows might reveal more immediate, local patterns, while longer windows can attend to broader features. The way this is calculated can be seen in Figure 4.4, where the outputs are concatenated, allowing different heads to focus on different aspects of the input sequence, some focusing on long-term dependencies and others on short-term. A more detailed description can be found in Section 2.5.1.9.

When the attention weights are calculated, they are multiplied element-wise with the input sequence, scaling each feature vector with the corresponding frame's attention weight.

With every frame in the sequence weighted with attention, these can soon be fed to the LSTM. This has to be done for every timestep, but before doing so, the predictions made from the previous timestep need to be concatenated with the input of the current timestep. This is done because sequences are involved; therefore, if the previous frame was a tackle, the probability of the next one being so is inherently increased.

The previous inputs are first weighted with a learnable weight to optimize past information's influence on the current state as it gives more control over the memory by dynamically deciding the significance of previous predictions learned by the patterns within the sequences.

Then after concatenating the current timestep and the weighted previous output, these can finally be fed to the LSTM together with the hidden state. The hidden state represents two values. The output of the LSTM from the previous timestep, the hidden state, reflects the short-term memory of the network by carrying information onto the next time steps. Notably, this is overwritten at every step. The other value it holds is the cell state which carries the long-term memory, the core memory, allowing it to maintain information over longer sequences. The cell state is as described in Section 2.5.3.5, updated by three different gates that determine what to forget, keep, and pass on.

After going through the LSTM, the output is fed through a dropout layer before a linear layer to obtain a class prediction for the frame.

These weights will be initialized using Kaiming initialization.

4.5.4 Implementation

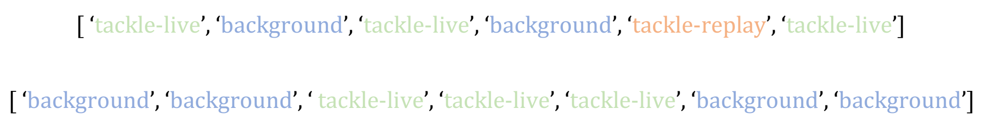

['tackle-live', 'background', 'tackle-live', 'background', 'tackle-replay', 'tackle-live']
['background', 'background', ' tackle-live', 'tackle-live', 'tackle-live', 'background', 'background']

Figure 4.5: Illustration of order in a sequence (bottom) and unordered sequence (top).

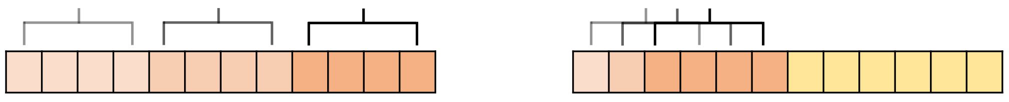
For implementing TempTAC, Jupyter Notebooks, PyTorch and the NVIDIA GPUs were used as for the spatial approach. Because learning the temporal dependencies between frames was necessary, the order of the frames was of high importance. It was essential to ensure that the frames fed to the model were ordered so that the input either contained a full sequence or segments of a full sequence, uninterrupted by other sequences, as illustrated in Figure 4.5. Before doing this, the games had to split into train, validation, and test sets to prevent data leakage and see sequences of the same tackles in separate sets.

To comply with the order criteria, labels were used to find the indices of tackles and extract the sequences. Given the imbalance in the dataset, a decision to undersample 'background' sequences was made. This was done by first finding the indices of 'background' sequences that have sufficient length and do not overlap other classes, followed by randomly selecting a start frame that will be accompanied by a sequence length set by a fixed-size window.

When sequences for both tackle classes and the background were obtained, data loaders were created to feed TempTAC sequences of frames. When creating the data loader, the decision had to be made regarding how to feed the frames:

- **Shifting window:** Window shifts/jumps over the concatenated collection of sequences. In the next iteration, the frame after the previous end frame, is the new start frame; see Figure 4.6a.
- **Sliding window:** Window slides over the concatenated collection of sequences. In the next iteration, all but the first are present again with a new slid-in; see Figure 4.6b.

For both of these windows, the length of the window could be changed. The decision was that experiments would be conducted using both.



(a) A shifting window of size 4.

(b) A sliding window of size 4.

Figure 4.6: A visualization of a shifting and a sliding window iterating in three steps over an input sequence.

For training the model, early stopping and a learning rate scheduler were implemented, similar to the spatial approach. Additionally, a weighted loss function, with weights determined by class representation and gradient clipping (*used to mitigate the problem of exploding gradients in NN by capping the gradients during backpropagation to maintain a defined range*) was employed to prevent overfitting. This approach was necessary because the number of frames could not be balanced due to the varying lengths of sequences not adding up, and it was desired to utilize as many full tackle sequences as possible while maintaining a balance. The data was divided into splits for training, validation, and testing with a split of 70

When optimizing hyperparameters to facilitate a better learning process, the same parameters were optimized as in the spatial approach (learning rate (η), L1-regularization strength (λ_1), L2-regularization strength (λ_2)).

4.5.5 Proposed Evaluation

To evaluate the TempTAC models, a simple CNN-based model will be created to serve as a baseline. The decision to use a CNN as the baseline was made due to CNNs being very efficient and simple to implement. Not only is it efficient in terms of computational resources, but it also learns quickly as CNNs are equipped with fewer parameters compared to RNNs. With the convolutional filter running along the time dimension, a feature extraction from the sequences is expected, similar to how they extract spatial hierarchies in images. This CNN will consist of one convolutional layer with a kernel size of 3 and padding (The amount of padding applied to the input before convolution-used to increase the dimensions of the output.) and stride of 1 (Indicating how much it slides between convolutions-used to reduce the resolution.), followed by one fully connected layers with ReLU activation. This simple CNN will be optimized with respect to the same parameters as for the baseline and meta-learner (η, λ_1, λ_2), while regular parameters such as nodes, filter sizes, and other internal model parameters are fixed. Accuracy, precision, recall, and F1-score, as described in Section 2.5.4, will also be used to evaluate TempTAC models against each other based on the different cropping methods described in Section 4.3, as well as against the best spatial approach model.

4.6 Optimizing Hyperparameters

To assess the impact of minor modifications in a preprocessing step before feature extraction, employing fixed models is crucial. This approach allows for determining whether these adjustments to the input images can enhance the models' learning capabilities, specifically in the context of DINOv2. For this purpose, Optuna [2], a sophisticated hyperparameter optimization framework, was utilized to fine-tune the models' hyperparameters with the original, unaltered images only preprocessed by the image processor. The results gained from these will serve as a baseline to compare with. A random state was set throughout the environment to ensure a controlled environment to accurately gauge the effect of preprocessing changes on the models' performance and that the models learn using the same set of frames.

It is important to highlight that the optimization process for the base models in Section 4.4 will be specifically geared towards maximizing the F1-score for their respective classes.

The choice of the F1-score as our focus metric stemmed from its ability to provide a balanced measure of model performance. Unlike other metrics, such as precision, recall, or accuracy, which can be artificially inflated by models overpredicting the dominant class, the F1-score offers a more balanced assessment. This goes for all models optimized in this thesis.

4.7 Chapter Summary

In this chapter, two primary methodologies for detecting tackles in soccer broadcast videos were presented. The first methodology exclusively considers spatial features, while the second incorporates both spatial and temporal dimensions. The architectures of all models employed are detailed, implementation strategies are explained, and the reasoning behind the choices is discussed. Additionally, the use of DINOv2 for spatial feature extraction is described, and how the TACDEC dataset, curated and detailed in Chapter 3, will be used. In the next chapter, the experiments will be reviewed, and the results obtained from this methodology will be analyzed.

Chapter 5

Experiments and Results: Spatial Approach

As detailed in Section 4.4 in Chapter 4, three simpler models were implemented, each targeting a specific class over the others. In this chapter, the results for each of these three models, alongside the meta-learner trying to combine their strengths, will be dissected to provide a comprehensive evaluation. This analysis will cover three distinct approaches, where a meta-learner is trained upon each of these feature extraction approaches.

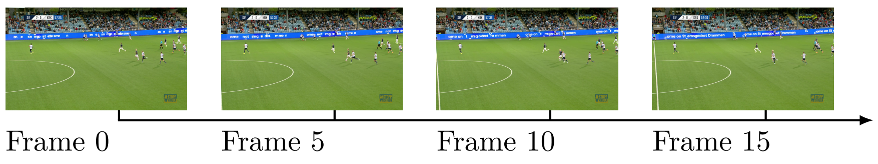
Experiment Overview

To provide a clearer overview of the experiments discussed below, Table 5.1 presents a tabular summary. Previous research, particularly in the field of soccer analysis, has extensively used centre cropping for image preprocessing [64, 65]. Recognizing that centre cropping can exclude important information, such as tackles occurring outside the central field of view, the impact of centre cropping was explored. Thus, alternative methods to enhance the information retained in the frames used for feature extraction were investigated.

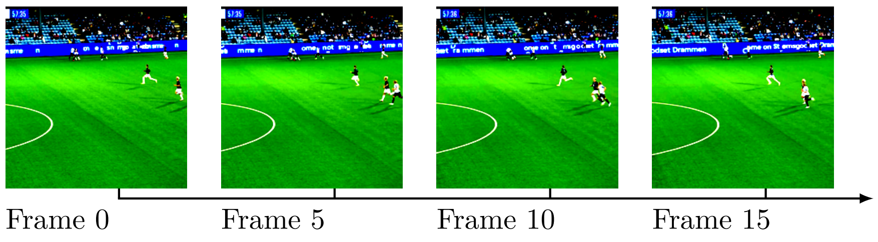
Section	**Experiment**	**Description**	**Aspect Ratio**	**Full Image**
5.1	Raw	Results from experiments where raw frames were centre-cropped, possibly losing important information.		
5.1.1	Base model	Results from a baseline model using CLS-tokens from centre-cropped frames.	4:3	✗
5.1.2	Model 1	Results from model targeting 'background' using CLS-tokens from centre-cropped frames.	4:3	✗
5.1.3	Model 2	Results from model targeting 'tackle-live' using CLS-tokens from centre-cropped frames.	4:3	✗
5.1.4	Model 3	Results from model targeting 'tackle-replay' using CLS-tokens from centre-cropped frames.	1:1	✗
5.2	Stretched	Results from experiments where frames were stretched before centre-cropping to preserve information.		
5.2.1	Model 4	Results from model targeting 'background' using CLS-tokens from stretched frames.	1:1	✓
5.2.2	Model 5	Results from model targeting 'tackle-live' using CLS-tokens from stretched frames.	1:1	✓
5.2.3	Model 6	Results from model targeting 'tackle-replay' using CLS-tokens from stretched frames.	1:1	✓
5.3	Padded	Results from experiments where frames were reflection-padded before centre-cropping to preserve information.		
5.3.1	Model 7	Results from model targeting 'background' using CLS-tokens from reflective padded frames.	1:1	✓
5.3.2	Model 8	Results from model targeting 'tackle-live' using CLS-tokens from reflective padded frames.	1:1	✓
5.3.3	Model 9	Results from model targeting 'tackle-replay' using CLS-tokens from reflective padded frames.	1:1	✓

Table 5.1: An overview of the different experiments and sections in this chapter that were conducted using spatial features. The blue rows indicate a shift in how the frames were altered prior to the image processor that performs centre-crop.

5.1 Centre Cropping



(a) Frames prior to processing.

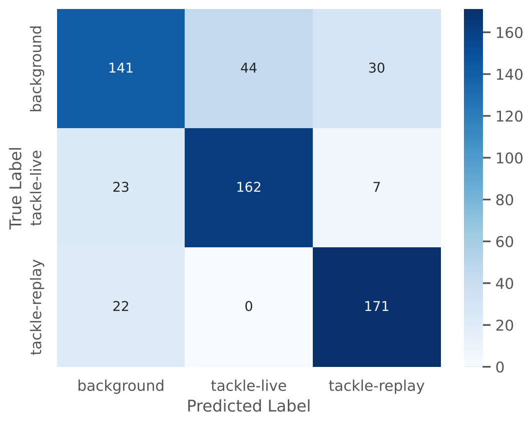


(b) Frames post processing.

Figure 5.1: Comparison of raw 'centre cropped' frames before and after processing.

In examining feature extraction using DINOv2 together with the image processor, the functionality of both components was delved into in detail. It was observed that the image processor implemented centre cropping on the frames, as illustrated in Figure 5.1b. This cropping is necessitated by DINOv2's requirement for input dimensions of 224 x 224 pixels, necessitating a match in the aspect ratio. Consequently, the image processor adjusted the frames to a quadratic shape. This adjustment inevitably leads to the exclusion of some visual information, which, given videos from TACDEC originally using a 4:3 aspect ratio, depicted in Figure 5.1a, could lead to a loss of information. In Section 5.2 and 5.3 the results using two different ways of modifying these frames prior to the image processor is presented.

5.1.1 Baseline Model



(a) Confusion matrix.

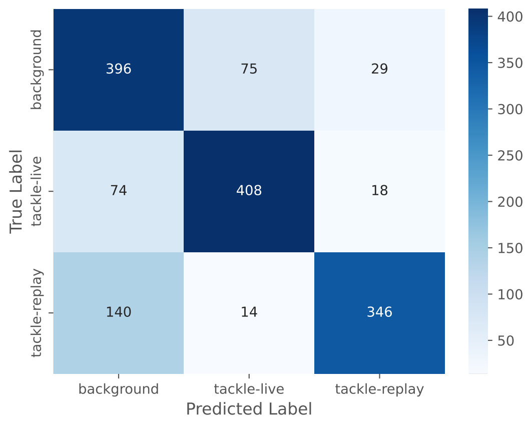
Event	Precision	Recall	F1-score	Support
Background	0.76	0.66	0.70	215
Tackle-live	0.79	0.84	0.81	192
Tackle-replay	0.82	0.89	0.85	193
Macro avg	0.79	0.80	0.79	600
Weighted avg	0.79	0.79	0.79	600
Accuracy			**0.79**	600

(b) Evaluation metrics.

Figure 5.2: Confusion matrix and evaluation metrics for the baseline ran on the same test set as meta-learners will use.

Forming a baseline for evaluation is important. To do so, the same architecture as for the base model was used, but instead of targeting a class, it was run on a balanced dataset while attempting to improve the overall F1-score, balancing precision and recall. The performance from this baseline can be seen in Table 5.2b and Figure 5.2a.

5.1.2 Model 1



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.65	0.79	0.71	500
Tackle-live	0.82	0.82	0.82	500
Tackle-replay	0.88	0.69	0.77	500
Macro avg	0.78	0.77	0.77	1500
Weighted avg	0.78	0.77	0.77	1500
Accuracy			**0.77**	1500

(b) Evaluation metrics.

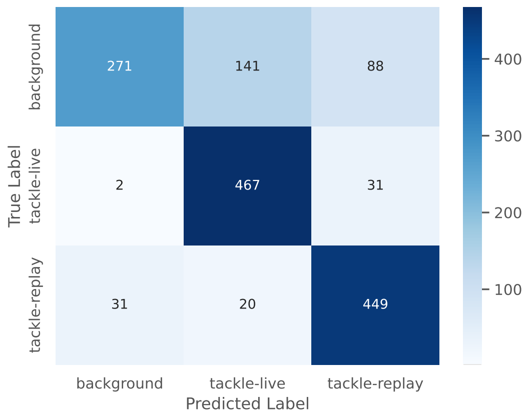
Figure 5.3: Confusion matrix and evaluation metrics for Model 1.

The confusion matrix in Figure 5.3a shows that the model accurately classifies 396 True Positives (TPs) out of the 500 frames for 'background.' However, it also has 214 false positives (74 'tackle-live' and 140 'tackle-replay'). The tendency to over-predict likely originates from its training on an imbalanced dataset, with the target class having twice as many samples compared to others. This imbalance, combined with a specific focus on optimizing the F1-score for the target class encourages the model to improve both precision and recall primarily for ’background.’ As a strategy to maximize both the class-wise and overall F1-score, the model may be predisposed to predict ’background’ more often to enhance recall or as an acceptable compromise in precision.

Despite the misclassifications, it achieves a recall of 0.79 for its target class, as indicated in Table 5.3b, beating the baseline. Although the model successfully predicts the ’background’ category with 79% accuracy, its tendency to overpredict this class results in a precision score of just 0.65, below the baseline. This discrepancy impacts the balance between precision and recall, as reflected in the F1-score of 0.71.

The model, despite being optimized for the ’background’ metric, excelled in predicting ’tackle-live’ with precision, recall, and F1-score each at 0.82, suggesting it inherently predicts this category more accurately, with 74 false negatives as ’background’ and 18 as ’tackle-replay.’ Meanwhile, ’tackle-replay’ was correctly identified 88% of the time with a precision of 0.88, although its recall was lower at 0.69 due to many instances being misclassified as ’background,’ resulting in an F1-score of 0.77.

5.1.3 Model 2



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.89	0.54	0.67	500
Tackle-live	0.74	0.93	0.83	500
Tackle-replay	0.79	0.90	0.84	500
Macro avg	0.81	0.79	0.78	1500
Weighted avg	0.81	0.79	0.78	1500
Accuracy			**0.79**	1500

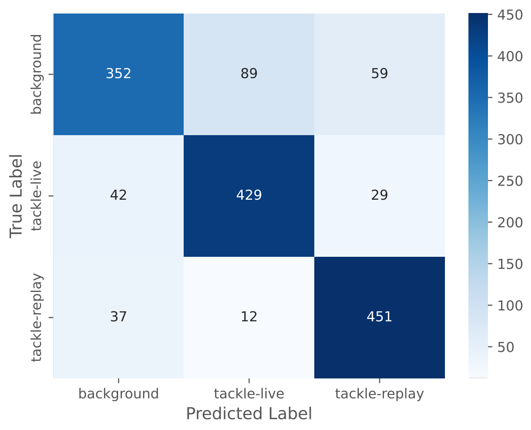
(b) Evaluation metrics.

Figure 5.4: Confusion matrix and evaluation metrics for Model 2.

Model 2, targeting ’tackle-live,’ effectively captures 467 TPs, achieving an impressive recall score of 0.93, as detailed in Table 5.4b and illustrated in Figure 5.4a. However, the model tends to overpredict its target class, classifying 141 FPs as ’background’ and 20 as ’tackle-replay,’ resulting in a precision score of 0.74. Consequently, Model 2 achieves an F1-score of 0.83 for its target class, higher than the baseline.

Model 2 demonstrates high precision (0.89) but low recall (0.54) in predicting ’background,’ resulting in an F1-score of 0.67, likely due to difficulty distinguishing between similar ’background’ and ’tackle-live’ images. For ’tackle-replay,’ the model shows better performance with an F1-score of 0.84, successfully identifying 90% of instances (449 TPs) and exhibiting a precision of 0.79 despite 199 FPs.

5.1.4 Model 3



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.82	0.70	0.76	500
Tackle-live	0.81	0.86	0.83	500
Tackle-replay	0.84	0.90	0.87	500
Macro avg	0.82	0.82	0.82	1500
Weighted avg	0.82	0.82	0.82	1500
Accuracy			**0.82**	1500

(b) Evaluation metrics.

Figure 5.5: Confusion matrix and evaluation metrics for Model 3.

In the third base model that targets ’tackle-replay’, an examination of the confusion matrix in Figure 5.5a and Table 5.5b reveals that it achieves the highest F1-score for its target class among all three models. It successfully identifies 451 TPs for ’tackle-replay,’ with 37 and 12 FNs misclassified as ’background’ and ’tackle-live,’ respectively, resulting in a recall score of 0.90. This slightly outperforms Model 2 for the same class. Furthermore, with only 59 FPs as ’background’ and 29 as ’tackle-live,’ the model achieves a precision score of 0.84.

Although the model was optimized and trained with a focus on ’tackle-replay,’ it performs pretty well on the other two, with an F1 Score of 0.76 for ’background’ and 0.83 for ’tackle-live.’ For ’background,’ it has a precision score of 0.82, higher than what Model 1 had, but it is still unable to capture as many with a recall score of 0.70. Gaining the same F1 Score for ’tackle-live’ as Model 2, it has a higher precision score of 0.74 but a lower recall score of 0.86 compared to Model 2’s 0.93.

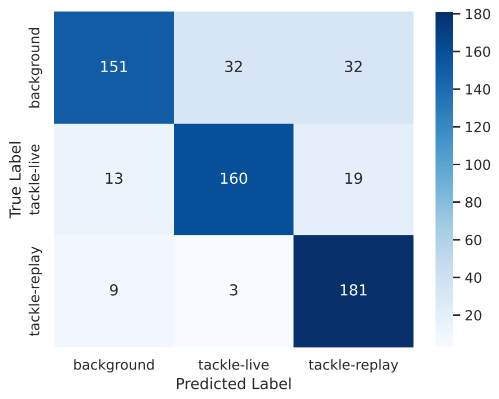
Model 3 beats the baseline on all metrics for all classes.

5.1.5 Meta-learner 1-3

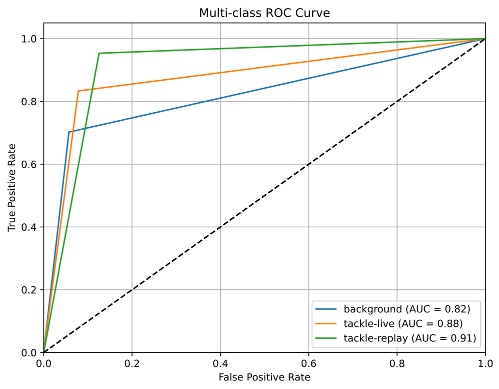
Event	Precision	Recall	F1-score	Support
Background	0.87	0.70	0.78	215
Tackle-live	0.82	0.83	0.83	192
Tackle-replay	0.78	0.94	0.85	193
Macro avg	0.82	0.82	0.82	600
Weighted avg	0.83	0.82	0.82	600
Accuracy			**0.82**	600

Table 5.2: Evaluation metrics for Meta-learner 1-3.

By looking through the evaluations of the base models above, we can see that all of them were able to beat the baseline in terms of the F1-score for their target class. When



(a) Confusion Matrix.



(b) ROC Curve.

Figure 5.6: Confusion Matrix and ROC Curve for Meta-learner 1-3 using CLS-tokens from centre-cropped frames.

their outputs were combined with a meta-learner to leverage their distinct strengths, the results, as depicted in Table 5.2 and Figure 5.6a, also surpassed the baseline in most metrics, but two.

For 'background,' the meta-learner demonstrated high precision, with 151 TPs out of 215, resulting in a precision score of 0.87. However, with 64 FNs, the recall score was lower at 0.70, yielding an F1-score of 0.78.

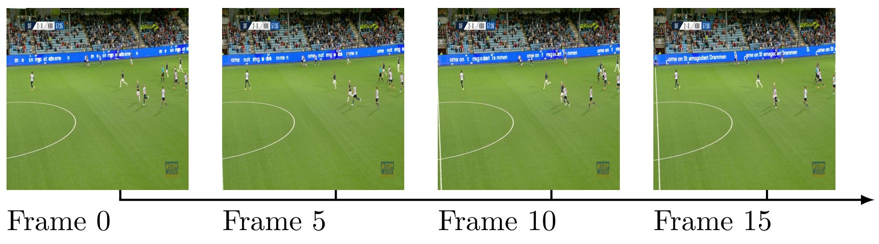
'Tackle-live' presented as the most balanced class, with the meta-learner achieving both a precision and a recall score of 0.82 and 0.83 respectively, leading to an F1-score of 0.83. This indicates effective classification, with only 32 FNs out of 192 and only 35 FPs. This suggests that even though Model 2 tended to over-predict 'tackle-live,' the combination of these models changed that behaviour.

The performance on 'tackle-replay' was also strong, particularly at the recall score of 0.94, with 181 TPs out of 193 and only 12 FNs. The 51 FPs reflect the precision score of 0.78. Despite this, the good recall skewed the F1-score to 0.85.

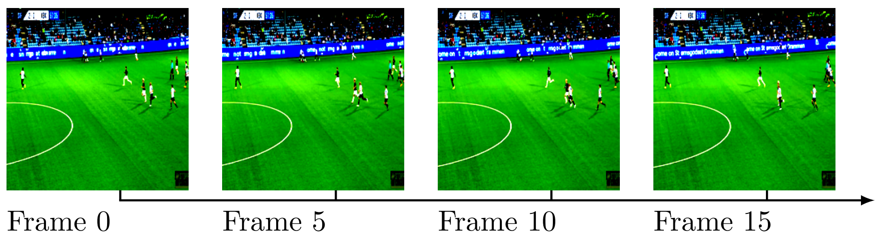
Comparatively, the baseline achieved a slightly higher recall for 'tackle-live' at 0.84 and a marginally better precision for 'tackle-replay' at 0.82. However, the meta-learner matched or surpassed the baseline in other metrics.

Notably, despite the meta-learner scoring lower in those two specific metrics, it demonstrated more evenly distributed accuracy and achieved higher overall macro and weighted averages across all evaluated metrics. This suggests that the combined model effectively utilizes the strengths of the individual models to improve overall performance.

5.2 Stretched Cropping



(a) Frames prior to processing.



(b) Frames post processing.

Figure 5.7: Comparison of 'stretched' frames before and after processing.

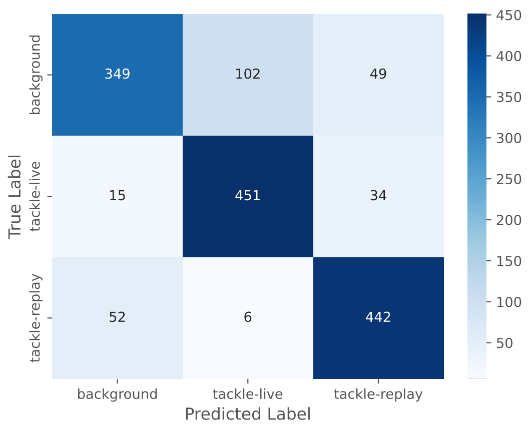
Figure 5.7a presents four distinct frames, selected at five-frame intervals, to illustrate their appearance prior to preprocessing for DINOv2. Here, they are already stretched, creating a squared appearance. This step was taken as the image processor for DINOv2 crops non-quadratic aspect ratios, as discussed in Section 5.1. Conversely, Figure 5.7b showcases the transformation of these frames post-preprocessing, where the whole scene is still to be seen. This might also mean that the feature vectors might carry more distinct information that can help distinguish between the classes.

To facilitate a comprehensive comparison of the base models discussed in this section, they will be evaluated against those described in Section 5.1 as well as the baseline model. This approach ensures that improvements or variations resulting from different processing applied to the frames can be effectively measured.

5.2.1 Model 4

Model 4, targeting 'background,' reflects progress in dealing with the challenge that, so far, has proven to be complicated. According to Figure 5.8a, Model 4 accurately identified 349 TPs, with 67 FPs resulting in a high precision as documented in Table 5.8b.

This model surpasses both Model 1 and the baseline in terms of F1-score, primarily attributed to its higher precision. Although Model 1 records a higher recall score of 0.79 compared to Model 4's 0.70, Model 4's precision advantage contributes to its superior F1-score. The confusion matrix reveals difficulty in distinguishing between 'tackle-live' and



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.84	0.70	0.76	500
Tackle-live	0.81	0.90	0.85	500
Tackle-replay	0.84	0.88	0.86	500
Macro avg	0.83	0.83	0.83	1500
Weighted avg	0.83	0.83	0.83	1500
Accuracy			**0.83**	1500

(b) Evaluation metrics.

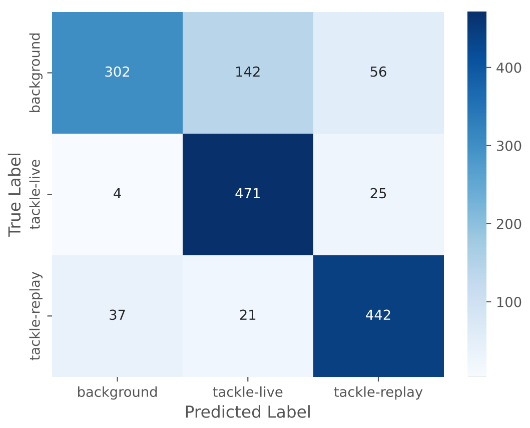
Figure 5.8: Confusion matrix and evaluation metrics for Model 4.

'background,' with 102 FNs incorrectly classified as 'tackle-live.' This highlights an area where Model 4 struggles despite its strong performance with precision for 'background,' likely due to the same reason we believe for Model 2.

For the 'tackle-live' and 'tackle-replay' classes, Model 4 records precision scores of 0.81 and 0.84 respectively. While these precision scores are lower than those achieved by Model 1, they surpass those of the baseline model. In terms of recall, Model 4 exhibits strong performance for 'tackle-replay' with a score of 0.90, surpassing both Model 1 and the baseline, although it slightly trails the baseline for 'tackle-live' recall. Nonetheless, it outperforms Model 1 in recall for 'tackle-replay,' with a score of 0.88 compared to 0.69.

Model 4 achieves a higher F1-score across all classes compared to its counterparts, with also the highest macro and weighted averages seen so far.

5.2.2 Model 5



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.84	0.70	0.76	500
Tackle-live	0.81	0.90	0.85	500
Tackle-replay	0.84	0.88	0.86	500
Macro avg	0.83	0.83	0.83	1500
Weighted avg	0.83	0.83	0.83	1500
Accuracy			**0.83**	1500

(b) Evaluation metrics.

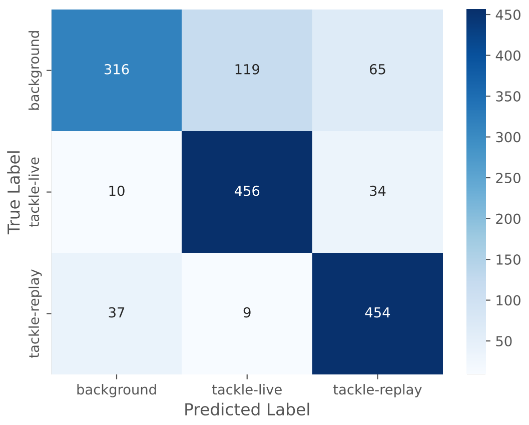
Figure 5.9: Confusion matrix and evaluation metrics for Model 5.

By observing the confusion matrix in Figure 5.9a and classification report in Table 5.9b for Model 5, which targets ’tackle-replay,’ we see that it performs exceptionally well by accurately identifying 471 TPs of the 500, achieving a remarkable score for recall of 0.94. However, its tendency to over-predict its target class is evident, with 142 FPs belonging to ’background’ and 21 to ’tackle-live’. Despite this high rate of correct identifications, the over-prediction results in a precision score of 0.74, which matches that of Model 2 and is lower than the baseline’s precision. Nonetheless, the high recall contributes to a still impressive F1-score of 0.83.

Like Models 2 and 4, Model 5 faces challenges in distinguishing between ’background’ and ’tackle-live,’ achieving a recall score of only 0.60 as there are too many FNs. This is higher than Model 2 but lower than the baseline. In predicting ’background,’ Model 5 makes 343 predictions and with 302 TPs, resulting in a precision score of 0.88, surpassing the baseline but slightly lower than Model 2.

Model 5 excels in its target class and demonstrates competitive performance for ’tackle-replay,’ effectively handling the tackle categories. However, like Models 1 and 2, it is hampered by a tendency to over-predict, which limits its overall effectiveness. This propensity for over-predicting and struggle to differentiate between ’background’ and ’tackle-live’ as a continuous problem affects its recall for ’background’ and undermines its potential to be a more balanced, reliable model. The over-predicting leads to diminished precision, but still allows for high recall and decent F1-scores within the targeted class.

5.2.3 Model 6



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.87	0.63	0.73	500
Tackle-live	0.78	0.91	0.84	500
Tackle-replay	0.82	0.91	0.86	500
Macro avg	0.82	0.82	0.81	1500
Weighted avg	0.82	0.82	0.81	1500
Accuracy			**0.82**	1500

(b) Evaluation metrics.

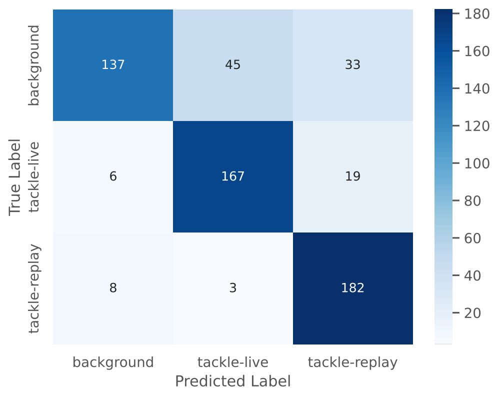
Figure 5.10: Confusion matrix and evaluation metrics for Model 6.

When moving onto the confusion matrix for Model 6, see Figure 5.10a, we can observe that for ’tackle-live’ and ’tackle-replay’ the model is able to capture 456 and 454 TPs, respectively. Doing so, Model 6 rightfully earns recall scores of 0.91 for both classes, reflected in Table 5.10b, with a precision score of 0.82 for its target class ’tackle-replay’ (65 Frames Per Seconds (FPSs) as ’background’ and 34 FPSs as’tackle-live’).

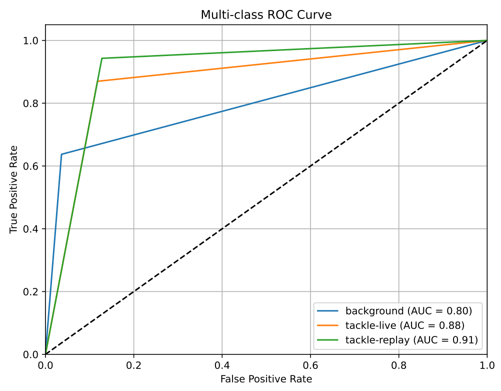
By matching the baseline model on precision and beating it by 0.02 on recall, Model 6 outperforms it on F1-score with a score of 0.86. Even though Model 6 also beats Model 3 on recall, failing to match it on precision makes it fall a bit behind in terms of the F1-score for its target class.

Once again, we see a problem for ’background’ with separating it from ’tackle-live’ with 119 FNs as ’tackle-live’. As a result, we see a poor score for recall of 0.63 for ’background,’ although the model has a precision score of 0.87, meaning it is mostly correct when predicting ’background,’ the F1-score is dragged down to 0.73 due to the model FNs.

5.2.4 Meta-learner 4-6



(a) Confusion Matrix.



(b) ROC Curve.

Figure 5.11: Confusion Matrix and ROC Curve for Meta-learner 4-6 using CLS-tokens from stretched frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.91	0.64	0.75	215
Tackle-live	0.78	0.87	0.82	192
Tackle-replay	0.78	0.94	0.85	193
Macro avg	0.82	0.82	0.81	600
Weighted avg	0.82	0.81	0.81	600
Accuracy			**0.81**	600

Table 5.3: Evaluation metrics for Meta-learner 4-6.

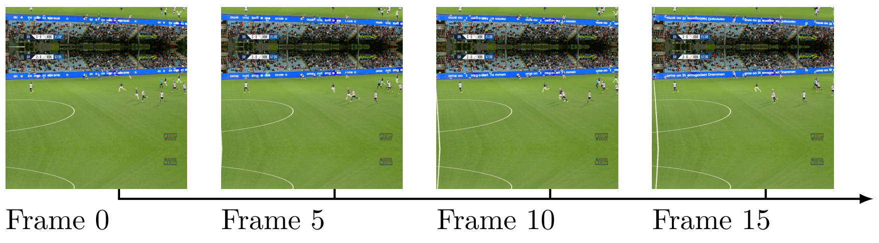
Meta-learner 4-6, utilizing the strengths of Models 4 through 6, does well in precision with a score of 0.91, as detailed in Table 5.3. It accurately identifies 137 TPs for 'background', see Figure 5.11a. However, its recall of 0.64 for 'background' is lower, having 78 FNs, primarily misclassified as 'tackle-live,' results in an F1-score of 0.75.

In 'tackle-live' performance, Meta-learner 4-6 shows an F1-score of 0.82, higher than Meta-learner 1-3 but with less balance between precision and recall. It misclassifies 19 FNs as 'tackle-replay' and 6 FNs as 'background'.

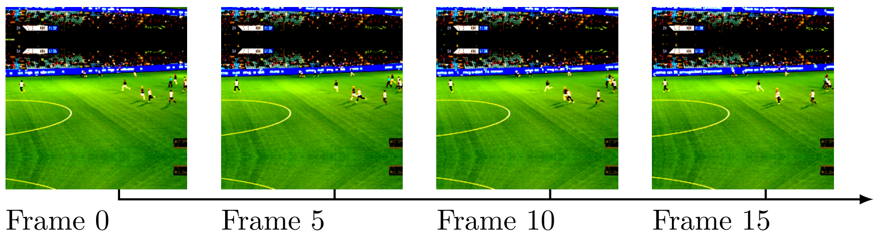
For 'tackle-replay,' Meta-learner 4-6 matches Meta-learner 1-3 in performance, with both showing a precision of 0.78 and a recall of 0.94, leading to an F1-score of 0.85. However, it slightly increases TPs and decreases FNs compared to Meta-learner 1-3.

Overall, Meta-learner 4-6 offers great precision for 'background' but falls short in recall, placing its overall performance between the baseline and Meta-learner 1-3. While it provides the highest recall for 'tackle-live' among compared models, its lower precision might be a concern if accuracy is prioritized. Similarly, for 'tackle-replay,' it shows robust recall but suffers from reduced precision, indicating a trade-off between identifying more true positives and accruing more false positives.

5.3 Padded Cropping



(a) Frames prior to processing.



(b) Frames post processing.

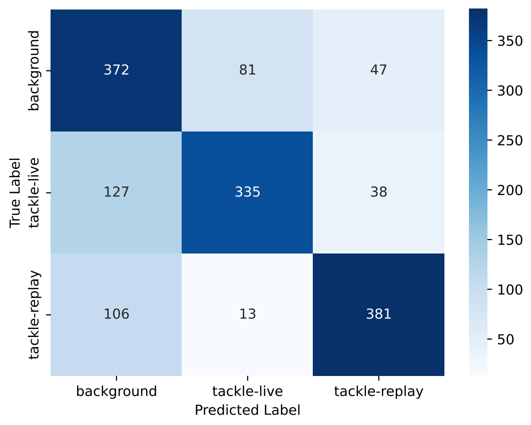
Figure 5.12: Comparison of 'border-reflected' frames before and after processing.

A third alternative method of image padding was explored. This technique was implemented so that the borders of the image are reflected onto the shortest dimension to produce a square image. How these frames looked prior to and post-preprocessing can be seen in Figure 5.12a and Figure 5.12b, respectively. The three base models were trained, following the same procedure as in the previous two sub-experiments, with the results of this to follow.

5.3.1 Model 7

Upon examining Figure 5.13a and the classification report in Table 5.13b, we can see that the overall performance of Model 7 is the poorest seen so far. This comes from the over-predicting for its target class 'background,' where it predicts 233 FPs. The explanation for this might lie in the way the frames appear when reflected upon themselves causing problems when over-represented. Even though the mutual information had a higher score overall, this can be closer related to the other two classes, making it harder to differentiate between the three classes. By predicting 'background' too often, the model might gain the highest F1-score, but at a great cost.

This is reflected in both 'tackle-live' and 'tackle-replay,' with the F1-score being 0.72 for 'tackle-live' and 0.79 for 'tackle-replay.' Prior to analyzing these results, no score sub 0.80 for both tackle classes had been seen. Model 1, also focusing on 'background' is the only model that has an F1-score below 0.81 for a tackle class, but with better



(a) Confusion matrix.

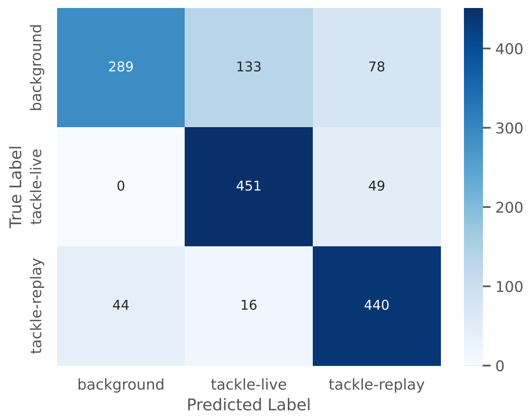
Event	Precision	Recall	F1-score	Support
Background	0.61	0.74	0.67	500
Tackle-live	0.78	0.67	0.72	500
Tackle-replay	0.82	0.76	0.79	500
Macro avg	0.74	0.73	0.73	1500
Weighted avg	0.74	0.73	0.73	1500
Accuracy			**0.73**	1500

(b) Evaluation metrics.

Figure 5.13: Confusion matrix and evaluation metrics for Model 7.

metrics all-round for 'background' and also for 'tackle-live.' Model 7 beats the baseline model only in recall, which might not surprise given the skewed prediction. 127 FPs as 'tackle-live' and 106 as 'tackle-replay' take the overall accuracy and F1-score to an unimpressive 0.73.

5.3.2 Model 8



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.87	0.58	0.69	500
Tackle-live	0.75	0.90	0.82	500
Tackle-replay	0.78	0.88	0.82	500
Macro avg	0.80	0.79	0.78	1500
Weighted avg	0.80	0.79	0.78	1500
Accuracy			**0.79**	1500

(b) Evaluation metrics.

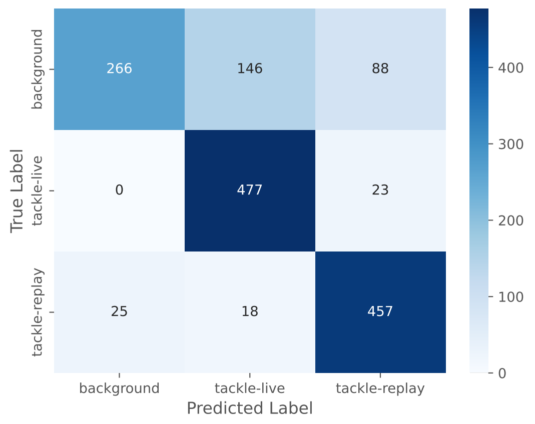
Figure 5.14: Confusion matrix and evaluation metrics for Model 8.

Model 8, which like Models 2 and 5 aimed to effectively classify 'tackle-live,' encountered the persistent issues distinguishing this class from 'background.' Figure 5.14a highlights this challenge, showing that there were 133 FPs as 'background' out of 600 total predictions for 'tackle-live,' which is reflected in a precision score of 0.75. Despite these misclassifications, Model 8 identified 451 TPs, achieving a recall score of 0.90 or 90% accuracy (for 'tackle-live'). This results in an F1-score of 0.82 for its target class, which is higher than the baseline but lower than that of Models 2 and 5.

Due to this misclassification, also the recall for ’background’ is affected. While the model can predict ’background’ with a precision score of 0.87, the poor recall score of 0.58 results in an F1-score of 0.69.

In the case of ’tackle-replay,’ Model 8 successfully identified 440 TPs of 500, achieving a score for recall of 0.88, which matches that of Model 5. However, unlike Model 5, which had a high precision score of 0.85, Model 8’s broader tendency to misclassify both ’tackle-live’ and ’background’ instances as ’tackle-replay’ resulted in a lower precision score of 0.75. Consequently, Model 8 recorded the lowest F1-score among the models for ’tackle-replay,’ at 0.82.

5.3.3 Model 9



(a) Confusion matrix.

Event	Precision	Recall	F1-score	Support
Background	0.87	0.58	0.69	500
Tackle-live	0.75	0.90	0.82	500
Tackle-replay	0.78	0.88	0.82	500
Macro avg	0.80	0.79	0.78	1500
Weighted avg	0.80	0.79	0.78	1500
Accuracy			**0.79**	1500

(b) Evaluation metrics.

Figure 5.15: Confusion matrix and evaluation metrics for Model 9.

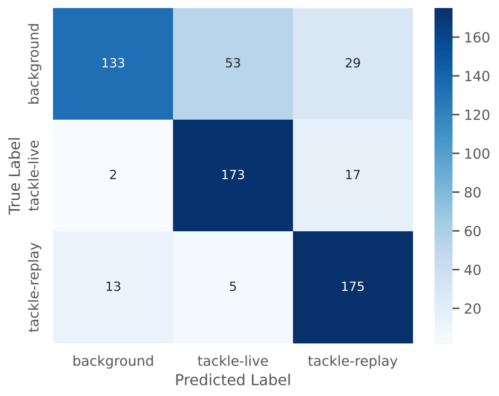
Model 9, focusing on ’tackle-replay,’ demonstrates notable performance metrics in Table 5.15b, achieving a recall of 0.91 and a precision of 0.80, which translates into an F1-score of 0.86. This score matches that of Model 6, surpasses the baseline, and is only slightly lower than Model 3 by 0.01. The slightly reduced precision for Model 9 compared to other models explains why it doesn’t reach the higher F1-score seen in Model 6. The confusion matrix in Figure 5.15a indicates 43 FNs for ’tackle-replay’ Additionally, the model incorrectly predicted 88 FPs as ’background’ and 23 as FPs as ’tackle-live,’ contributing to its lower precision.

In terms of predicting ’background,’ Model 9 achieves a high precision score of 0.91 but struggles with a low recall of 0.53 due to 146 instances being misclassified as ’tackle-live.’ This misclassification rate results in a drained F1-score of 0.67, reflecting the ongoing challenge in distinguishing ’background’ from ’tackle-live.’

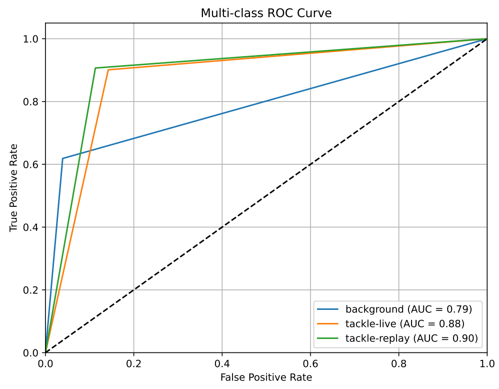
For ’tackle-live,’ Model 9’s precision drops to 0.74 again thought to be due to the visual similarities with ’background.’ However, it excels in recall, capturing 477 TPs of the 500 for a score of 0.95, the highest among the evaluated models. This results in an

F1-score of 0.84, surpassing Model 3 and matching Model 6, underscoring its effectiveness in capturing its target class despite precision challenges.

5.3.4 Meta-learner 7-9



(a) Confusion Matrix.



(b) ROC Curve.

Figure 5.16: Confusion Matrix and ROC Curve for Meta-learner 7-9 using CLS-tokens from reflective padded frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.90	0.62	0.73	215
Tackle-live	0.75	0.90	0.82	192
Tackle-replay	0.79	0.91	0.85	193
Macro avg	0.81	0.81	0.80	600
Weighted avg	0.82	0.80	0.80	600
Accuracy			**0.80**	600

Table 5.4: Evaluation metrics for Meta-learner 7-9.

Finally, the third Meta-learner uses Models 7-9 to try and combine their strengths. In Figure 5.3.4 and Table 5.4, we can see the confusion matrix and classification report displaying the performance of the model.

Examining the confusion matrix reveals that the model achieves 133 TPs for ’background’ with only 15 FPs, as shown in Table 5.4. Although its precision of 0.90 surpasses Meta-learners 1-3 and the baseline, it slightly trails behind Meta-learners 4-6 who score 0.91. Meta-learners 7-9, particularly Models 8 and 9 with scores below 0.60, struggle with recall, recording the lowest at 0.62. This model exceeds the baseline in F1-score for ’background’ but falls behind the other meta-learners.

For ’tackle-live,’ a high recall of 0.90 and a lower precision of 0.75 result in an F1-score of 0.82. The precision drops by 0.25 due to 53 FPs as ’background’ and 5 as ’tackle-replay’, marking the lowest precision among all meta-learners and the baseline model. However, it still outperforms the baseline in F1-score thanks to high recall, equals Meta-learners 4-6, and falls short of Meta-learners 1-3.

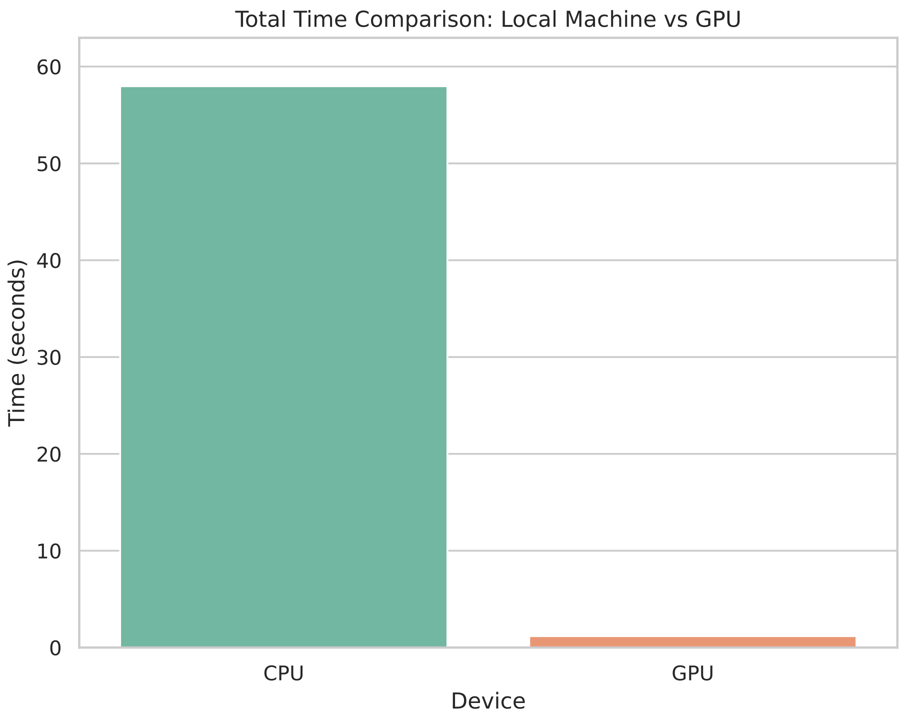


Figure 5.17: Illustration of a full inference of 1 second using the meta-learners on a CPU vs. GPU.

Recording the highest precision among meta-learners at 0.79, with only 46 FPs in 221 'tackle-replay' predictions, this model is surpassed only by the baseline's 0.82. However, its high recall of 0.91, correctly identifying 175 TPs out of 193 possible, secures an F1-score of 0.85, equaling the baseline and other meta-learners.

5.4 (End-to-End) System Performance

Given that all experiments ran on a NVIDIA DGX-1 with several GPUs and the fact that these were resources available under Slurm Workload Manager which is a workload manager that offers job scheduling for Linux systems, by allocating user resources from the cluster, inference time might vary as it would be dependent on how much resources that are available. Therefore, the decision was made to utilize a local machine instead.

My local machine is a MacBook Pro 2019 using a 2.6 GHz 6-core Intel Core i7 CPU with 16 GB RAM. Unfortunately, I do not have a GPU on my local machine, which does not enable the acceleration of model computations that GPUs provide. However, for the purpose of measuring inference time, the decision to run inference on my MacBook Pro was made. This approach allows evaluation of how the models would perform under conditions where all resources are dedicated, unfortunately, on a CPU rather than the idealized conditions of a high-performance computing environment offered by a GPU.

For the inference timing, the same test set previously used to evaluate the base models in this chapter was employed. Inference using the meta-learners on this test set took approximately 1.034 seconds, over a total of 1,500 samples or frames. This timing refers specifically to the processing of the CLS-tokens extracted by DINOv2. It is important to note that if the model were applied to a completely new video clip, additional time would be required to first extract these CLS-tokens, adding an extra step to the overall processing time.

To maximize the efficiency of the feature extraction process, NVIDIA DGX-1 equipped with Tesla V100 GPUs were used. This high-performance computing environment accelerated the data processing by offering 512 Gigabytes of memory, 40,960 NVIDIA CUDA cores, and 5,120 Tensor Cores.

For research purposes, it is also essential to evaluate the performance of CLS-token extraction using DINOv2 on my local machine, along with the inference tests. The extraction process for the same 1,500 frames, corresponding to the number of frames in the test set, took a total of 3,477.01 seconds, which translates to approximately 57 minutes and 57 seconds. This equates to an extraction time of about 2.318 seconds per frame.

To further evaluate system performance, we measured inference time on the NVIDIA GPUs for comparison, despite the potential risk of resource allocation.

For feature extraction using the GPUs, processing 1,500 frames took a total of 70.21 seconds or 46.8 milliseconds per frame. This is approximately 50 times faster than my local machine. The videos used had a frame rate of 25 fps, meaning 25 frames are processed per second. Timing inference on the meta-learners on the GPUs showed that processing a single CLS-token took 0.28 milliseconds, which translates to 7 milliseconds for 25 frames. Adding this to the DINOv2 processing time for 25 frames, which is 1.17 seconds, results in a total of 1.177 seconds to fully process one second of video.

On my local machine, processing a frame through DINOv2 took 2.318 seconds. Multiplying this by the fps gives 57.95 seconds. Adding inference time for a CLS-token, which takes 0.00069 seconds multiplied by 25 fps, equals 17.25 milliseconds. This means that the same process that takes 1.177 seconds on the GPUs requires 57.97 seconds on my local machine, reflected in Figure 5.17.

The time required for model optimization was also assessed, as changes in input data might necessitate model adjustments. Running 150 trials for hyperparameter optimization took 181.69 seconds on the GPUs, approximately 3 minutes and 1.8 seconds. On a local machine, the same process took 269.60 seconds, or about 4 minutes and 29.6 seconds.

Understanding these measurements is essential for evaluating the model's efficiency and responsiveness, especially when quick adjustments are needed to accommodate new data or operational requirements. Efficient hyperparameter optimization allows the model to adapt effectively without much downtime.

To store the meta-learners, space not only for the meta-learner itself but also for the three base-learners is needed. The meta-learner required 8 kB, and each base-learner took up 283 kB, totaling 857 kB, or 0.857 MB.

5.5 Chapter Summary

This chapter presents the results of experiments that utilized spatial features extracted by DINOv2, represented by CLS-tokens, using the TACDEC dataset introduced in Chapter 3. By developing meta-learners that utilized 'experts,' each specialized in a particular class, an improvement in the F1-scores for all meta-learners compared to the baseline model was observed.

Exploring whether altering the way frames are fed to an image processor would yield improvements were also made. Typically, frames are centre-cropped to a (224, 224) input size, which can omit information when the original aspect ratio is not 1:1. By modifying the frames before cropping, some improvements among base learners across various classes and metrics were observed, while a decrease in others occurred.

To broaden the evaluation and shift perspectives, we will move beyond analyzing features within frames and investigate temporal features between frames in Chapter 6.

Chapter 6

Experiments and Results: Temporal Approach

In Chapter 5, the results obtained from training different meta-learners by exploiting the predictive capabilities of base models trained to target specific classes were presented. We also saw how different modifications to the frames prior to cropping by the image processor could change performance. In this chapter, the results of the temporal approach presented in Section 4.5 under Chapter 4 using two different techniques determining how the model sees patterns in the data will be showcased. These results will be compared to the results of a baseline, the best model from Chapter 5 and each other. As in Chapter 5, F1-score will be the focus metric.

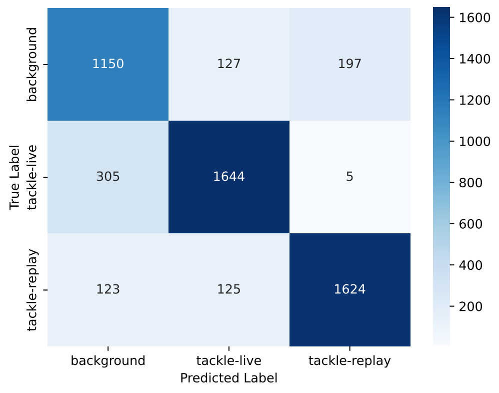
6.1 Experiment Overview

Section	Experiment	Description	Aspect Ratio	Full Image
6.2	**Simple CNN**	Baseline trained on CLS-tokens from centre-cropped raw frames.	4:3	✗
6.3	**Shifting Window**			
6.3.1	Centre Cropping	Results from experiments where raw frames were centre-cropped, possibly losing important information.	4:3	✗
6.3.1.1	Window Size 25	Results from using shifting window of size 25.	4:3	✗
6.3.1.2	Window Size 50	Results from using a shifting window of size 50.	4:3	✗
6.3.1.3	Window Size 75	Results from using a shifting window of size 75.	4:3	✗
6.3.2	Stretched Cropping	Results from experiments where frames were stretched before centre-cropping to preserve information.	1:1	✓
6.3.2.1	Window Size 25	Results from using a shifting window of size 25.	1:1	✓
6.3.2.2	Window Size 50	Results from using a shifting window of size 50.	1:1	✓
6.3.2.3	Window Size 75	Results from using a shifting window of size 75.	1:1	✓
6.3.3	Padded Cropping	Results from experiments where frames were reflective-padded before centre-cropping to preserve information.	1:1	✓
6.3.3.1	Window Size 25	Results from using a shifting window of size 25.	1:1	✓
6.3.3.2	Window Size 50	Results from using a shifting window of size 50.	1:1	✓
6.3.3.3	Window Size 75	Results from using a shifting window of size 75.	1:1	✓
6.4	**Sliding Window**			
6.4.1	Centre Cropping	Results from experiments where raw frames were centre-cropped, possibly losing important information.	4:3	✗
6.4.1.1	Window Size 25	Results from using a shifting window of size 25.	4:3	✗
6.4.1.2	Window Size 50	Results from using a shifting window of size 50.	4:3	✗
6.4.1.3	Window Size 75	Results from using a shifting window of size 75.	4:3	✗
6.4.2	Stretched Cropped	Results from experiments where frames were stretched before centre-cropping to preserve information.	1:1	✓
6.4.2.1	Window Size 25	Results from using a sliding window of size 25.	1:1	✓
6.4.2.2	Window Size 50	Results from using a sliding window of size 50.	1:1	✓
6.4.2.3	Window Size 75	Results from using a sliding window of size 75.	4:3	✓
6.4.3	Padded Cropping	Results from experiments where frames were reflective-padded before centre-cropping to preserve information.	1:1	✓
6.4.3.1	Window Size 25	Results from using a sliding window of size 25.	1:1	✓
6.4.3.2	Window Size 50	Results from using a sliding window of size 50.	1:1	✓
6.4.3.3	Window Size 75	Results from using a sliding window of size 75.	4:3	✓

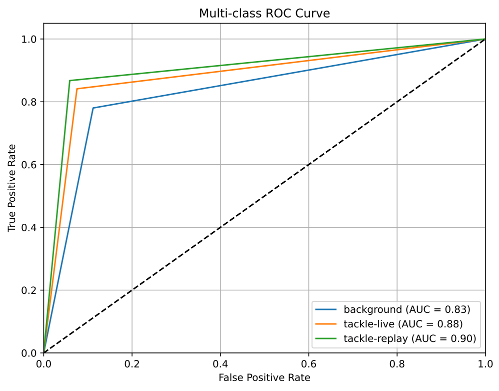
Table 6.1: An overview of the different experiments in this chapter that were conducted using temporal features.

Table 6.1 provides an overview of the sections and results following our experiments utilizing TempTAC with the blue rows indicating a shift in how we feed the TempTAC data.

6.2 Simple CNN



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.1: Confusion Matrix and ROC Curve for the baseline CNN using CLS-tokens from centre-cropped frames with a shifting window of 50 frames.

Event	Precision	Recall	F1-score	Support
Background	0.73	0.78	0.75	1474
Tackle-live	0.87	0.84	0.85	1954
Tackle-replay	0.89	0.87	0.88	1872
Macro avg	0.83	0.83	0.83	5300
Weighted avg	0.84	0.83	0.83	5300
Accuracy			**0.83**	5300

Table 6.2: Evaluation metrics for the simple CNN with a shifting window of 25 frames on centre cropped CLS-tokens.

To form a baseline, similar to the one developed for the spatial approach in Chapter 5, a simple CNN was developed. The performance of this can be found in Figure 6.1 and Table 6.2. It is worth noting that this baseline beats the Meta-learner 1-3, proving best from Chapter 5, on F1-scores for both tackles.

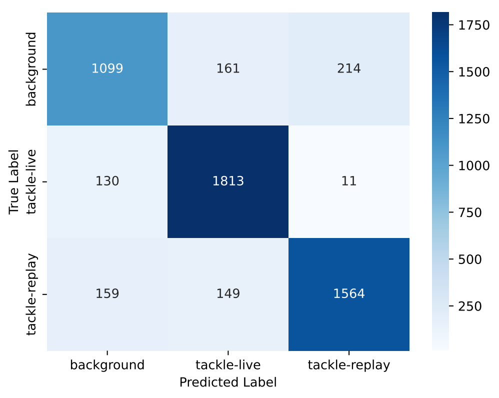
6.3 Shifting Window

This section presents the results using TempTAC for the three types of alterations conducted before feeding TempTAC via DINOv2's image processor, using shifting windows as described in Section 4.3. Experiments were conducted with three different window sizes for each alteration.

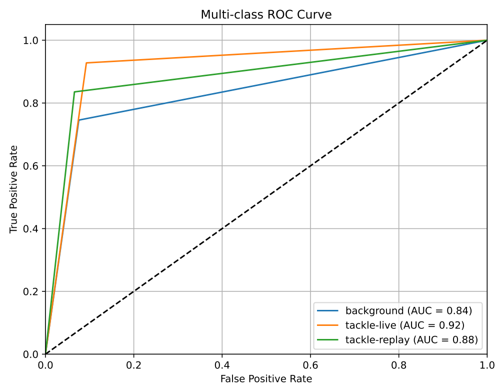
6.3.1 Centre Cropping

This section, like Section 5.1, contains results from experiments conducted using CLS-tokens extracted from the raw frames using a centre-cropping technique by DINOv2's image processor.

6.3.1.1 Window Size 25



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.2: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a shifting window of 25 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.79	0.75	0.77	1474
Tackle-live	0.85	0.93	0.89	1954
Tackle-replay	0.87	0.84	0.85	1872
Macro avg	0.84	0.84	0.84	5300
Weighted avg	0.84	0.84	0.84	5300
Accuracy			**0.84**	5300

Table 6.3: Evaluation metrics for TempTAC with a shifting window of 25 frames on centre cropped CLS-tokens.

The first experiment conducted involved using TempTAC on CLS-tokens derived from the centre-crop, with a window size of 25. As shown in Table 6.3, the precision and recall scores are well balanced at 0.79 and 0.75, respectively, resulting in an F1-score of 0.77. In Chapter 5, 'background' was identified as the hardest class to distinguish. The balanced scores indicate that temporal features may help the model distinguish this class more effectively than relying solely on spatial features. This is also evident in the confusion matrix in Figure 6.2a and the ROC curve (blue) in Figure 6.2b, where the AUC score for 'background' is 0.84, outperforming Meta-learner 1-3 by 0.02 and the simple CNN by 0.01 in both AUC and F1-scores.

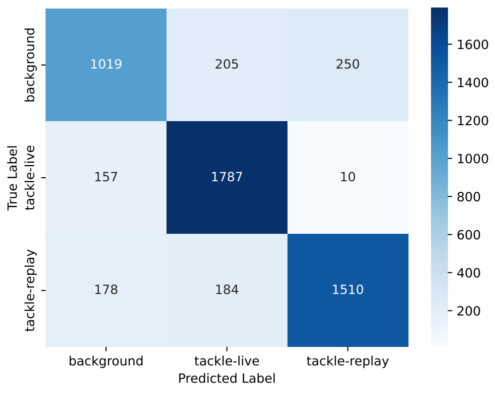
The highest recorded F1-score for 'tackle-live' is 0.89, with a precision of 0.85 and a recall of 0.93. Here, the model surpasses Meta-learner 1-3 in precision and recall. Although TempTAC's precision score is 0.02 lower than that of the simple CNN, it has a 0.09 higher recall score and a 0.04 higher AUC score for 'tackle-live,' achieving a score of 0.92, the highest recorded for this class.

For 'tackle-replay,' the F1-score of 0.85 is slightly lower than the baseline score of

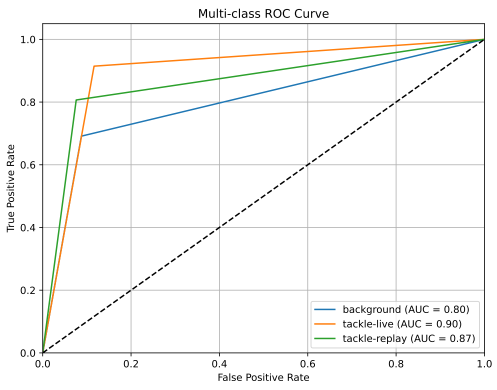
0.88. Despite achieving a precision score of 0.87 and a recall score of 0.84, TempTAC only matches Meta-learner 1-3 in F1-score. However, TempTAC shows a much better balance between precision and recall than Meta-learner 1-3, which has scores of 0.79 and 0.94, respectively. We also observe that only 11 FPs were misclassified as ’tackle-live.’ While the simple CNN misclassified just 5 ’tackle-live’ instances as ’tackle-replay,’ both models represent improvements compared to the results seen in the spatial experiments.

With balanced precision and recall across all three classes, TempTAC achieves the highest overall accuracy and weighted averages of any model so far. In the following subsections, this model will be referred to as Centre-W25.

6.3.1.2 Window Size 50



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.3: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a shifting window of 50 frames.

Event	Precision	Recall	F1-score	Support
Background	0.75	0.69	0.72	1474
Tackle-live	0.82	0.91	0.87	1954
Tackle-replay	0.85	0.81	0.83	1872
Macro avg	0.81	0.80	0.81	5300
Weighted avg	0.81	0.81	0.81	5300
Accuracy			**0.81**	5300

Table 6.4: Evaluation metrics for TempTAC with a shifting window of 50 frames on centre cropped CLS-tokens.

Using a window size of 50, this TempTAC model processes longer sequences than the previous one, covering two seconds of frames instead of one. As shown in Table 6.4, the weighted averages and overall accuracy are 0.81, lower than that of Centre-W25. This trend is consistent across the ROC curves, where Centre-W25 outperforms every category.

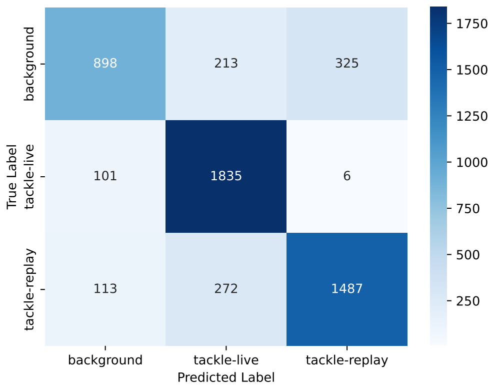
For the ’background’ class, this model achieves a recall score of 0.69 and a precision score of 0.75. Both scores are lower than those of Meta-learner 1-3 and Centre-W25.

Although it narrowly surpasses the simple CNN in precision (0.75 vs. 0.73), it lags in recall and thus has a lower F1-score.

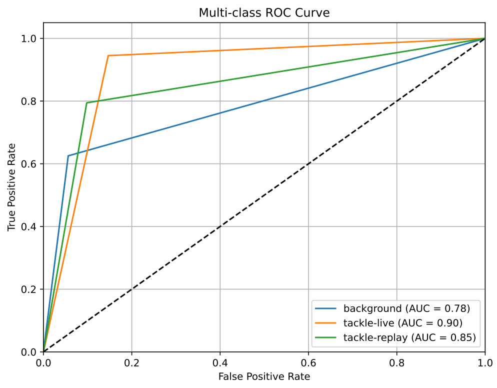
For 'tackle-live' and 'tackle-replay,' performance decreases compared to Centre-W25. While the model still recalls 91% with a precision of 82%, its F1-score of 0.87 for 'tackle-live' outperforms Meta-learner 1-3's score of 0.83 and beats the simple CNN in F1-score due to the high recall. The confusion matrix in Figure 6.3a shows that 1,787 TPs of the total 1,954 'tackle-live' instances, reflecting the high recall score, although the precision took a hit from the FPs.

For 'tackle-replay,' the model's F1-score of 0.83 is lower than those of Meta-learner 1-3, the simple CNN, and Centre-W25, reflected in the lower AUC score in Figure 6.3b. Combined with the performance for the other two classes, this results in the lowest averages and accuracy so far in this chapter. Similar to Centre-W25, this model will be referred to as Centre-W50.

6.3.1.3 Window Size 75



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.4: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a shifting window of 75 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.81	0.63	0.70	1436
Tackle-live	0.79	0.94	0.86	1942
Tackle-replay	0.82	0.79	0.81	1872
Macro avg	0.81	0.79	0.79	5250
Weighted avg	0.81	0.80	0.80	5250
Accuracy			**0.80**	5250

Table 6.5: Evaluation metrics for TempTAC with a shifting window of 75 frames on centre cropped CLS-tokens.

The results obtained from feeding the model sequences of three seconds (or CLS-tokens for 75 frames) can be seen in Figure 6.3.1.3 and Table 6.5. Similar to Centre-W50, increasing the window size resulted in a decrease in averages and overall accuracy.

The confusion matrix in Figure 6.4a reveals a high number of FPs for 'background,' reflected in the low recall score of 0.63. Although this model achieved the highest precision score (0.81) compared to Centre-W25 and Centre-W50, the low recall led to an F1-score of just 0.70. This score is poorer than of Centre-W25, Centre-W50, the simple CNN, and Meta-learner 1-3. Figure 6.4b illustrates a low AUC score for 'background,' making this the first model to fall below 0.80 in AUC.

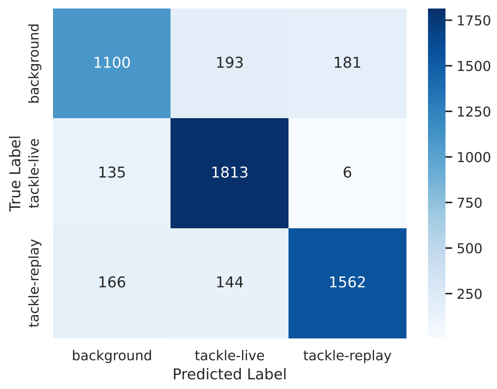
With only 107 FNs out of 1,942, the model achieves an impressive recall score of 0.94, which skews the F1-score to 0.86 despite a precision score of only 0.79 for 'tackle-live.' This F1-score is higher than Meta-learner 1-3 (0.83) and the simple CNN (0.85).

For 'tackle-replay,' precision and recall scores of 0.82 and 0.79, respectively, result in an F1-score of 0.81. This puts the model behind all the models compared for 'tackle-replay,' as well as all other meta-learners and the baseline model from the previous chapter. This performance is reflected in the macro and weighted averages, as well as the overall accuracy.

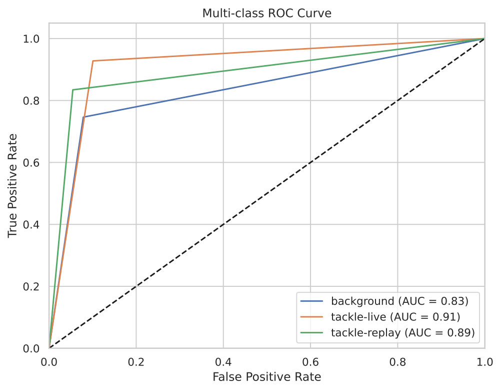
6.3.2 Stretched Cropping

This section will present results for the same window sizes using a shifting window but on CLS-tokens that were stretched prior to DINOv2's image processing.

6.3.2.1 Window Size 25



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.5: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a shifting window of 25 frames.

In Table 6.6, we can see that the model achieves F1-scores of 0.77, 0.88, and 0.86 for 'background,' 'tackle-live,' and 'tackle-replay,' respectively. These results align with the AUC scores shown in Figure 6.5b, which are 0.83, 0.91, and 0.89 in the same order.

A closer look reveals that the model's performance for 'background' is identical to that of Centre-W25, outperforming Meta-learner 1-3, the baseline model, and, thereby, both Centre-W50 and Centre-W75.

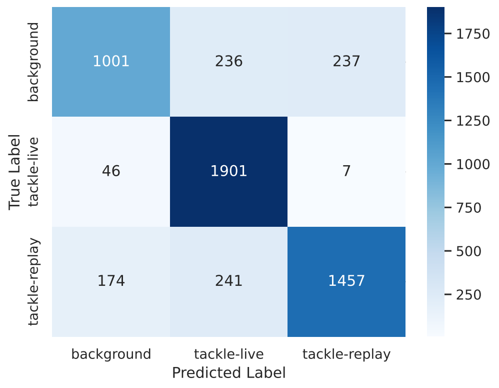
Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.79	0.75	0.77	1474
Tackle-live	0.84	0.93	0.88	1954
Tackle-replay	0.89	0.83	0.86	1872
Macro avg	0.84	0.84	0.84	5300
Weighted avg	0.84	0.84	0.84	5300
Accuracy			**0.84**	5300

Table 6.6: Evaluation metrics for TempTAC with a shifting window of 25 frames on stretched cropped CLS-tokens.

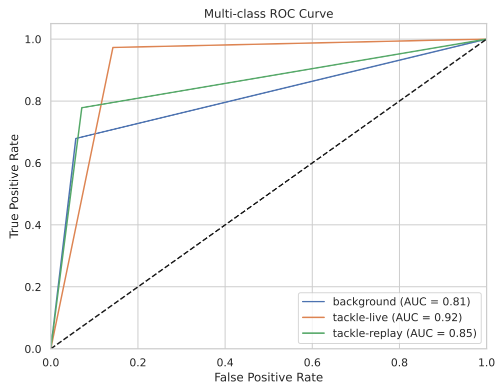
For 'tackle-live,' the model achieves an impressive recall score of 0.93, with only 141 FNs out of 1,954 samples. The precision score of 0.84 explains the high F1-score of 0.88, suggesting that the model reliably detects live tackles.

In the 'tackle-replay' class, the model records a precision score of 0.89 (higher than Centre-W25 and equal to the simple CNN) and a recall score of 0.83, which falls short of both Centre-W25 and the simple CNN. As a result, the F1-score for 'tackle-replay' is higher than Centre-W25 but lower than the simple CNN due to the recall score. The same procedure for naming is followed, and this model will be referred to as Stretched-W25.

6.3.2.2 Window Size 50



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.6: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a shifting window of 50 frames.

When increasing the shifting window size, the averages and overall accuracy decrease, as seen in Table 6.7, similar to what occurred with the centre-crop.

The standout result is the high recall score for 'tackle-live,' which reaches 0.97, surpassing all other models. The confusion matrix in Figure 6.6a confirms this, with only 53 FNs out of 1,954 instances. This indicates the model correctly identifies a record high 97% of all 'tackle-live' instances, doing so with 80% precision and 477 FPs. Due to

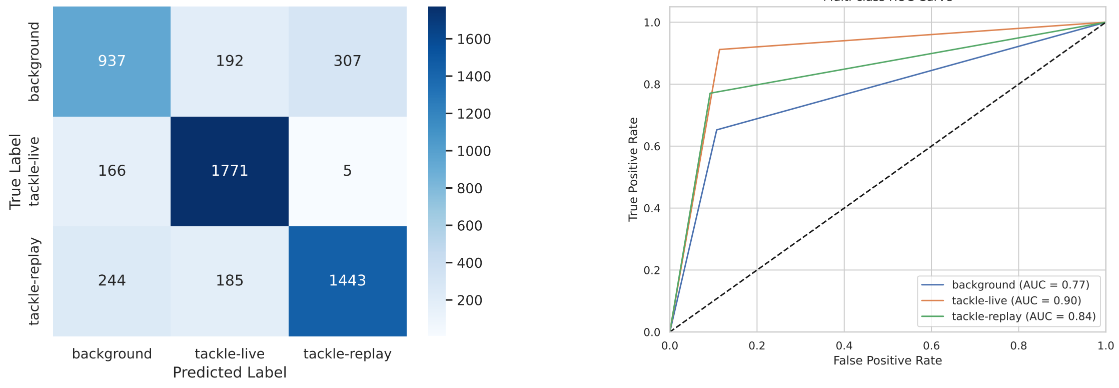
Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.82	0.68	0.74	1474
Tackle-live	0.80	0.97	0.88	1954
Tackle-replay	0.86	0.78	0.82	1872
Macro avg	0.83	0.81	0.81	5300
Weighted avg	0.83	0.82	0.82	5300
Accuracy			**0.82**	5300

Table 6.7: Evaluation metrics for TempTAC with a shifting window of 50 frames on stretched cropped CLS-tokens.

slightly lower precision score, it ranks second in F1-score for this class (0.88), just 0.01 behind Centre-W25, with an identical AUC score as shown in Figure 6.6b.

While the model captures ’tackle-live’ effectively, a recall score of 0.78 for ’tackle-replay’ suggests it struggles more with replay tackles. Despite this, there are only 244 FPs, reflected in the high precision score of 0.86, which beats Meta-learner 1-3 and narrowly surpasses Centre-W50 but trails behind the simple CNN. The model, now referred to as Stretched-W50, ultimately falls behind all these models in F1-score.

6.3.2.3 Window Size 75



(a) Confusion Matrix.

(b) ROC Curve.

Figure 6.7: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a shifting window of 75 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.70	0.65	0.67	1436
Tackle-live	0.82	0.91	0.87	1942
Tackle-replay	0.82	0.77	0.80	1872
Macro avg	0.78	0.78	0.78	5250
Weighted avg	0.79	0.79	0.79	5250
Accuracy			**0.79**	5250

Table 6.8: Evaluation metrics for TempTAC with a shifting window of 75 frames on stretched cropped CLS-tokens.

In Table 6.8, we observe another decrease in performance with the largest shifting window size. The macro averages score 0.78, and the weighted averages score 0.79.

Analyzing the class-wise performance, the 'background' class has an F1-score of only 0.67, the lowest score recorded so far. This is due to 499 FNs and 410 FPs, as detailed in the confusion matrix in Figure 6.7a. This class's performance lags behind all other models evaluated in this chapter, including Meta-learner 1-3, which is reflected in the 'background' ROC curve (blue) in Figure 6.7b.

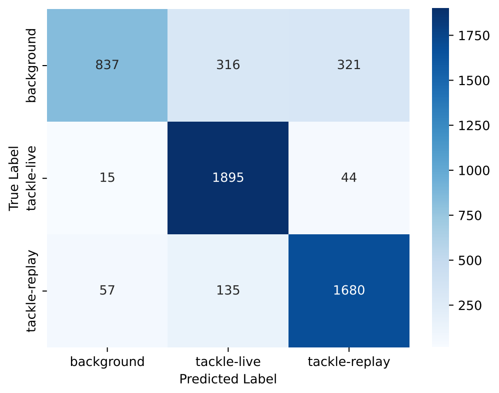
The model accurately recalls 1,771 TPs for 'tackle-live,' missing only 179 (FNs) to achieve a recall score of 0.91. Despite a slightly lower precision score of 0.82, the high recall yields an F1-score of 0.87, higher than the simple CNN but lower than the other TempTAC models. The model decently recalls live tackles but misclassifies 'background' and 'tackle-replay' as FPs.

For 'tackle-replay,' recall and precision scores are 0.82 and 0.77, respectively. As with 'background,' this class scores the lowest in both metrics and thus has a lower F1-score. Despite using the same window size as Centre-W75, this model shows no improvement but a decline in performance.

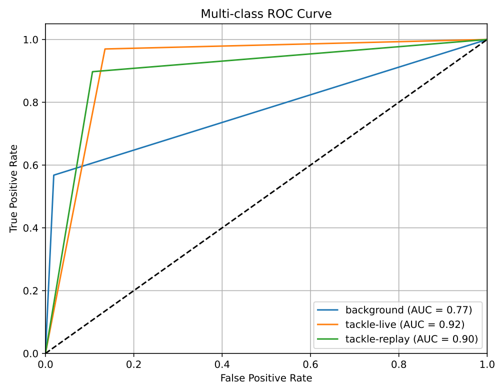
6.3.3 Padded Cropping

This section presents results for the same window sizes using a shifting window on CLS-tokens derived from frames that were reflectively padded prior to DINOv2's image processing.

6.3.3.1 Window Size 25



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.8: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a shifting window of 25 frames.

Using the window size of 25, which proved effective in both Section 6.3.1 and Section 6.3.2, results in the lowest average F1-scores compared to Centre-W25, Stretched-W25, and the simple CNN. The macro averages are also lower, though the weighted averages are on par with Meta-learner 1-3. This inconsistency suggests that the model performs less consistently across classes than its competitors.

Event	Precision	Recall	F1-score	Support
Background	0.92	0.57	0.70	1474
Tackle-live	0.81	0.97	0.88	1954
Tackle-replay	0.82	0.90	0.86	1872
Macro avg	0.85	0.81	0.81	5300
Weighted avg	0.84	0.83	0.82	5300
Accuracy			**0.83**	5300

Table 6.9: Evaluation metrics for TempTAC with a shifting window of 25 frames on reflected cropped CLS-tokens.

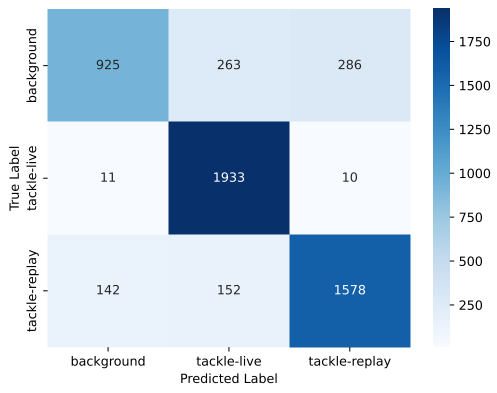
Despite achieving a precision score of 0.92 for ’background,’ as shown in Table 6.9, with only 72 FPs, the 637 FNs result in a recall score of 0.57. The ROC curve (blue) in Figure 6.8b also reflects this with a low TP rate despite the low FP rate.

For ’tackle-live,’ the model performs exceptionally well, achieving an F1-score of 0.88. ’Tackle-live’ remains a class where models consistently perform well despite declines in the other two classes. Like Stretched-W50, this model captures 97% of ’tackle-live’ instances, with only 59 FNs, as shown in Figure 6.17a. The curve (orange) in Figure 6.8b shows a TP rate just below 1.0. However, due to the lower precision resulting from 451 FPs, the F1-score is 0.88—better than the simple CNN and Meta-learner 1-3, equal to Stretched-W25, and 0.01 lower than Centre-W25.

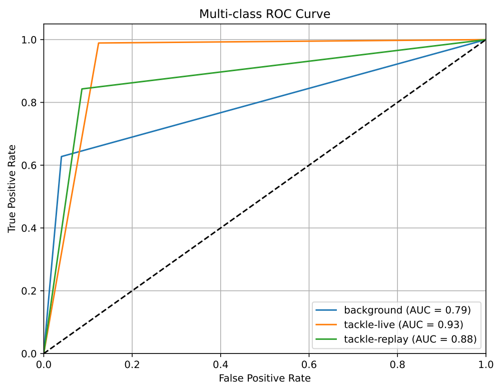
With 192 FNs, the model achieves a well-deserved recall score of 0.90, the highest of any model in this chapter, surpassing the previous high held by the simple CNN. However, the precision score of 0.82 results in an F1-score of 0.86.

Overall, with an accuracy of 0.83, this model’s performance was overshadowed by its inability to balance the metrics as effectively as Centre-W25 and Stretched-W25, primarily due to over 600 ’background’ instances being misclassified as ’tackle-live’ or ’tackle-replay. This model will be referred to as Padded-W25.

6.3.3.2 Window Size 50



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.9: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a shifting window of 50 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.86	0.63	0.72	1474
Tackle-live	0.82	0.99	0.90	1954
Tackle-replay	0.84	0.84	0.84	1872
Macro avg	0.84	0.82	0.82	5300
Weighted avg	0.84	0.84	0.83	5300
Accuracy			**0.84**	5300

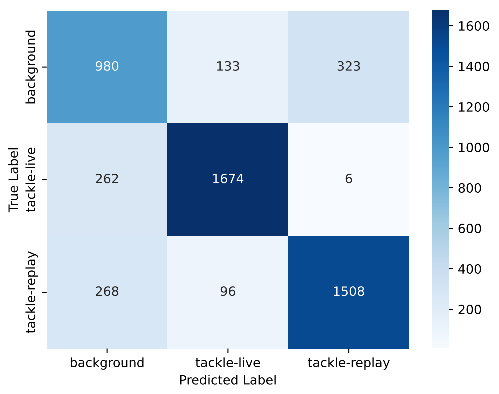
Table 6.10: Evaluation metrics for TempTAC with a shifting window of 50 frames on reflected cropped CLS-tokens.

Doubling the window size once again, a decline in performance was anticipated, as this has been the pattern so far. However, Figure 6.9b shows that the ROC curve for 'tackle-live' (orange) almost aligns with a TP rate of 1.0. Table 6.10 reveals that the model recalls 99% of all 'tackle-live' instances, with only 21 FNs, as indicated in Figure 6.9a. This score surpasses Padded-W25 and all other models in the same metric. Despite the confusion matrix showing 415 FPs, the precision score of 0.82 leads to an F1-score of 0.90, the highest recorded so far.

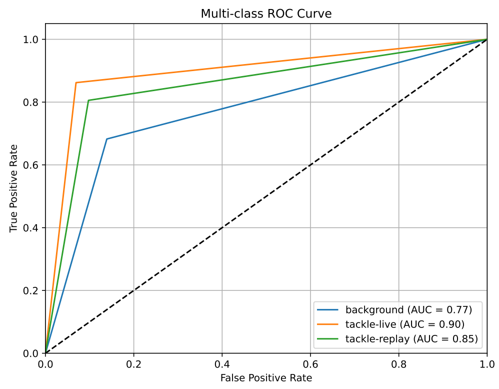
For 'background,' the model has a recall score of 0.63, slightly better than Padded-W25 but still worse than other competitors. Figure 6.9b shows that the AUC score for 'background' is better than Padded-W25, the model with a window size of 25. With 153 FPs, a precision score of 0.86 helps lift the F1-score to 0.72.

For 'tackle-replay,' the model demonstrates balanced precision, recall, and F1 scores, each at 0.84, suggesting reliable detection of replay tackles. Additionally, the averages and overall accuracy of 0.84, as presented in the table, reflect an improvement over Padded-W25, which was surprising given previous results. The model outperforms Centre-W50, Stretched-W50, the simple CNN, and Meta-learner 1-3 in overall accuracy and most averages.

6.3.3.3 Window Size 75



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.10: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a shifting window of 75 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.65	0.68	0.67	1436
Tackle-live	0.88	0.86	0.87	1942
Tackle-replay	0.82	0.81	0.81	1872
Macro avg	0.78	0.78	0.78	5250
Weighted avg	0.80	0.79	0.79	5250
Accuracy			**0.79**	5250

Table 6.11: Evaluation metrics for TempTAC with a shifting window of 75 frames on reflected cropped CLS-tokens.

In the final evaluation for shifting windows, a TempTAC model trained with a shifting window of size 75 is assessed using the same CLS-tokens as for the two results above. Figure 6.10b reveals that the ROC curves for all classes follow a similar trajectory, indicating balanced precision and recall across the three classes. Table 6.11 confirms this, with no more than a 0.03 difference between class-wise precision and recall scores.

Despite this balance, the model scores similarly to Stretched-W75 for the 'background' class, achieving precision and recall scores of 0.65 and 0.68, respectively. An F1-score of 0.67 reflects the model's difficulty in distinguishing 'background' from the tackle classes.

Detecting live tackles poses less of a challenge, with a precision score of 0.88 based on 1,674 TPs and only 229 FPs. However, the model struggles with 'background,' resulting in 262 FNs misclassified as 'background.' Despite this, it correctly identifies 86% of all 'tackle-live' instances, yielding an F1-score of 0.87, which is comparable to Stretched-W75 and outperforms Centre-W75, the simple CNN, and Meta-learner 1-3.

For 'tackle-replay,' the model performs better than Stretched-W75, achieving a precision of 0.82 and a recall of 0.81, but remains below average for this class with an F1-score of 0.81.

Poor performance in the 'background' and 'tackle-replay' classes lowers all averages except one (precision's weighted average of 0.80), resulting in an overall accuracy of 0.79.

6.4 Sliding Window

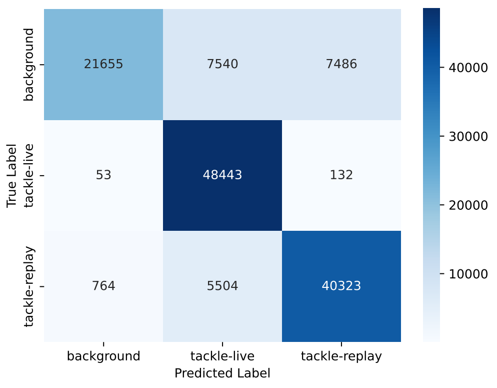
This section presents the results of using TempTAC with the same hyperparameters used in the experiments in Section 6.3 for the three types of frame alterations performed before passing them through DINOv2's image processor. The experiments used sliding windows as described in Section 4.3.

It is worth noting that the total support, reflected in both the confusion matrices and metric tables, will be higher in this analysis. This is because frames are seen multiple times as the window slides over the sequence, depending on the window size.

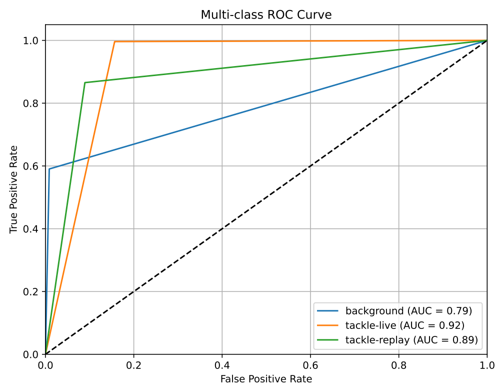
6.4.1 Centre Cropping

This section, like Section 6.3.1, contains results from experiments conducted using CLS-tokens extracted from the raw frames using a centre-cropping technique by DINOv2's image processor.

6.4.1.1 Window Size 25



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.11: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a sliding window of 25 frames.

Event	Precision	Recall	F1-score	Support
Background	0.96	0.59	0.73	36681
Tackle-live	0.79	0.99	0.88	48628
Tackle-replay	0.84	0.87	0.85	46591
Macro avg	0.86	0.82	0.82	131900
Weighted avg	0.86	0.84	0.83	131900
Accuracy			**0.84**	131900

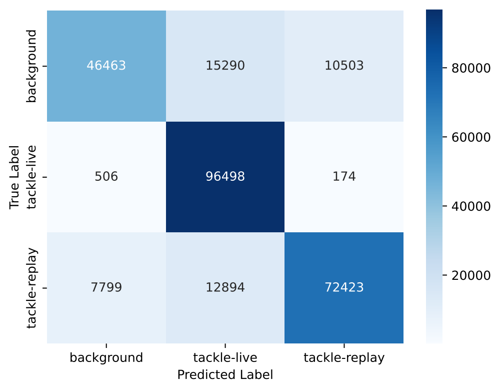
Table 6.12: Evaluation metrics for TempTAC with a sliding window of 25 frames on centre cropped CLS-tokens.

Starting with raw frames using a sliding window of size 25, Table 6.12 shows that the model achieves a precision score of 0.96 for 'background,' meaning it correctly predicts 'background' 96% of the time, which is a record high. However, this is overshadowed by a recall score of 0.59, reducing the F1-score to 0.73. Figure 6.11b also illustrates this with both a low FP rate and a low TP rate.

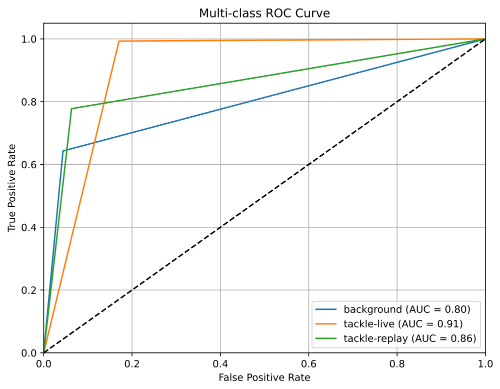
For 'tackle-live,' the precision and recall scores are 0.79 and 0.99, respectively. According to Figure 6.11a, there are only 185 FNs, while 48,443 are correctly classified. This results in an F1-score of 0.88, which surpasses the simple CNN. Despite almost perfect recall, Centre-W25's higher precision ultimately outperforms this model for 'tackle-live,' with an F1-score of 0.89. Ideally, the ROC curve for 'tackle-live' (orange) should shift left to reduce FPs.

’Tackle-replay’ shows balanced performance with precision and recall scores of 0.84 and 0.87, respectively, suggesting that the model is most reliable with this class. In contrast, ’tackle-live,’ despite achieving 99% recall, has nearly 20% FPs. Therefore, the model’s F1-score for ’tackle-replay’ falls behind Centre-W25, Stretched-W25, Padded-W25, and the simple CNN, which still demonstrate the highest performance. Similar to what was done for the shifting window models, this model will be referred to as Centre-W25*, where the asterisk (*) marks the use of a sliding window.

6.4.1.2 Window Size 50



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.12: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a sliding window of 50 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.85	0.64	0.73	72256
Tackle-live	0.77	0.99	0.87	97178
Tackle-replay	0.87	0.78	0.82	93116
Macro avg	0.83	0.80	0.81	262550
Weighted avg	0.83	0.82	0.81	262550
Accuracy			**0.82**	262550

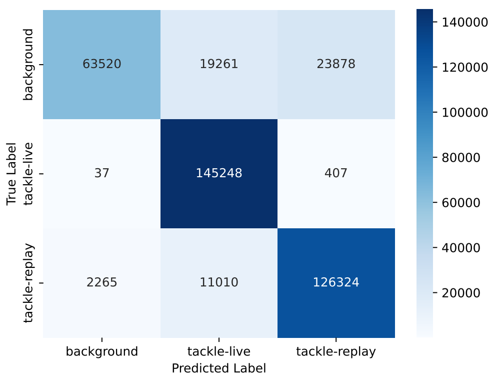
Table 6.13: Evaluation metrics for TempTAC with a sliding window of 50 frames on centre cropped CLS-tokens.

When increasing the window sizes for shifting windows in the previous section, performance generally decreased, except for once. The macro and weighted averages in Table 6.13 show a similar pattern, with all metrics declining.

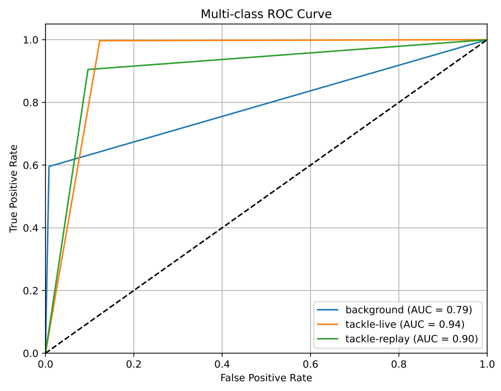
The confusion matrix in Figure 6.12 reveals that ’background’ had many FNs misclassified as ’tackle-live,’ which was a recurring problem in the spatial approach in Chapter 5. In previous results in this chapter, misclassifications were mostly balanced between the two tackle classes, but this trend shifts here. Table 6.13 reflects this, with a low recall score of 0.64 for ’background’ and a precision score of 0.77 for ’tackle-live.’ Additionally, a similar number of FNs for ’tackle-replay’ were misclassified as ’tackle-live,’ impacting the recall. However, 'tackle-live' achieves another high recall score of 0.99, suggesting that the sliding window approach easily recognizes this class also reflected in Figure 6.11b. This results in an F1-score of 0.88, which matches Stretched-W25 and Padded-W25, surpasses the simple CNN but is just 0.01 behind Centre-W25.

With all averages lower than Centre-W25*, it is not surprising that the model now referred to as Centre-W50* has an overall accuracy of 82%, falling behind previously tested models.

6.4.1.3 Window Size 75



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.13: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from centre-cropped frames with a sliding window of 75 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.97	0.60	0.74	106659
Tackle-live	0.83	0.99	0.90	145692
Tackle-replay	0.84	0.90	0.87	139599
Macro avg	0.88	0.83	0.84	391950
Weighted avg	0.87	0.85	0.85	391950
Accuracy			**0.85**	391950

Table 6.14: Evaluation metrics for TempTAC with a sliding window of 75 frames on centre cropped CLS-tokens.

Table 6.14 shows an increase in all averages compared to Centre-W25* and Centre-W50*. The ROC curves in Figure 6.13b reveal a similar curve for 'tackle-live' (orange) as Centre-W50*, but shifted further to the left, indicating higher precision. This is confirmed by the table, which notes a precision score of 0.83 and a recall score of 0.99. This consistency across all models using a sliding window in recalling 99% of live tackles is impressive. Despite sharing the 0.99 recall score with other models, this approach has fewer FNs, even with around 35% more support. The high precision leads to a record F1-score of 0.90, making it the first model to surpass Centre-W25, which scored 0.89.

The trend observed in Centre-W50*, where most 'background' instances were FNs misclassified as ’tackle-live,’ does not continue here. Instead, most ’background’ instances were FNs as ’tackle-replay,’ as seen in Figure 6.13a. Despite this, precision remains high for both ’tackle-live’ (0.83) and ’tackle-replay’ (0.84). The confusion matrix also shows 13,275 FNs for ’tackle-replay,’ which accounts for only about 10% of the 139,599 instances, leading to a recall score of 0.90 and an F1-score of 0.87.

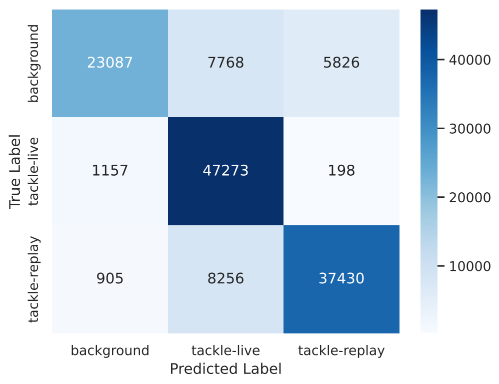
For ’background,’ we see an impressive precision score of 0.97, with only 2,302 FPs and 63,520 TPs. Unfortunately, the misclassifications of ’background’ to the tackle classes result in an F1-score of just 0.74.

Overall, the model’s precision and recall effectiveness, despite issues recalling ’background,’ give it the highest overall accuracy at 85%. For both tackle classes, it also achieves the highest F1-scores, except for ’tackle-replay,’ where only the simple CNN surpasses it. However, this model’s F1-score for ’tackle-live’ is a staggering 0.04 higher.

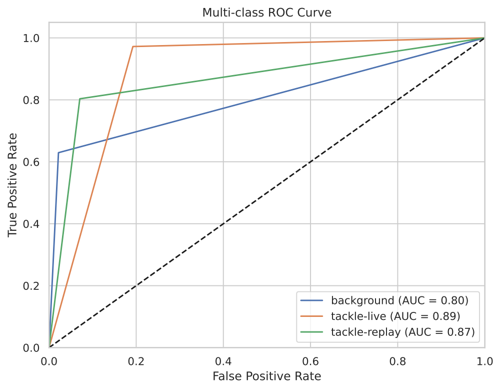
6.4.2 Stretched Cropping

This section, like Section 6.3.2, contains results from experiments conducted using CLS-tokens that were stretched prior to the centre-cropping technique used by DINOv2’s image processor.

6.4.2.1 Window Size 25



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.14: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a sliding window of 25 frames.

The ROC curves in Figure 6.14a show that the FP rate for ’background’ (blue) is very low, while the TP rate for ’tackle-live’ (orange) is high. ’Tackle-replay’ (green) lies between these for both metrics. Table 6.15 confirms this with a high recall score of 0.97 for ’tackle-live,’ though not quite as high as the 0.99 achieved by Centre-W25*, Centre-W50*, and Centre-W75*. With a precision score of 0.75, indicated by the FPs in the confusion matrix in Figure 6.14a, it is outperformed by those three models, as well as

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.92	0.63	0.75	36681
Tackle-live	0.75	0.97	0.84	48628
Tackle-replay	0.86	0.80	0.83	46591
Macro avg	0.84	0.80	0.81	131900
Weighted avg	0.83	0.82	0.81	131900
Accuracy			**0.82**	131900

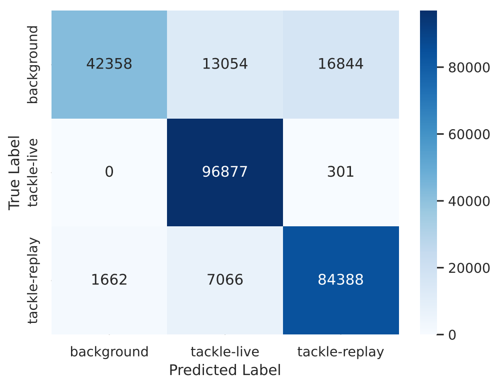
Table 6.15: Evaluation metrics for TempTAC with a sliding window of 25 frames on stretched cropped CLS-tokens.

the simple CNN, Centre-W25, and Meta-learner 1-3. The balance between precision and recall results in an F1-score of 0.84 for 'tackle-live,' which is higher than Meta-learner 1-3 (0.83) but lower than all other models.

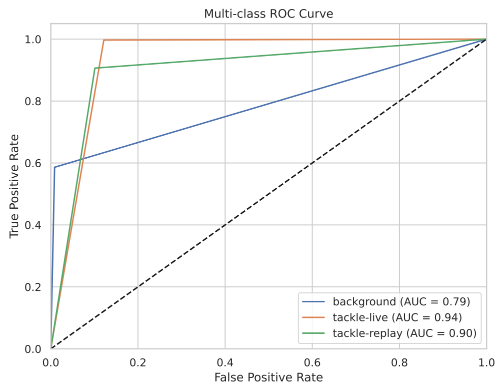
Although Table 6.15 reveals a high precision score of 0.92 for 'background,' the FNs lead to a recall score of 0.63, bringing the F1-score down to 0.75. This is just below Centre-W25 and Stretched-W25, which both had scores of 0.77, matching the simple CNN.

For 'tackle-replay,' the precision score of 0.86 is reflected in the ROC curve (green) and the confusion matrix, which shows only 6,024 FPs against 37,430 TPs. However, the model fails to capture 9,161 FNs, about 20% of the total instances, which results in a recall score of 0.80. The F1-score of 0.83 is unimpressive given other results and places the model near the middle. The averages are lower than those of Centre-W25*, Centre-W25, Stretched-W25, and the simple CNN across all metrics, as well as being outperformed by Meta-learner 1-3 and Reflected-W25 in most. This model will be referred to as Stretched-W25*.

6.4.2.2 Window Size 50



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.15: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a sliding window of 50 frames.

Event	Precision	Recall	F1-score	Support
Background	0.96	0.59	0.73	72256
Tackle-live	0.83	0.99	0.90	97178
Tackle-replay	0.83	0.91	0.87	93116
Macro avg	0.87	0.83	0.83	262550
Weighted avg	0.87	0.85	0.84	262550
Accuracy			**0.85**	262550

Table 6.16: Evaluation metrics for TempTAC with a sliding window of 50 frames on stretched cropped CLS-tokens.

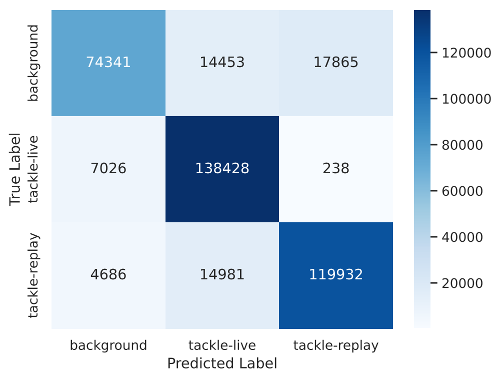
When increasing the window size from 25 to 50 using the sliding window on the raw frames, performance declined. However, Table 6.16 shows improvements for the tackle classes this time. Specifically, the F1-scores are 0.90 and 0.87 for 'tackle-live' and 'tackle-replay,' respectively, matching Centre-W75*, which holds the highest cumulative F1-scores for the tackle classes among all models. The ROC curves in Figure 6.15b show that both 'tackle-live' (orange) and 'tackle-replay' (green) are close to the top-left corner which indicates a perfect classification. These high TP values for the tackle classes are mirrored in Figure 6.15a, where there are only 301 FNs for 'tackle-live' compared to 96,877 TPs, confirmed by a recall score of 0.99 in the table. The confusion matrix reveals almost no FPs for 'background,' resulting in a precision score of 0.96. However, the table also shows a great number of FNs, which lowers the recall score to 0.59. Despite the high precision, this results in an unimpressive F1-score of 0.73. In detecting tackles occurring in replays, the model is nearly as good, achieving the same precision score but a slightly lower recall score of 0.91. This model records the highest recall scores for both 'tackle-live' and 'tackle-replay' among all models evaluated so far. With recall scores of 99% and 91% for the two classes, respectively, this model captures almost every tackle frame but struggles with 'background.' Further, this model will be referred to as Stretched-W50*.

6.4.2.3 Window Size 75

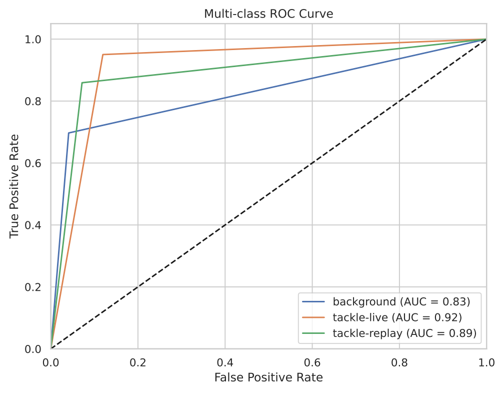
Event	Precision	Recall	F1-score	Support
Background	0.86	0.70	0.77	106659
Tackle-live	0.82	0.95	0.88	145692
Tackle-replay	0.87	0.86	0.86	139599
Macro avg	0.85	0.84	0.84	391950
Weighted avg	0.85	0.85	0.85	391950
Accuracy			**0.85**	391950

Table 6.17: Evaluation metrics for TempTAC with a sliding window of 75 frames on stretched cropped CLS-tokens.

In Section 6.4.1, Centre-W75* demonstrated the best performance. The confusion matrix in Figure 6.7a shows more FPs for 'background' than Centre-W75*, which achieved a precision score of 0.96. Despite this model having 11,712 FPs, it still achieves



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.16: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from stretched frames with a sliding window of 75 frames.

a precision score of 0.86. Unlike most other models, the recall score for ’background’ is 0.70, a value previously matched or surpassed only by Centre-W25, Stretched-W25, and the simple CNN. Beating the simple CNNs’ F1-score of 0.75, this model matches the others with a score of 0.77. The AUC score in Figure 6.16b is 0.83 for ’background,’ just behind Centre-W25 (0.84). For ’tackle-live,’ both models are tied, while this model leads by 0.01 for ’tackle-replay.’ For ’tackle-live,’ a high recall score of 0.95 results in an F1-score of 0.88, despite a precision score of 0.82. Although not the highest score observed, it is still higher than the simple CNN and Meta-learner 1-3. For ’tackle-replay,’ the scores are well-balanced, with precision and recall scores of 0.87 and 0.86, respectively, as reflected in the ROC curve. This model, from this point, namely Stretched-W75*, outperforms Centre-W75* for ’background,’ but performs slightly worse for the tackle classes, ultimately achieving the same overall accuracy of 85%.

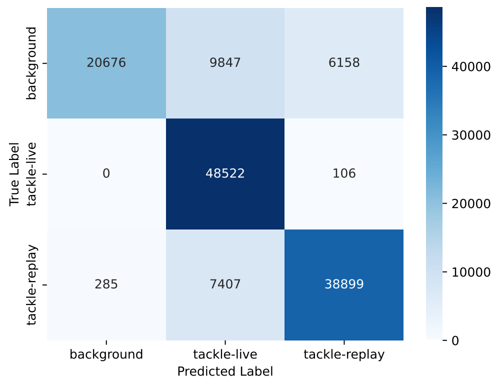
6.4.3 Padded Cropping

This section, like Section 6.4.3, contains results from experiments conducted using CLS-tokens derived from frames that were reflectively padded prior to DINOv2’s image processing.

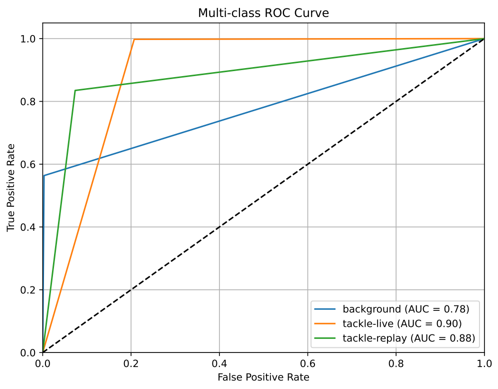
6.4.3.1 Window Size 25

The confusion matrix in Figure 6.17a and the ROC curve for ’background’ (blue) reveal a low rate of FPs. With only 285 FPs, a precision score of 0.91, as seen in Table 6.18, is justified. However, the TP rate is also low, reflected in the recall score of 0.57 due to 16,005 FNs, resulting in an F1-score of 0.70.

The ROC curve for ’tackle-live’ (orange) displays a very high TP rate, which the confusion matrix confirms, with only 106 FNs. Despite a recall score of 0.97, frequent misclassification of the other two classes as FNs reduces the F1-score to 0.88. This matches Centre-W25* but outperforms Stretched-W25* and the simple CNN.



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.17: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a sliding window of 25 frames.

Event	Precision	Recall	F1-score	Support
Background	0.99	0.56	0.72	36681
Tackle-live	0.74	0.99	0.85	48628
Tackle-replay	0.86	0.83	0.85	46591
Macro avg	0.86	0.80	0.80	131900
Weighted avg	0.85	0.82	0.81	131900
Accuracy			**0.82**	131900

Table 6.18: Evaluation metrics for TempTAC with a sliding window of 25 frames on reflected cropped CLS-tokens.

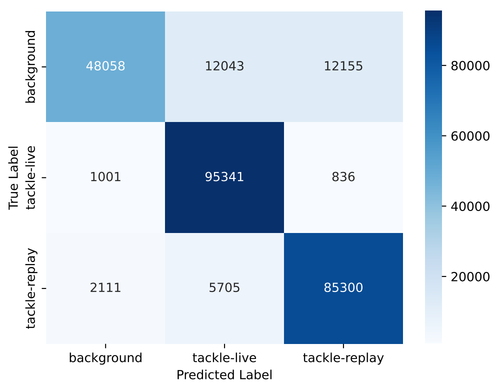
For 'tackle-replay,' the model has 6,264 FPs and 38,899 TPs, yielding a precision score of 0.86. With 7,692 FNs, the recall score is 0.83. The F1-score of 0.85 matches Centre-W25* and surpasses Stretched-W25*, but falls short of the simple CNN for 'tackle-replay.'

Although there are only 285 FPs and a record-high precision score of 0.99 for 'background,' the recall score is just 0.56 due to 16,205 FNs.

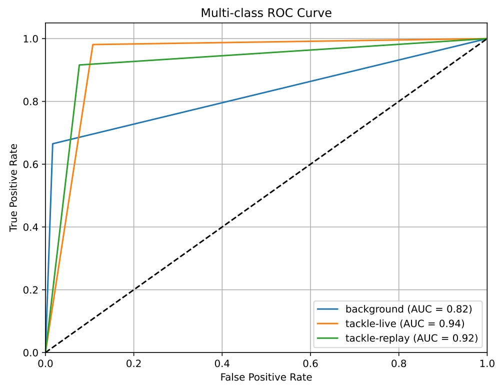
Overall, even though this model achieves high precision and recall scores (0.99 for 'background' precision and 0.99 for 'tackle-live' recall), the rest of its performance isn't strong enough to achieve more than 0.80 and 0.81 in macro and weighted averages, respectively. The overall accuracy of 0.82 matches Stretched-W25* but lags behind the simple CNN and Centre-W25*.

6.4.3.2 Window Size 50

When increasing the window size to 50, Table 6.20 shows an improvement in performance, as seen when doubling the window size from Stretched-W25*. The ROC curves in Figure 6.18b are quite similar to those of Stretched-W50* and can be difficult to distinguish visually. While Stretched-W25* achieved impressive F1-scores of 0.73, 0.90, and 0.87 for 'background,' 'tackle-live,' and 'tackle-replay,' this model outperforms it on all fronts with scores of 0.78, 0.91, and 0.89, respectively. This improvement is also reflected in the AUC scores.



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.18: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a sliding window of 50 frames.

Event	**Precision**	**Recall**	**F1-score**	**Support**
Background	0.94	0.67	0.78	72267
Tackle-live	0.84	0.98	0.91	97178
Tackle-replay	0.87	0.92	0.89	93116
Macro avg	0.88	0.85	0.86	262550
Weighted avg	0.88	0.87	0.87	262550
Accuracy			**0.87**	262550

Table 6.19: Evaluation metrics for TempTAC with a sliding window of 50 frames on reflected cropped CLS-tokens.

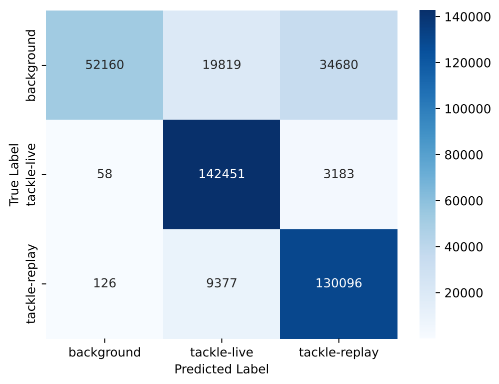
The confusion matrix in Figure 6.18a shows that 'tackle-live' had only 1,837 FNs and 95,341 TPs. This is reflected in the recall score from the table and the TP rate in the ROC curve (orange). In addition to the high recall, it also scores well in precision with 0.84, despite 17,748 FPs, resulting in a well-deserved F1-score of 0.91, the highest so far.

The model recalls 92% of all 'tackle-replay' instances, achieving 85,300 TPs and only 7,816 FNs. The recall score of 0.92 is the highest recorded and the first to outperform the simple CNN for 'tackle-replay' F1-score (0.89 vs. 0.88). It is also the first to match Meta-learner 1-3 in F1-score for 'background.' With a precision score of 0.94, only 3,112 FPs, and 48,058 TPs, it reaches an F1-score of 0.78 despite a high number of FNs (24,198).

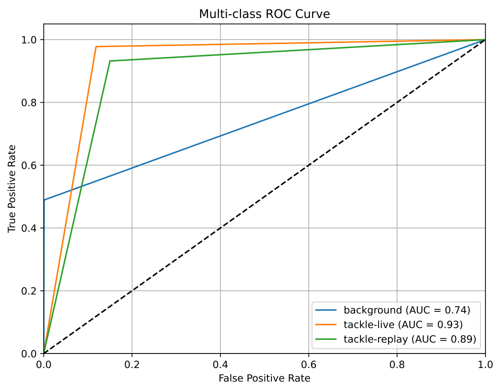
Overall, this model surpasses all others in our research, achieving higher averages across all metrics and a remarkable 87% overall accuracy.

6.4.3.3 Window Size 75

Looking at the ROC curves in Figure 6.19b, we see the blue curve representing 'background' all the way to the left. Table 6.20 reveals that this is due to 'background' having a precision score of 0.99, the highest recorded so far. There are only 184 FPs compared to 52,160 TPs, as shown in the confusion matrix in Figure 6.19a. However,



(a) Confusion Matrix.



(b) ROC Curve.

Figure 6.19: Confusion Matrix and ROC Curve for TempTAC using CLS-tokens from reflective padded frames with a sliding window of 75 frames.

Event	Precision	Recall	F1-score	Support
Background	0.99	0.49	0.65	106659
Tackle-live	0.83	0.98	0.90	145692
Tackle-replay	0.77	0.93	0.85	139599
Macro avg	0.87	0.80	0.80	391950
Weighted avg	0.86	0.83	0.81	391950
Accuracy			**0.83**	391950

Table 6.20: Evaluation metrics for TempTAC with a sliding window of 75 frames on reflected cropped CLS-tokens.

there are also 54,499 FNs, which is reflected in both the ROC curve and the recall score of 0.49, the lowest recorded. This leads to an F1-score of 0.65, which falls short of other models, even trailing behind Stretched-W75 and Padded-W75, both of which scored 0.67.

For ’tackle-live’ and ’tackle-replay,’ the ROC curves are quite high, indicating a strong TP rate, which is also evident in the confusion matrix. ’Tackle-live’ has 3,241 FNs but an impressive 142,451 TPs, resulting in a recall score of 0.98, meaning the model captures 98% of instances in this class, with a precision score of 0.83.

’Tackle-replay’ struggles somewhat more, achieving a recall score of 0.93 and a precision score of only 0.77. This results in an F1-score of 0.85, which is no better than the simple CNN and on par with Meta-learner 1-3. In contrast to the other tackle class, ’tackle-replay’ achieves an F1-score of 0.90, which is among the higher scores.

Overall, this model’s performance is less impressive and less balanced than several others evaluated.

6.5 (End-to-End) System Performance

Similar to the system performance evaluation for the spatial approach in Section 5.4, a comparable evaluation for TempTAC will be detailed. This includes assessing performance when running solely on my local machine's CPU running with a 2.6 GHz 6-core Intel Core i7 with 16 GB RAM, as well as on the available NVIDIA GPUs.

6.5.1 Shifting Window Inference

Given the different window sizes, it was decided to evaluate TempTAC for each of these. Running TempTAC on a CPU with a window size of 25 took 2.05 seconds on the same test set used for evaluating the models. This processed 5,300 CLS-tokens, meaning each frame took 0.00038 seconds, or 0.0095 seconds per second of video. Seen in Section 5.4, DINOv2 used 2.318 seconds to extract features from one frame on a CPU. Multiplying this by the FPS (25) and adding the inference for 25 frames, results in a total processing time of 57.96 seconds per second of video for end-to-end processing. In comparison, the total time on the GPUs was 0.0281 seconds for the test set, almost 75 times faster. With DINOv2 handling the feature extraction exploiting the GPU, this equals 1.18075 seconds per second of video, a big difference from the CPU.

When the window size increased to 50, inference time also increased. LSTMs need to forward their hidden states, and with a larger sequence, managing these states increases inference time. Processing the entire test set took 2.2734 seconds on the CPU and 0.0494 seconds on the GPU. Adding feature extraction, the total remains approximately the same, as the changes are marginal for both CPU and GPU.

With a window size of 75, inference took 2.4947 seconds on the CPU and 0.0642 seconds on the GPU, which again doesn't affect the overall time per second of video.

6.5.2 Sliding Window Inference

When using a sliding window, the number of calculations increases, resulting in a greater computational burden, power consumption, and processing time. On the same test set as the shifting window, the CPU required 53.3560 seconds for a sliding window size of 25, while the GPU only needed 0.6298 seconds. With a sliding window, most frames are processed multiple times due to overlap, so averages are used. On average, the CPU took 0.0101 seconds to process each frame, which equals 0.2525 seconds for 25 frames (one second). Adding this to the DINOv2 feature extraction time of 57.95 seconds, the average end-to-end processing time for one second of video is 58.20 seconds on the CPU.

On the GPU, the computation time for 25 CLS-tokens is so minimal that it does not impact the overall time by much, adding just 0.002 seconds to the 1.17 seconds required for DINOv2 feature extraction.

Increasing the sliding window size to 50, CPU processing time rose to 68.63 seconds per second of video, while the GPU required just 1.311 seconds. While the CPU time to do isolated inference increased by over 4 seconds with the larger window, the GPU time increased by only around 0.5 seconds. With a window size of 75, the CPU used 74.2 seconds per second of video to do an end-to-end, including the feature extraction, compared to only 1.385 seconds for the GPU. This demonstrates that not only does

GPU usage enhance performance, but it is also much more scalable. Notably, the biggest computational burden lies with DINOv2.

6.5.3 Memory Usage

While the meta-learners and base models needed a total of 0.857 MB for storage, TempTAC requires more memory due to its parameter requirements for managing sequences. With a size of 33.3 MB, TempTAC is over 38 times larger than the spatial-based model.

6.6 Chapter Summary

This chapter examines the effectiveness of the TempTAC model using temporal features for tackle detection in soccer videos, applying shifting and sliding windows on CLS-tokens representing images processed through various methods such as centre cropping, stretching, and padding. An assessment of the system's performance was conducted with window sizes of 25, 50, and 75 for each window type to compare the impact of sequence length on model performance. Performance tended to decrease with larger window sizes, underscoring the challenges of managing longer sequences.

In Chapter 7, these results will be discussed along with an examination of the different challenges and limitations faced.

Chapter 7

Discussion

This chapter discusses the results, limitations, and challenges encountered during the experiments in Chapters 5 and 6, where models were developed for the automatic detection of tackles using solely spatial information or in combination with temporal features.

7.1 Importance of Well-curated Datasets

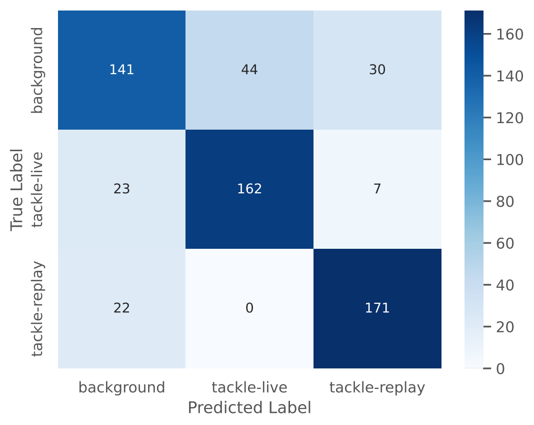
Working with machine learning, particularly supervised learning, underscores the importance of using high-quality labelled data. For a model to generalize well and remain robust, the dataset must be both sufficient in size and diverse in composition. While edge cases will inevitably exist, careful consideration of these factors during dataset curation is crucial.

Through the creation of TACDEC, a dataset was developed to further push research in event detection in soccer, addressing the lack of similar datasets in existing publicly available datasets. TACDEC offers 425 labelled videos of tackle events from the Norwegian Eliteserien, created through a partnership with ForzaSys and NTF (Norsk Toppfotball). The dataset encompasses tackle events from 19 different teams, totalling 836 instances, and includes the variety of camera settings used in broadcast videos.

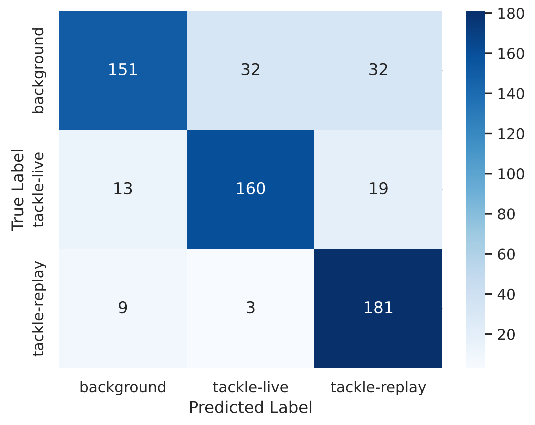
7.2 Spatial Approach

This section will discuss the results and some of the limitations and challenges encountered throughout research on the spatial approach.

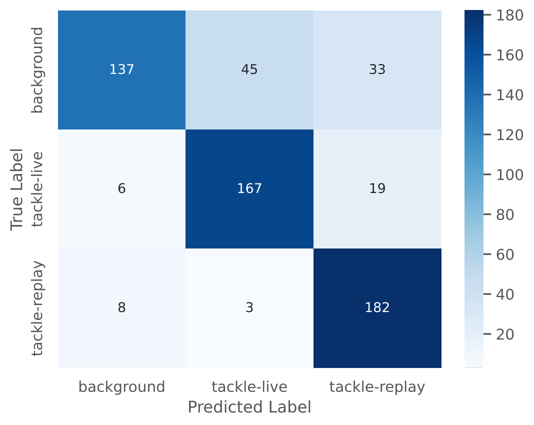
7.2.1 Comparison of Meta-learners



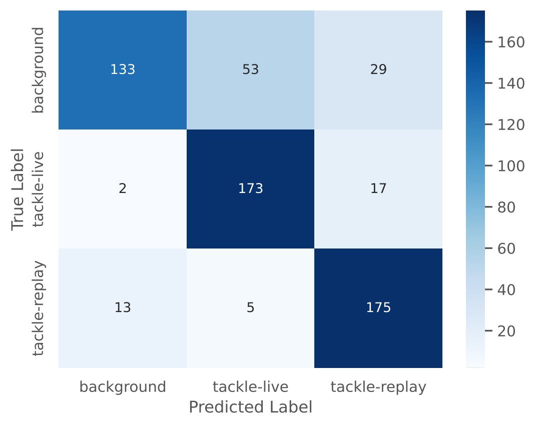
(a) Baseline



(b) Meta-learner 1-3



(c) Meta-learner 4-6



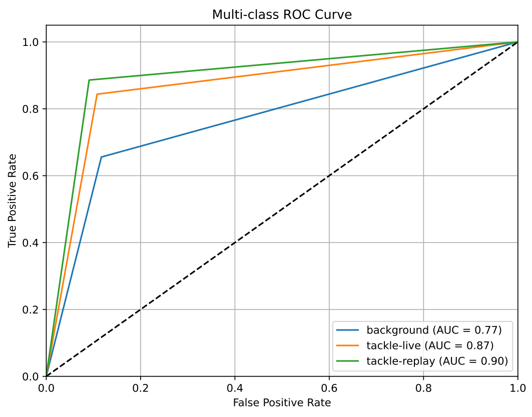
(d) Meta-learner 7-9

Figure 7.1: Confusion matrices for the baseline and meta-learners.

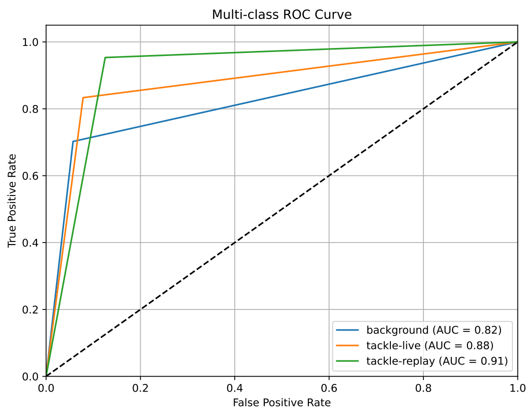
The evaluation of base models and meta-learners, as detailed in Figures 7.1 and 7.2, reveals that all meta-learners effectively identify 'tackle-replay' instances, achieving high recall scores. Specifically, Meta-learners 1-3 and 4-6 report a recall of 0.94, marginally higher than Meta-learners 7-9's 0.91, all exceeding the baseline's recall of 0.89. These high recall rates indicate the successful utilization of specific base models' strengths in identifying the correct classes.

However, as shown in Figure 7.2, the precision for 'tackle-replay' across meta-learners does not reach the higher levels observed in some base models and the baseline, due to inherent disagreements among the base models. This sacrifice in precision to maximize recall is illustrated in the confusion matrices in Figure 7.1, highlighting the trade-offs made.

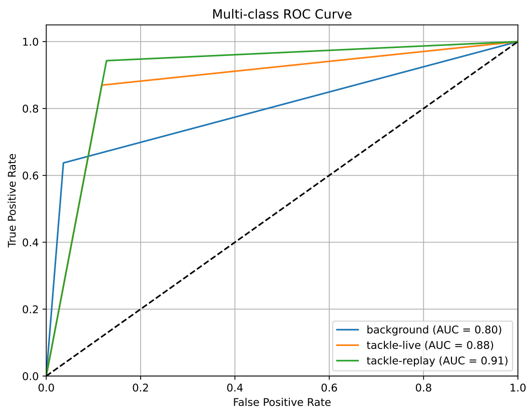
For 'tackle-live,' as detailed in Figure 7.1, Meta-learners 4-6 and 7-9 show more effective capture rates of instances, with higher recall scores compared to the baseline and Meta-learners 1-3.



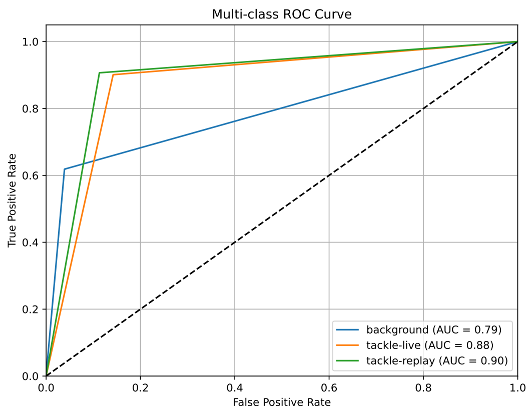
(a) Baseline



(b) Meta-learner 1-3



(c) Meta-learner 4-6



(d) Meta-learner 7-9

Figure 7.2: ROC Curves for the baseline and meta-learners.

Despite this, their precision is lower, reflecting a trade-off that is also captured in their confusion matrices.

The improvements in balancing precision and recall for 'background' are significant, surpassing the baseline in F1-score, as reflected in Figure 7.2. Furthermore, all meta-learners exhibit improvement in the AUC score for 'background,' compared to the baseline, as shown in Figure 7.2. This indicates a better overall trade-off between false positives and negatives.

Event	**Baseline**		**1-3**		**4-6**		**7-9**	
	Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Background	0.76	0.66	0.87	0.70	0.91	0.64	0.90	0.62
Tackle-live	0.79	0.84	0.82	0.83	0.78	0.87	0.75	0.90
Tackle-replay	0.82	0.89	0.78	0.94	0.78	0.94	0.79	0.91

Table 7.1: Consolidated evaluation scores among the meta-learners.

In conclusion, as illustrated across various figures and summarized in Figure 7.2, meta-learners that aggregate predictions from multiple base models offer distinct advantages, particularly in handling classes that are difficult to distinguish. This approach demonstrates the potential benefits of targeting individual classes with specialized models, although the overall enhancements may vary.

Event	**Baseline**	**1-3**	**4-6**	**7-9**
	F1-score	**F1-score**	**F1-score**	**F1-score**
Background	0.70	0.78	0.75	0.73
Tackle-live	0.81	0.83	0.82	0.82
Tackle-replay	0.85	0.85	0.85	0.85

Table 7.2: F1-scores among the meta-learners.

Selecting a Model

Selecting the optimal model for identifying tackles, whether they occur in replays or live, indeed hinges on prioritizing certain aspects of model performance, particularly the trade-offs between recall and precision.

For High Recall (Capturing as Many Tackles as Possible):

1. **Focus on replay tackles:**

   (a) **Meta-learners 1-3 & 4-6:** If the main objective is to capture as many instances of tackles in replays as possible and the tolerance for FPs is relatively high, then Meta-learners 1-3 or 4-6 would be excellent choices. Both achieve a high recall of 0.94, indicating their effectiveness in identifying most of the TP, even at the expense of capturing more FPs.

2. **Focus on live tackles:**

(a) **Meta-learner 7-9:** This meta-learner is well-suited for capturing 'tackle-live' instances with a high recall, making it ideal if the primary interest is in tackles occurring for the first time. However, it does come with a higher rate of FPs.

(b) **Meta-learner 1-3:** If reducing FPs is also a concern while still needing to effectively detect most 'tackle-live' instances, then Meta-learner 1-3 offers a balanced solution. It provides a good compromise between capturing TPs and minimizing errors, making it suitable for scenarios where both aspects are important.

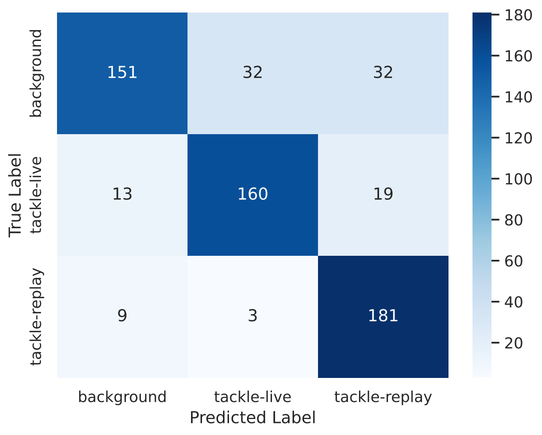
For High Precision (Minimizing FPs):

1. **Meta-Learner 1-3:** If the priority shifts towards minimizing FPs, ensuring that the tackles identified are highly likely to be correct, then Meta-Learner 1-3 proves decent, with the fewest FPs among the Meta-learners.

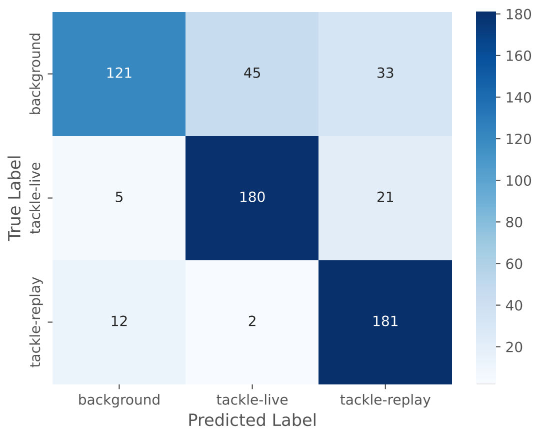
In conclusion, the choice of model should align with the specific requirements of the task at hand. Whether it is maximizing the identification of tackles, ensuring the accuracy of those identifications, or finding a balance between these elements, each model offers distinct advantages that cater to different operational needs.

It is important to mention that this analysis does not take into account the computational resources each model requires as the differences were minimal. Based on the presented results, selecting the best model solely by overall performance would favour Meta-learner 1-3, which demonstrates the strongest results. Its AUC is equal to or greater than that of other models across all classes, offering the best trade-off between FPs and FNs.

7.2.2 Influence of Model Architecture



(a) Small base model Meta-learner 1-3

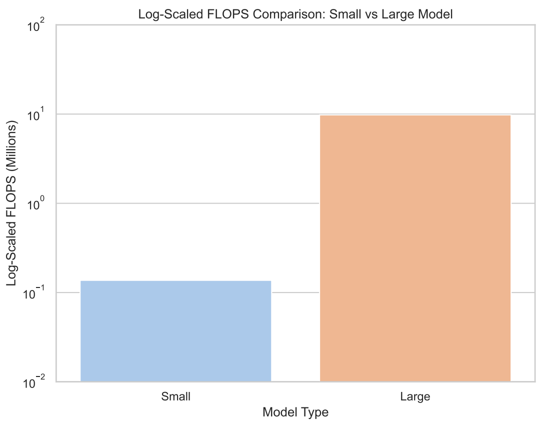


(b) Large base model Meta-learner 1-3*

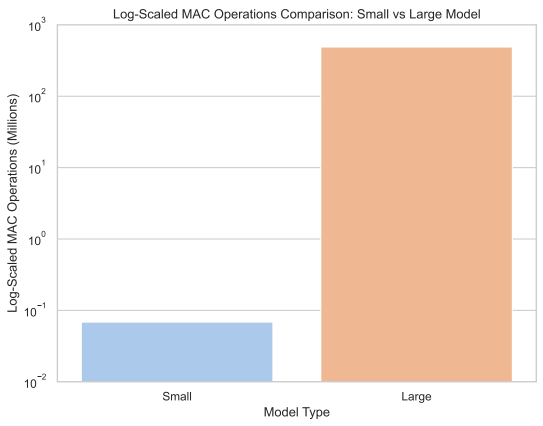
Figure 7.3: Comparison of confusion matrices for meta-learners built upon small and large base models.

Event	1-3			1-3*		
	Precision	Recall	F1-score	Precision	Recall	F1-score
Background	0.87	0.70	0.78	0.88	0.61	0.72
Tackle-live	0.82	0.83	0.83	0.79	0.87	0.83
Tackle-replay	0.78	0.94	0.85	0.77	0.93	0.84

Table 7.3: Comparison between Meta-learner 1-3 and 1-3*.



(a) A logarithmic comparison of the two models FLOPS required.



(b) A logarithmic comparison of the two models MAC operations performed.

Figure 7.4: System performance comparison between the small and large model.

To determine the optimal configuration of base models in terms of layers and nodes, two models were created: a large, complex model and a smaller, less complex one. Initial experiments showed that although the larger model performed decently through hyperparameter optimization, it did not yield significant improvements when integrated into the meta-learners compared to the smaller model. Given the large model's extensive parameter count of 4,894,723 versus the smaller model's 68,483 parameters, the latter was competitive and more efficient, influencing the decision to choose simplicity.

The computational efficiency of the two models was analyzed by focusing on their multiply-accumulate operations (Multiply-accumulates (MACs)), which are crucial in many deep-learning computations. The smaller model required approximately 68,000 MACs, while the larger model needed around 4.89 million MACs, see Figure 7.4b. This indicates that the larger model demands about 71.5 times more MAC operations, significantly increasing the computational load and impacting efficiency factors like training time and resource consumption.

Further, the Floating Point Operations Per Second (FLOPS) of both models were measured, reflecting their computational demands. Running on a local machine, Figure 7.4a reveals that the smaller model required about 137 KFLOPS, while the larger model necessitated 9.8 MFLOPS, again approximately 71.5 times higher. This not only affects execution time but also increases energy consumption and heat generation, which can lead to higher operational costs and the need for more robust cooling solutions.

Additionally, optimization times during the hyperparameter tuning phase were significantly different. The smaller model completed 150 trials in about 3 minutes, whereas the larger model took around 51 minutes—nearly 17 times longer. This substantial difference further highlights the practical advantages of a less complex model, especially in scenarios requiring frequent retraining or real-time applications.

When adjusting the training dataset size to 8,000 samples to ensure balance, the larger model showed a high tendency to overfit, while the smaller model still delivered competitive results. This, coupled with the efficiency gains from a reduced parameter count and the stability of a simpler training environment, underscored the decision to favour a more compact model architecture.

Lastly, the implementation of ReLU activation functions supports the use of the smaller model, as evidence suggests that such models can learn effectively even with fewer parameters. The smaller base model architecture was chosen for its balanced performance and simpler learning path, as detailed in Table 7.3.

7.2.3 Base Models vs. Meta-learners

Three different Meta-learners were developed using the predictions of three base models, trained to specialize in a class each. Despite this, we saw, like for Model 9 (targeting 'tackle-replay'), performing better for F1-score on 'tackle-live' than what Model 8 (targeting 'tackle-live') does. This suggests that there might be common features between replay and live tackles that Model 9 is capturing more effectively and, therefore, has learned a more generalized representation of the tackles. A key takeaway is that even though we saw a general improvement in the meta-learners, there were base models that performed very similarly and maybe even be preferred, like Model 4 (targeting 'background'), which has an overall accuracy of 0.83, above its Meta-learner 4-6 (0.81) but also the other two meta-learners.

In Chapter 5, the classification introduced distinct challenges compared to that of Chapter 6, as each frame is evaluated in isolation without considering temporal dependencies. This can lead to increased susceptibility to noise or slight variations in visual cues, which may result in misclassification. However, this approach provides greater flexibility for real-time or near-real-time applications by allowing immediate labelling of individual frames.

Despite these challenges, the models achieved high recall scores, emphasizing their robustness in identifying both live and replay tackle events.

7.2.4 Handling Imbalance

Given the imbalanced nature of TACDEC, where non-tackle frames are classified as background, addressing this imbalance was crucial. In trying to do so, different techniques were applied.

Initially, weighted loss functions were employed, adjusting the loss based on class distribution. However, this did not yield the desired results, leading to trials with oversampling the underrepresented classes.

To oversample these underrepresented classes, a technique called SMOTE (Synthetic Minority Over-sampling Technique) was first attempted. The idea of SMOTE is to map instances of similar classes based on features and then take random samples from a linear projection between instances from the same class in the hope of them being quite alike. Essentially, it tries to select samples that are close to the feature space.

Likely due to the use of the CLS-tokens as the feature vector, a phenomenon called 'the curse of dimensionality' is thought to have played its part. This phenomenon suggests that when working with high-dimensionality data, 1024-feature vector in this case, there are so many variables that the data becomes sparse and spread out in the feature space. It is as though each sample acts like a new corner, with empty spaces in between them.

To tackle this, one could use unsupervised methods, like PCA to reduce the dimensionality, but this also had little effect likely because of the low entropy within the data.

Trying another data generation algorithm called ADASYN without results led to a shift towards downsampling instead. Oversampling with duplicates was avoided because it did not increase the diversity of the underrepresented classes and would, therefore, be less helpful for the desired generalization.

After several unsuccessful attempts with different methods, the decision was made to undersample the overrepresented classes to achieve balance in the dataset. As detailed in Section 4.4.2.3, 4000 instances were sampled for the target class and 2000 instances for the other two classes. This strategy, combined with a random seed (used to initialize random generators to reproduce) for splitting the data into training, validation and test sets, ensured that models targeting the same classes across different crops were exposed to identical data.

7.3 Temporal Approach

This section will discuss the results and some of the limitations and challenges encountered throughout the research for the temporal approach.

7.3.1 Selecting a Model

As in Chapter 5, models will be proposed based on recall and precision importance, along with an 'efficient model' that requires minimal computational resources while maintaining good performance.
For High Recall (Capturing as Many Tackles as Possible):

1. **Focus on replay tackles:**
   (a) **Reflected-W50*:** This model is ideal if the primary goal is to capture as many instances of tackles in replays as possible, and there is some tolerance for FPs. With a recall of 92% for 'tackle-replay,' it excels in detecting these events. Additionally, it maintains a precision of 87% for 'tackle-replay,' making it a great choice for recalling replays of tackles.
   (b) **Reflected-W75*:** This model performs slightly better than Reflected-W50* with 93% against 92% for recalling 'tackle-replay', but offers lower precision for the same class with a 10% decrease.
2. **Focus on live tackles:**
   (a) **Reflected-W50*:** This model is well-suited for capturing 'tackle-live' instances with a high recall with 98%, while doing so with a precision of 84%. It notably also recalls 'tackle-replay' with 92% making it ideal if the primary interest is in tackles occurring for the first time while also proving strong for replays.
   (b) **Stretched-W50*** offers 99% recall for 'tackle-live' with a precision of 83% which offers a model that almost guarantees all TPs to be captured while also not adding too many FPs.

For High Precision (Minimizing FPs):

1. **Reflected-W50*:** Averaging the best precision with 94% for 'background', 84% for 'tackle-live', 87% for 'tackle-replay', while also recalling almost every live tackle as highlighted under the models for high recalls proving to be the most balanced model.

Efficient Model:

1. **Stretched-W25:** While being a much more efficient and smaller model, this model proves to have an overall great balance with an F1-score of 0.77 for 'background', 0.88 for 'tackle-live' and 0.86 for 'tackle-replay', it is capable of capturing 93% of live tackles with 84% precision and recall 83% of replay tackles with a precision of 89%. In addition, it has a great balance for 'background' with over 75% for both metrics.

7.3.2 Shifting vs. Sliding Window

Event	Reflected-W50			Reflected-W50*		
	Precision	**Recall**	**F1-score**	**Precision**	**Recall**	**F1-score**
Background	0.86	0.63	0.72	0.94	0.67	0.78
Tackle-live	0.82	0.99	0.90	0.84	0.98	0.91
Tackle-replay	0.84	0.84	0.84	0.87	0.92	0.89
Accuracy			**0.84**			**0.87**

Table 7.4: Consolidated evaluation scores Reflected-W50 and Reflected-W50*.

While the use of a sliding window led to a performance improvement over the shifting window method, it came at a cost. Section 6.5 assessed the performance of both techniques across two different systems. Although Reflected-W50* demonstrated strong performance, its reliance on a sliding window required great resources. Processing a CLS-token took 0.00025 seconds on a GPU and 0.013 seconds on a CPU.

In contrast, the shifting window method, such as in Reflected-W50, required only 0.0000093 seconds on a GPU and 0.00043 seconds on a CPU to accomplish the same task. This makes the shifting window approach approximately 30 times faster on the CPU and 28 times faster on the GPU. The large difference is due to the overlapping nature of the sliding window, while the shifting window jumps to completely new unseen data at every shift. Table 7.4 shows that impressive performance is still achievable using a shifting window, which is beneficial in situations with limited computational resources.

Observed variations in support within the classification reports are attributed to the window sizes used for training. When utilizing smaller window sizes of 25 and 50, it was possible to fully populate windows for 5300 data points. However, increasing the window size to 75 reduced the support to 5250, as the remaining samples could not fully form an additional window. This reduction in support occurs because each training window requires a fixed number of consecutive data points, and the larger window sizes need more data to complete, which limits the overall number of available windows for training.

7.3.3 Handling Imbalance and Background Frames

When handling the imbalance for the spatial approach, as described in Section 7.2.4, undersampling the overrepresented classes was done. In TACDEC, the distribution

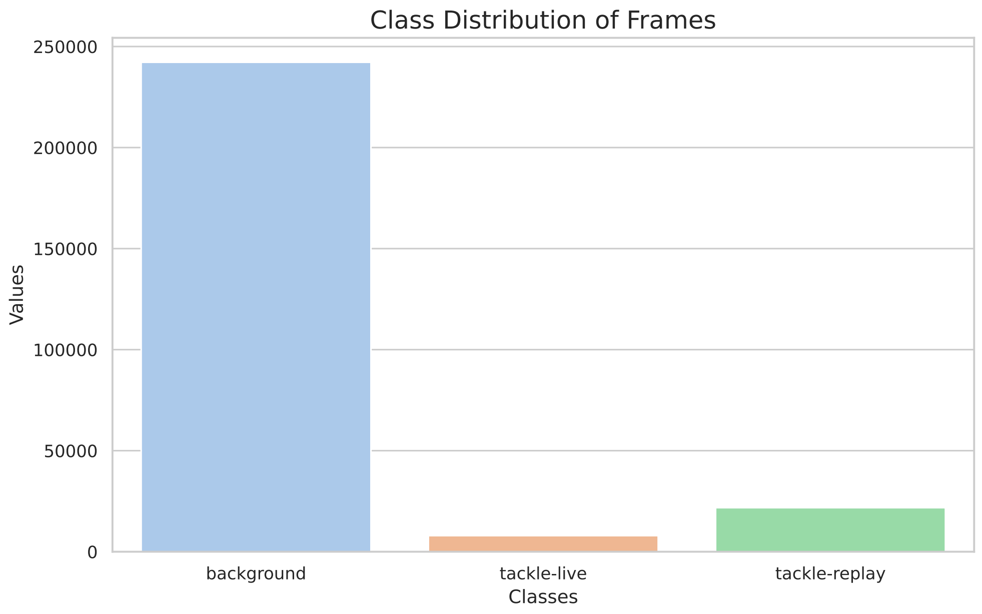


Figure 7.5: Distribution of frames given class when undersampled 'incomplete' instances to respective full classes.

Class ID	Average Sequence Length	Count	Frames
background	206.81	1171	242,180
tackle-live	21.88	363	7,941
tackle-replay	46.03	473	21,774

Table 7.5: Average Sequence Lengths and Counts by Class.

when sampling 'tackle-live-incomplete' and 'tackle-replay-incomplete' to their respective full classes is as seen in Figure 7.5. As we can see from the distribution, 'background' is far better represented than the other two classes with 242,187 frames, while 'tackle-live' has 7,941, and 'tackle-replay' has 21,774. This is also reflected in the number of sequences in Table 7.5.

Given the average sequence length of 206.81 frames, simply equalizing the number of sequences among classes was impractical. Instead, the dataset was balanced based on the total number of frames, effectively aligning the number of frames fed into the model.

Considering a strategy to randomly undersample the 'background' sequences to a length of 25 frames, equivalent to one second of video at the given FPS was implemented. This approach resulted in a more balanced dataset, as illustrated in Table 7.6. The undersampled dataset could then be split into training, validation, and test sets. This undersampling procedure was consistently applied across all crops.

Class ID	Average Sequence Length	Count	Frames
background	25.00	500	12,500
tackle-live	21.88	363	7,941
tackle-replay	44.90	280	12,573

Table 7.6: Average Sequence Lengths and Counts by Class after undersampling.

7.4 Common

This section will discuss some of the limitations and challenges encountered for both the spatial and temporal approaches.

7.4.1 Spatial Features vs. Spatio-Temporal Feature

Event	**1-3**		**Simple CNN**		**Stretched-W50***		**Reflected-W75***	
	Precision	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**	**Precision**	**Recall**
Background	0.87	0.70	0.73	0.78	0.86	0.70	0.94	0.67
Tackle-live	0.82	0.83	0.87	0.84	0.82	0.95	0.84	0.98
Tackle-replay	0.78	0.94	0.89	0.87	0.87	0.86	0.87	0.92

Table 7.7: Consolidated evaluation scores among the some of the best models.

Event	**1-3 F1-score**	**Simple CNN F1-score**	**Stretched-W75* F1-score**	**Reflected-W50* F1-score**
Background	**0.78***	0.75	0.77	**0.78***
Tackle-live	0.83	0.85	0.88	**0.91**
Tackle-replay	0.85	0.88	0.86	**0.89**
Macro avg	0.82	0.83	0.84	**0.86**
Weighted avg	0.82	0.83	0.85	**0.87**
Accuracy	0.82	0.83	0.85	**0.87**

Table 7.8: F1-scores and averages among the models presented in Table 7.7.

Section 7.2.1 concluded that Meta-learner 1-3 was the best and most balanced meta-learner. Table 7.7 compares Meta-learner 1-3, the simple CNN created as a baseline for the temporal approach, and two of the top-performing models from Chapter 6. After testing both a meta-learner without memory and an LSTM-based approach in TempTAC, we see that 'tackle-live' and 'tackle-replay' performance improves greatly when comparing Meta-learner 1-3 to the other three models. Table 7.8 shows that Reflected-W50* matches Meta-learner 1-3 in F1-score for 'background' and surpasses for both tackle classes, increasing overall accuracy from 0.82 to 0.87 by incorporating a temporal feature memory. The simple CNN and Stretched-W75* also outperform Meta-learner 1-3 overall. Figure 7.6 shows the class-wise probabilities predicted by Stretched-W50* in a window where the sequence shifts from 'background' to 'tackle-replay.' Despite the change in class, the model initially relies on the previous prediction and memory, even though there's evidence of a class change. However, with continued

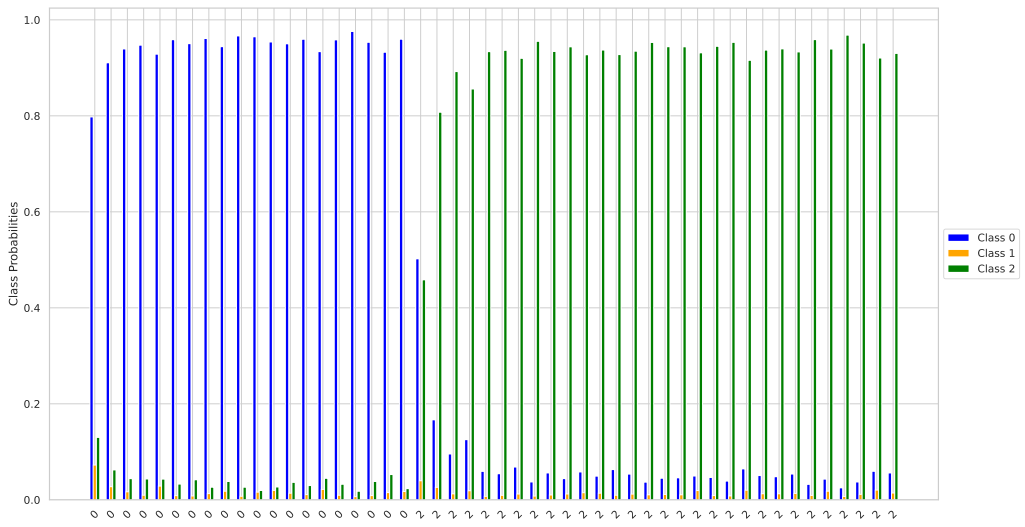


Figure 7.6: Illustration of the class-wise probabilities from Stretched-W50* with the true class along the x-axis. The sequence marks a shift from 'background' to 'tackle-replay.'

input, the shift occurs. It is also worth noting that this change, where the probability is distributed between two classes, could mark the starts and ends of sequences.

7.4.2 Preprocessing Prior to Feature Extraction

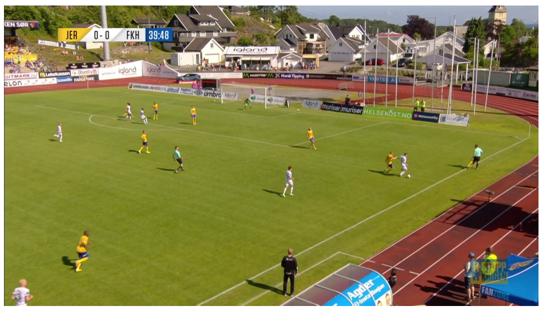
A part of this research was to see if it was possible to get more features out of the feature extraction by altering the frames in different ways to change the aspect ratio from 4:3 to 1:1, so that the centre cropping done by DINOv2's image processor did not cut out parts of the frame. Section 7.2.1 concluded that the best, most balanced model was Meta-learner 1-3, which used frames that were not altered. But a closer look in Table 7.1 reveals that despite this, both Meta-learner 4-6 using CLS-tokens from stretched frames and Meta-learner 7-9 using CLS-tokens from reflective-padded frames achieve a much higher recall for 'tackle-live' with 4-6 also achieving the same recall for 'tackle-replay'.

In Chapter 5, we observed that altering frames in various ways, particularly through stretching and padding by reflection, increased the average mutual information among the features. This indicates a stronger relationship between the features, effectively enhancing the informativeness of the data with respect to the CLS-token features. In simpler terms, this means that the features share more information.

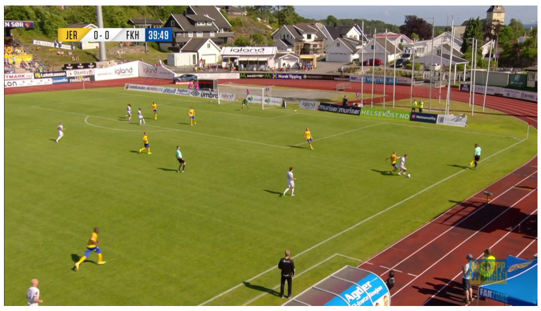
This observation was also evident in the performance of the base models. For unaltered frames, only Model 3 achieved averages and overall accuracy above 0.80. In contrast, for the stretched frames, all three base models (Model 4, 5, 6) achieved similar high metrics, whereas the models trained on padded frames displayed inconsistent results. Similarly, in Chapter 6, the most effective models, which were the ones recommended, were trained not on CLS-tokens from unaltered frames but rather on those from stretched and padded frames.

For additional analysis on the difference in CLS-tokens, see Chapter B.

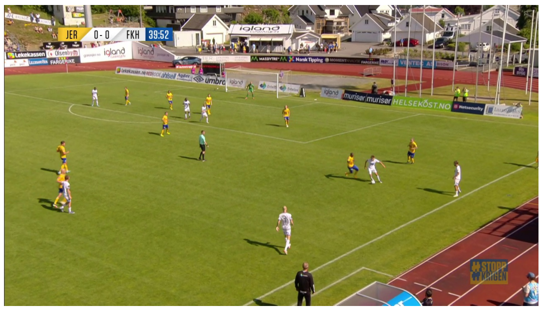
7.4.3 Influence of Shot Type



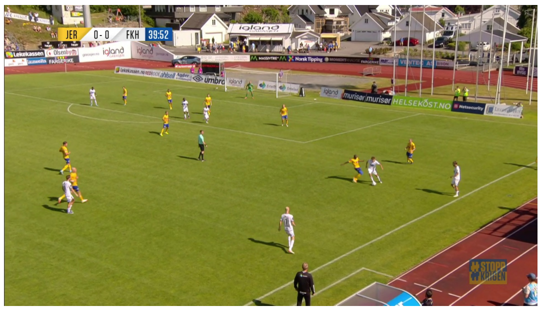
(a) ’background’



(b) ’tackle-live’



(c) ’background’



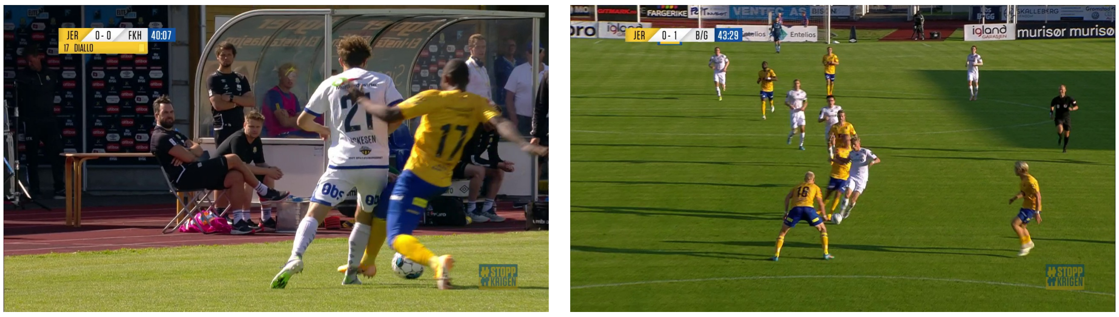
(d) ’tackle-live’

Figure 7.7: Figure displaying the similarities between some instances of ’background’ and ’tackle-live’ that might cause complications.

Throughout the research, it was observed that several base models and even the meta-learners struggled to differentiate between two specific classes: ’background’ and ’tackle-live’. This challenge is thought to stem from the high degree of similarity between some of the frames, which is illustrated in Figure 7.7. A close examination reveals that Figure 7.7a and Figure 7.7b are from the same game and are separated by only 3 frames. Given a frame rate of 25 frames per second (FPS), this time difference translates to just 0.12 seconds, or 120 milliseconds, meaning they occur within the same second. The same applies to Figure 7.7c and Figure 7.7d. Despite these temporal and visual similarities, images a) and c) are classified as ’background,’ while b) and d) are classified as ’tackle-live.’ This classification is based on the definition of a tackle sequence as outlined in Section 2.7, where the tackles in b) and d) have already been initiated and are, therefore, ongoing.

There is a huge difficulty in distinguishing these pairs of frames when viewed in isolation. Even for human observers, it can be challenging to determine whether a tackle has been initiated or if a player is about to make another move before engaging with an opponent. We also saw that despite it being hard to differentiate between ’background’ and the tackle classes for the TempTAC models in Chapter 6, there was a more balanced misclassification in contrast to what seen for the base models and the meta-learners, suggesting that the TempTAC models have a more nuanced understanding of the classes, despite the recall not improving too much.

For tackles that occur in replay, the story is a bit different. Contrary to the problem of 'background' and 'tackle-live,' 'tackle-replay' differs quite a bit. When looking at tackles being played in replay, different camera angles are often used, as seen in Figure 7.8a, where there the use of a close-up shot is popular, but also other views, such as seen in Figure 7.8b.



(a) Close-up View

(b) Goal-end View

Figure 7.8: Example displaying the shot type often occurring in replays.

7.4.4 Stochasticity

When working with machine learning, there is a lot of stochasticity in the algorithms and processes. This means that some processes involve a degree of randomness or uncertainty. For instance, when training a neural network, weights that are not initialized manually are often initialized randomly, which means that depending on the initial values, the model might vary despite being trained on the same data. Many machine learning algorithms, such as the Adam optimizer used, contain a stochastic element in updating its weights by random batches of training data, also meaning that the batch size in the data loaders directly affects Adam. This work recognizes that stochasticity can lead to varying training trajectories and, consequently, models with different performance characteristics. To manage this, Kaiming initializations were used to establish consistent starting points for model weights and set random seeds when splitting the dataset into training, validation, and test sets to maintain reproducibility. However, the choice was made not to fix random seeds for training itself to avoid overly constraining the optimization process. This allows the model to explore a wider range of training trajectories, promoting better generalization and potentially discovering more optimal solutions.

7.4.5 Data Leakage

When working on both experiments, data leakage had to be assessed. Data leakage means there is information about the target data in the training data. When this is the case, there will be a bias as the model has already seen this data or at least something very similar and we would see results that are overly optimistic and not a true representation of the models' performance, hence a poor evaluation. In this case, when working with videos, it was necessary to ensure that no frames were included in more than one set out of the training, validation, and test sets, which is standard practice. In addition to this, it was crucial to ensure that frames from the same sequences were not distributed across different sets, due to the high similarity between frames from the same sequence, particularly those close in time. To ensure that there was no data leakage, training, validation, and testing were also split by game. By taking this approach, it was ensured that no sequence from a particular game in the training set could be found in either the validation or test set, despite not being from the same sequence.

7.5 Limitations

The TACDEC dataset, which served as the training data for all models developed in this thesis, is limited to soccer videos from the Norwegian Eliteserien. This specificity could restrict the generalizability of the models to other leagues or sports, as it solely includes soccer footage. Moreover, the dataset primarily comprises clips initially labelled as yellow cards, potentially introducing bias since it excludes numerous non-penalized tackles, although several non-penalized tackles were labelled as they also appeared throughout the videos. This selection criterion of using clips of yellow cards may affect the dataset's representativeness.

All videos in the TACDEC dataset include audio features, which were not utilized in this research. Incorporating these features could potentially enhance performance. For instance, running Automated Speech Recognition (ASR) on commentary audio and using the textual commentary content (as discussed by curators of ASR datasets such as [18]) or analyzing game audio in videos without commentary might provide valuable insights. Game audio can reveal increases in audience volume, collisions, and even players' reactions to tackles, which could be crucial for detecting such events.

Additionally, variations in camera angles, exemplified in Figure 7.8, could impact model performance and its ability to generalize, particularly if teams with differing camera settings are promoted to the Eliteserien. The dataset's applicability is further constrained by weather conditions, as it includes no examples from matches played on snowy fields, which is often seen in locations like Tromsø.

All models are also constrained by their reliance on CLS-tokens. As discussed in Sections 5.4 and 6.5, this reliance necessitates substantial computational resources to function with low latency in near-real-time scenarios.

In this thesis, models that labelled each frame as either 'background,' 'tackle-live,' or 'tackle-replay' were evaluated. Some models achieved recall scores above 90% for both tackle events. It is important to note that this research and the models developed conducted a frame-by-frame classification, which differs from sequence classification. The goal was to determine whether the frame of interest was part of a tackle occurring live, in replay, or if neither occurred.

7.6 Chapter Summary

This chapter delves into the limitations and challenges encountered in developing and evaluating spatial and temporal models for the automatic detection of tackles in soccer videos, as explored in previous chapters. It discusses the creation and impact of the TACDEC dataset, which addresses a gap in available resources by providing a comprehensive collection of labelled tackle events. The dataset's limitations, including its focus on the Norwegian Eliteserien and its reliance on yellow card clips, highlight potential biases and the need for broader data to enhance model generalizability. Additionally, the chapter evaluates the computational efficiency and performance trade-offs between complex and simpler model architectures, illustrating the benefits of using less resource-intensive models without significant loss in performance. This discussion extends to the practical applications and potential expansions of this research, emphasizing how these models can be integrated into broadcast enhancements and amateur soccer analysis, potentially transforming how sports performances are analyzed and enjoyed.

The following chapter will reflect on the key lessons learned throughout this research, revisit the initial research questions and objectives, the contributions and propose directions for future research. This comprehensive review aims to encapsulate the insights gained and the potential for advancing this field further.

Chapter 8

Conclusion

In this thesis, the automatic detection of tackles in broadcast soccer videos using AI was investigated. This chapter presents insights, revisits and answers the research questions and objectives, suggests potential use cases and directions for future research, as well as presents the contributions made.

8.1 Insights and Lessons Learned

Over the course of this study, I have learned that tackles are events that can be hard to detect, given the strong similarities that some of the frames share. The similarities between a frame of regular gameplay and a frame where a tackle occurs live proved hard to differentiate when using the CLS-tokens as a frame representation. Despite this, we were able to develop several models showcasing performances that exceeded our expectations using a 1024-feature vector to represent these similar instances. I learned that by incorporating temporal features, the models could better understand tackles and the context in which they appear. By also exploring different methods of changing appearances and aspect ratios of frames, I saw an increase in the information present post feature extraction. This led to improved performance, as demonstrated by the standout models from Chapter 6.

8.2 Revisiting the Research Questions and Objectives

This section revisits and addresses the research questions introduced in Section 1.2, drawing on the findings from this study. The six subquestions and the research objectives presented are explored to support the answer to the main research question.

RQ1. Does developing a model that combines simpler models, each specifically targeting a distinct class, lead to performance improvements?
The short answer is yes, and we did see a general improvement in implementing these class-wise 'experts' across the overall balance between the classes. For Meta-learner 1-3, we saw an F1-score for 'background' higher than for all the other, which also proved to

be the hardest class to recall, but a deeper dive also reveals that base models, such as Model 3 had an F1-score of 0.87 for 'tackle-replay', which is higher than Meta-learner 1-3. The same story does not go for Meta-learner 4-6, with all base models with an F1-score of 0.83 or higher for 'tackle-live', Meta-learner 4-6 only achieved 0.82, but with a higher recall (0.94) for the same class than seen at any of its base models. But then again, for Meta-learner 7-9, neither of the base models saw an F1-score of 0.69, while the Meta-learner achieved 0.73, most likely from balancing between the high precisions of Model 8 (0.87) and Model 9 (0.91) and the higher recall of Model 7 (0.74). All the meta-learners beat the baseline for 'background' and 'tackle-live' while matching it for 'tackle-replay'.

RQ2. What is the benefit of employing recurrent neural networks with memory capabilities compared to models lacking such recurrent structures?
As outlined in Section 7.4.1, incorporating memory enhanced the performance by enabling the contextualization of each frame within a broader sequence and using this to make predictions more accurately.

RQ3. What are the trade-offs with using a sequence-based model using a sliding vs. a shifting window over an input sequence?
Although transitioning from a shifting window to a sliding window configuration resulted in performance enhancements, it also required higher computational costs. Observations confirm that, even with limited computational resources, constructing a model that achieves an overall accuracy of 84% is feasible. This was accomplished by utilizing a shifting window approach, which, while more resource-efficient, has compromises in terms of precision, recall, and accuracy compared to the more resource-intensive TempTAC models operating with sliding windows.

RQ4. How well can classification be performed using features extracted from a pretrained feature extractor?
In Section 7.4.1, several well-performing models were developed, achieving F1-scores as high as 0.78, 0.91, and 0.89 for 'background,' 'tackle-live,' and 'tackle-replay,' respectively using a frame representation in the CLS-tokens extracted by DINOv2.

RQ5. How can different aspect ratios impact learning and is there a way that is more effective than the regular centre cropping?
As outlined in Section 7.4.2, changing the way frames appear when fed to an image processor that uses the centre crop to obtain a 1:1 aspect ratio prior to DINOv2 impacted the overall results. By changing the aspect ratio from 4:3, where a centre crop normally would cut out information, to an aspect ratio of 1:1, the performance increased in most cases in Chapter 5 and 6, while all the stand-out TempTAC models were using the alternative methods.

RQ6. Can a dataset that enables training a model to recognize different types of tackle events in soccer broadcast video be created?
Throughout this thesis, we developed the TACDEC dataset, which enabled us to create several robust models capable of detecting tackles occurring live and distinguishing these from tackles occurring in replays.

Research Objectives
Our research objectives included investigating existing datasets on tackle events in soccer. Finding none that comprehensively covered tackles (existing datasets labelled fouls, which are not always tackles and are typically penalized), we aimed to advance soccer event detection research by creating our own dataset, TACDEC. We also explored whether variations in lighting conditions, camera angles, and player appearances, common in sports broadcasts, hinder learning. Our findings with the TACDEC dataset suggest that these factors do not impede model performance. In fact, the 'tackle-replay' class benefited from varying camera angles and lighting conditions, making it easier to distinguish from other classes. Conversely, the challenges were more pronounced when different classes occurred under similar conditions, as for 'background' and 'tackle-live', highlighting the nuances in effectively training our models.

We will use the findings from our subquestions to answer our main research question:

Main Research Question:
How can a framework be developed for automatically detecting tackle events in soccer videos using ML?

By creating a dataset with labelled tackle sequences, we were able to extract CLS-tokens using DINOv2. Utilizing these tokens as a representation for the frames, we developed two frameworks: one focused solely on spatial features using ensemble learning, and another that incorporates temporal features by leveraging LSTMs. Both frameworks demonstrated strong performances. We saw that the resources we had available, we could see a model running with a latency of just .17 per second with 87% overall accuracy. With the current setup, there is then approximately a 10-second delay for processing a minute of video. This translates to a delay of about 7.5 minutes for processing one half of a soccer game (45 minutes). Consequently, if a tackle occurs just before the halftime break, it would be processed before the start of the second half. Although this is not within seconds of a tackle, suggested in the motivation in Section 1.1, we conclude, based on our observations, that the framework developed successfully enables the automatic detection of tackle events in soccer videos using machine learning, with timings that could comfortably fit within typical game intermissions.

8.3 Potential Use Cases

Through this research, a new dataset, TACDEC, was developed in Chapter 3, which is the first dataset offering labelled tackle sequences in full offering both labelled tackles that occur in live and replay. It is fair to anticipate that TACDEC will advance research in tackle detection. Further, the models developed can serve as a tool for broadcast enhancement, which could automatically generate highlights of tackles and more engaging fan content without the need for manual labelling as a cover for the limitation of AI-producer presented in Section 2.2. These models could also streamline the labour-intensive process of sequence labelling, offering model-assisted labelling, where sequences detected by the model are only manually reviewed, thereby saving time and reducing costs. The same idea can be used to further expand the dataset to create and offer a dataset even larger. When integrated with player-tracking technology, these models could be utilized by coaches and players for analysis, enabling the extraction of tackles performed by specific players. This capability would present a method for analysing player performances and tracking development, particularly in amateur soccer where such analysis is yet to be proposed. Furthermore, considering the limited number of referees in amateur leagues, the models developed could be integrated with affordable technology to provide a basic, fully automated Video Assistant Referee (VAR) system, leveraging both traditional soccer event detectors and the tackle detection models.

8.4 Future Work

Through our research, we saw several strong performances, with the TempTAC impressing the most, and more specifically Reflected-W50* with an overall accuracy of 87%. As we presented in Chapter 4, we use CLS-tokens generated by feature extraction from DINOv2 as a representation for each frame. Generating these CLS-tokens takes, as we outlined in Section 5.4 and Section 6.5, a considerable amount of time unless there are sufficient resources. When employing DINOv2, we utilized one of its largest models, encompassing 304 million parameters. It would be interesting to explore how well the models perform using CLS-tokens extracted by smaller DINOv2 versions—specifically, the small version with 22.1 million parameters and the base version with 86 million parameters. Reducing the model size could decrease processing time, potentially offering a favourable trade-off between computational resource requirements and performance. In future research, it would be valuable to investigate the performance of models developed without using CLS-tokens, which have been central to our methodology. Specifically, employing raw frames directly with convolutional filters to extract features represents a fundamentally different approach to handling the data. This method would leverage the intrinsic capability of convolutional neural networks CNN to capture spatial hierarchies in the image data (like we saw the simple CNN was able to do on the CLS-tokens), which might prove effective in recognizing the complex patterns such as tackles from various angles in soccer offers. Exploring this approach could provide insights into the comparative advantages or disadvantages of using pre-defined embeddings like CLS-tokens versus direct feature learning from raw data. When a thorough analysis of both computational resources and whether using CLS-tokens is more beneficial as frame representations, a framework could be implemented into the AI-Producer presented in Section 2.2 to extend the current automated highlights offered. Furthermore, it would also be interesting to explore the expansion of the TACDEC dataset by incorporating additional labelled data from various leagues. This enhancement could improve both the robustness and the generalization capabilities of the dataset.

8.5 Contributions

Through this thesis, we have advanced the field of CV within soccer, specifically by researching and providing a dataset that uniquely labels entire videos with tackle events, distinguishing between live plays and replays. This pioneering dataset has been recognized and published by the ACM [1], underscoring the importance of our contributions. The paper and poster assosiated with the publication can be found in Chapter A in the appendix. More importantly, this contribution includes all artefacts available for other researchers to use TACDEC [2] [3] with also a Jupyter Notebook for doing feature extraction of the CLS-tokens using DINOv2. We explored how visual modifications prior to feature extraction—such as stretching or padding images—affect the data passed through for feature extraction. Our research contributed valuable insights by demonstrating that these modifications can impact model performance. The results we saw for TempTAC clearly proved that such preprocessing techniques can enhance the quality and relevance of the features extracted, leading to improved model accuracy and robustness. Additionally, we investigated the use of CLS-tokens from ViT (an aggregate representation for different image patches processed by transformers) in CV for classification tasks. This approach parallels techniques commonly used in natural language processing, highlighting the versatility and potential of transformer architectures in diverse applications. Jupyter Notebooks used for both the spatial approach in Chapter 5 and the temporal approach in Chapter 6 is available through GitHub [4].

[1]https://dl.acm.org/doi/10.1145/3625468.3652166
[2]https://huggingface.co/datasets/SimulaMet-HOST/TACDEC
[3]https://zenodo.org/records/10611979
[4]https://github.com/simula/forzify

Bibliography

[1] Rockson Agyeman, Rafiq Muhammad and Gyu Sang Choi. 'Soccer Video Summarization Using Deep Learning'. In: *2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. 2019 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR). San Jose, CA, USA: IEEE, Mar. 2019, pp. 270–273. ISBN: 978-1-72811-198-8. DOI: 10.1109/MIPR.2019.00055. URL: https://ieeexplore.ieee.org/document/8695329/ (visited on 03/05/2024).

[2] Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta and Masanori Koyama. *Optuna: A Next-generation Hyperparameter Optimization Framework*. 25th July 2019. arXiv: 1907.10902 [cs, stat]. URL: http://arxiv.org/abs/1907.10902 (visited on 11/04/2024). preprint.

[3] Ksenia Balabaeva and Sergey Kovalchuk. 'Comparison of Temporal and Non-Temporal Features Effect on Machine Learning Models Quality and Interpretability for Chronic Heart Failure Patients'. In: *Procedia Computer Science* 156 (2019), pp. 87–96. ISSN: 18770509. DOI: 10.1016/j.procs.2019.08.183. URL: https://linkinghub.elsevier.com/retrieve/pii/S1877050919311020 (visited on 15/03/2024).

[4] *ChatGPT*. URL: https://chatgpt.com (visited on 14/05/2024).

[5] *CVAT*. URL: https://www.cvat.ai/ (visited on 12/02/2024).

[6] Adrien Deliege, Anthony Cioppa, Silvio Giancola, Meisam J Seikavandi, Jacob V Dueholm, Kamal Nasrollahi, Bernard Ghanem, Thomas B Moeslund and Marc Van Droogenbroeck. 'Soccernet-v2: A dataset and benchmarks for holistic understanding of broadcast soccer videos'. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, pp. 4508–4519. (Visited on 19/11/2023).

[7] Peter J. Denning, Douglas E Comer, David Gries, Michael C. Mulder, Allen Tucker, A. Joe Turner and Paul R Young. 'Computing as a discipline'. In: *Computer* 22.2 (1989), pp. 63–70. (Visited on 14/11/2023).

[8] 'Digital Consumer Trends 2023_Key Highlights'. In: (2023). (Visited on 25/11/2023).

[9] Sayed Dorcheh, Mehdi Sarkhoosh, Cise Midoglu, Saeed Sabet, Tomas Kupka, Michael Riegler, Dag Johansen and Pål Halvorsen. *SmartCrop: AI-Based Cropping of Soccer Videos.* 2023. (Visited on 27/04/2024).

[10] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit and Neil Houlsby. *An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale.* 3rd June 2021. arXiv: 2010.11929 [cs]. URL: http://arxiv.org/abs/2010.11929 (visited on 26/04/2024). preprint.

[11] *Facebook/Dinov2-Large · Hugging Face.* 16th Jan. 2024. URL: https://huggingface.co/facebook/dinov2-large (visited on 29/03/2024).

[12] Jiale Fang, Calvin Yeung and Keisuke Fujii. *Foul Prediction with Estimated Poses from Soccer Broadcast Video.* 14th Feb. 2024. arXiv: 2402.09650 [cs]. URL: http://arxiv.org/abs/2402.09650 (visited on 06/05/2024). preprint.

[13] *Forzify.* URL: https://www.forzasys.com/Forzify.html (visited on 18/03/2024).

[14] Kunihiko Fukushima. 'Neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position'. In: *Biological cybernetics* 36.4 (1980), pp. 193–202. (Visited on 04/12/2023).

[15] Sushant Gautam. 'AI-based Soccer Game Summarization: From Video Highlights to Dynamic Text Summaries'. 27th Sept. 2022. DOI: 10.13140/RG.2.2.24627.91681. (Visited on 28/03/2024).

[16] Sushant Gautam, Cise Midoglu, Saeed Shafiee Sabet and Dinesh Baniya Kshatri. 'Assisting Soccer Game Summarization via Audio Intensity Analysis of Game Highlights'. In: (). (Visited on 19/04/2024).

[17] Sushant Gautam, Cise Midoglu, Saeed Shafiee Sabet, Dinesh Baniya Kshatri and Pål Halvorsen. 'Soccer Game Summarization Using Audio Commentary, Metadata, and Captions'. In: *Proceedings of the 1st Workshop on User-centric Narrative Summarization of Long Videos.* MM '22: The 30th ACM International Conference on Multimedia. Lisboa Portugal: ACM, 10th Oct. 2022, pp. 13–22. ISBN: 978-1-4503-9493-2. DOI: 10.1145/3552463.3557019. URL: https://dl.acm.org/doi/10.1145/3552463.3557019 (visited on 15/03/2024).

[18] Sushant Gautam, Mehdi Houshmand Sarkhoosh, Jan Held, Cise Midoglu, Anthony Cioppa, Silvio Giancola, Vajira Thambawita, Michael A. Riegler, Pål Halvorsen and Mubarak Shah. *SoccerNet-Echoes: A Soccer Game Audio Commentary Dataset.* 2024. arXiv: 2405.07354 [cs.SD].

[19] Silvio Giancola, Mohieddine Amine, Tarek Dghaily and Bernard Ghanem. 'SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).* 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW). Salt Lake City, UT: IEEE, June 2018, pp. 1792–179210. ISBN: 978-1-5386-6100-0. DOI: 10.1109/CVPRW.2018.00223. URL: https://ieeexplore.ieee.org/document/8575386/ (visited on 04/03/2024).

[20] Xavier Glorot and Yoshua Bengio. 'Understanding the Difficulty of Training Deep Feedforward Neural Networks'. In: *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics.* Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics. JMLR Workshop and Conference Proceedings, 31st Mar. 2010, pp. 249–256. URL: https://proceedings.mlr.press/v9/glorot10a.html (visited on 19/04/2024).

[21] *Glossary - Football Terms | IFAB.* URL: https://www.theifab.com/laws/latest/glossary/football-terms/ (visited on 12/03/2024).

[22] Peng Gong, Danielle J. Marceau and Philip J. Howarth. 'A Comparison of Spatial Feature Extraction Algorithms for Land-Use Classification with SPOT HRV Data'. In: *Remote Sensing of Environment* 40.2 (1992), pp. 137–151. ISSN: 0034-4257. DOI: 10.1016/0034-4257(92)90011-8. URL: https://www.sciencedirect.com/science/article/pii/0034425792900118 (visited on 09/01/2024).

[23] *GPT UiO – UiOs personverntrygge KI-chat - Universitetet i Oslo.* URL: https://www.uio.no/tjenester/it/ki/gpt-uio/index.html (visited on 14/05/2024).

[24] Tomoki Haruyama, Sho Takahashi, Takahiro Ogawa and Miki Haseyama. 'Similar Scene Retrieval in Soccer Videos with Weak Annotations by Multimodal Use of Bidirectional LSTM'. In: *Proceedings of the 2nd ACM International Conference on Multimedia in Asia.* MMAsia '20: ACM Multimedia Asia. Virtual Event Singapore: ACM, 7th Mar. 2021, pp. 1–8. ISBN: 978-1-4503-8308-0.

DOI: 10.1145/3444685.3446280.
URL: https://dl.acm.org/doi/10.1145/3444685.3446280 (visited on 03/05/2024).

[25] David Haynes, Steven Corns and Ganesh Kumar Venayagamoorthy. ‘An Exponential Moving Average Algorithm’. In: *2012 IEEE Congress on Evolutionary Computation.* 2012 IEEE Congress on Evolutionary Computation (CEC). Brisbane, Australia: IEEE, June 2012, pp. 1–8. DOI: 10.1109/CEC.2012.6252962. URL: http://ieeexplore.ieee.org/document/6252962/ (visited on 15/05/2024).

[26] Kaiming He, Xiangyu Zhang, Shaoqing Ren and Jian Sun. *Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification.* 6th Feb. 2015. arXiv: 1502.01852 [cs]. URL: http://arxiv.org/abs/1502.01852 (visited on 15/03/2024). preprint.

[27] Sepp Hochreiter and Jürgen Schmidhuber. ‘Long Short-Term Memory’. In: *Neural Computation* 9.8 (Nov. 1997), pp. 1735–1780. ISSN: 0899-7667. DOI: 10.1162/neco.1997.9.8.1735. eprint: https://direct.mit.edu/neco/article-pdf/9/8/1735/813796/neco.1997.9.8.1735.pdf. URL: https://doi.org/10.1162/neco.1997.9.8.1735 (visited on 20/03/2024).

[28] *How NASA JPL Employs AI to Advance Our Understanding of Martian Frost Patterns.* URL: https://labelbox.com/customers/nasa-jpl-martian-frost-map-customer-story/ (visited on 04/11/2023).

[29] *How Sportsbooks Use Opta Data To Make Major Tournaments More Entertaining.* Stats Perform. URL: https://www.statsperform.com/resource/opta-betting-casestudies/ (visited on 12/03/2024).

[30] David H Hubel and Torsten N Wiesel. *Brain and visual perception: the story of a 25-year collaboration.* Oxford University Press, 2004. (Visited on 10/12/2023).

[31] Andreas Husa, Cise Midoglu, Malek Hammou, Pål Halvorsen and Michael A. Riegler. ‘HOST-ATS: Automatic Thumbnail Selection with Dashboard-Controlled ML Pipeline and Dynamic User Survey’. In: *Proceedings of the 13th ACM Multimedia Systems Conference.* MMSys ’22. New York, NY, USA: Association for Computing Machinery, 5th Aug. 2022, pp. 334–340. ISBN: 978-1-4503-9283-9. DOI: 10.1145/3524273.3532908. URL: https://dl.acm.org/doi/10.1145/3524273.3532908 (visited on 15/03/2024).

[32] Andreas Husa, Cise Midoglu, Malek Hammou, Steven A. Hicks, Dag Johansen, Tomas Kupka, Michael A. Riegler and Pål Halvorsen. ‘Automatic Thumbnail Selection for Soccer Videos Using Machine Learning’. In: *Proceedings of the 13th ACM Multimedia Systems Conference.* MMSys ’22.

New York, NY, USA: Association for Computing Machinery, 5th Aug. 2022, pp. 73–85. ISBN: 978-1-4503-9283-9. DOI: 10.1145/3524273.3528182. URL: https://dl.acm.org/doi/10.1145/3524273.3528182 (visited on 15/03/2024).

[33] David Hutchison, Takeo Kanade, Josef Kittler, Jon M. Kleinberg, Friedemann Mattern, John C. Mitchell, Moni Naor, Oscar Nierstrasz, C. Pandu Rangan, Bernhard Steffen, Madhu Sudan, Demetri Terzopoulos, Doug Tygar, Moshe Y. Vardi, Gerhard Weikum, Moez Baccouche, Franck Mamalet, Christian Wolf, Christophe Garcia and Atilla Baskurt. ‘Action Classification in Soccer Videos with Long Short-Term Memory Recurrent Neural Networks’. In: *Artificial Neural Networks – ICANN 2010*. Ed. by Konstantinos Diamantaras, Wlodek Duch and Lazaros S. Iliadis. Vol. 6353. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010, pp. 154–159. DOI: 10.1007/978-3-642-15822-3_20. URL: http://link.springer.com/10.1007/978-3-642-15822-3_20 (visited on 03/05/2024).

[34] IMDb. *Louis Aimé Augustin Le Prince*. URL: https://www.imdb.com/name/nm1284117/bio/?ref_=nm_ov_bio_sm (visited on 17/11/2023).

[35] Yudong Jiang, Kaixu Cui, Leilei Chen, Canjin Wang and Changliang Xu. ‘SoccerDB: A Large-Scale Database for Comprehensive Video Understanding’. In: *Proceedings of the 3rd International Workshop on Multimedia Content Analysis in Sports*. 16th Oct. 2020, pp. 1–8. DOI: 10.1145/3422844.3423051. arXiv: 1912.04465 [cs]. URL: http://arxiv.org/abs/1912.04465 (visited on 14/03/2024).

[36] Glenn Jocher, Ayush Chaurasia and Jing Qiu. *Ultralytics YOLO*. Version 8.0.0. Jan. 2023. URL: https://github.com/ultralytics/ultralytics (visited on 04/03/2024).

[37] Gudberg Jonsson, Bjarkadottir, Baldvin Gislason Bern, Andy Borrie and Magnus Magnusson. ‘Detection of Real-Time Patterns in Sports: Interactions in Football’. In: 1st Jan. 2003, pp. 37–45. ISBN: 978-2-7237-0025-2. (Visited on 11/12/2023).

[38] Evan Jåsund Kassab, Håkon Maric Solberg, Sushant Gautam, Saeed Shafiee Sabet, Thomas Torjusen, Michael Riegler, Pål Halvorsen and Cise Midoglu. ‘TACDEC: Dataset of Tackle Events in Soccer Game Videos’. In: *Proceedings of the ACM Multimedia Systems Conference 2024 on ZZZ*. MMSys ’24: ACM Multimedia Systems Conference 2024. Bari Italy: ACM, 15th Apr. 2024, pp. 250–256. DOI: 10.1145/3625468.3652166. URL: https://dl.acm.org/doi/10.1145/3625468.3652166 (visited on 06/05/2024).

[39] Joe Kilian and Hava T. Siegelmann. ‘The Dynamic Universality of Sigmoidal Neural Networks’. In: *Information and Computation* 128.1 (July 1996), pp. 48–56. ISSN: 08905401.

DOI: 10.1006/inco.1996.0062.
URL: https://linkinghub.elsevier.com/retrieve/pii/S0890540196900620 (visited on 13/05/2024).

[40] *Labelbox | Data-centric AI Platform for Building & Using AI.*
URL: https://labelbox.com/ (visited on 12/02/2024).

[41] Hyunsu Lee.
'The Rise of ChatGPT: Exploring Its Potential in Medical Education'.
In: *Anatomical Sciences Education* n/a.n/a (). ISSN: 1935-9780.
DOI: 10.1002/ase.2270. URL:
https://onlinelibrary.wiley.com/doi/abs/10.1002/ase.2270 (visited on 14/05/2024).

[42] Andrzej Maćkiewicz and Waldemar Ratajczak.
'Principal Components Analysis (PCA)'.
In: *Computers & Geosciences* 19.3 (1993), pp. 303–342. ISSN: 0098-3004.
DOI: 10.1016/0098-3004(93)90090-R.
URL: https://www.sciencedirect.com/science/article/pii/009830049390090R (visited on 04/02/2024).

[43] *Midjourney.* Midjourney.
URL: https://www.midjourney.com/website (visited on 14/05/2024).

[44] Cise Midoglu, Saeed Shafiee Sabet, Mehdi Houshmand Sarkhoosh, Mohammad Majidi, Sushant Gautam, Håkon Maric Solberg, Tomas Kupka and Pål Halvorsen. 'AI-Based Sports Highlight Generation for Social Media'.
In: *Proceedings of the 3rd Mile-High Video Conference.* MHV '24.
New York, NY, USA: Association for Computing Machinery, 14th Mar. 2024, pp. 7–13. DOI: 10.1145/3638036.3640799.
URL: https://dl.acm.org/doi/10.1145/3638036.3640799 (visited on 14/03/2024).

[45] Lia Morra, Francesco Manigrasso, Giuseppe Canto, Claudio Gianfrate, Enrico Guarino and Fabrizio Lamberti. 'Slicing and Dicing Soccer: Automatic Detection of Complex Events from Spatio-Temporal Data'.
In: *Image Analysis and Recognition.*
Ed. by Aurélio Campilho, Fakhri Karray and Zhou Wang.
Lecture Notes in Computer Science.
Cham: Springer International Publishing, 2020, pp. 107–121.
ISBN: 978-3-030-50347-5. DOI: 10.1007/978-3-030-50347-5_11.
(Visited on 14/11/2023).

[46] Olav Andre Nergård Rongved, Markus Stige, Steven Alexander Hicks, Vajira Lasantha Thambawita, Cise Midoglu, Evi Zouganeli, Dag Johansen, Michael Alexander Riegler and Pål Halvorsen. 'Automated Event Detection and Classification in Soccer: The Potential of Using Multiple Modalities'. In: *Machine Learning and Knowledge Extraction* 3.4 (4 Dec. 2021), pp. 1030–1054.

ISSN: 2504-4990. DOI: 10.3390/make3040051.
URL: https://www.mdpi.com/2504-4990/3/4/51 (visited on 13/03/2024).

[47] Olav A. Norgard Rongved, Steven A. Hicks, Vajira Thambawita, Hakon K. Stensland, Evi Zouganeli, Dag Johansen, Michael A. Riegler and Pal Halvorsen. 'Real-Time Detection of Events in Soccer Videos Using 3D Convolutional Neural Networks'. In: *2020 IEEE International Symposium on Multimedia (ISM).* 2020 IEEE International Symposium on Multimedia (ISM). Naples, Italy: IEEE, Dec. 2020, pp. 135–144. ISBN: 978-1-72818-697-9. DOI: 10.1109/ISM.2020.00030.
URL: https://ieeexplore.ieee.org/document/9327961/ (visited on 04/03/2024).

[48] *One Month On: 5 Billion Engaged with the FIFA World Cup Qatar 2022™.*
URL: https://www.fifa.com/tournaments/mens/worldcup/qatar2022/news/origin1904-p.cxm.fifa.com/one-month-on-5-billion-engaged-with-the-fifa-world-cup-qatar-2022-tm (visited on 14/03/2024).

[49] *Opta Data from Stats Perform.* Stats Perform.
URL: https://www.statsperform.com/opta/ (visited on 12/03/2024).

[50] *Opta Event Definitions.* Stats Perform.
URL: https://www.statsperform.com/opta-event-definitions/ (visited on 12/03/2024).

[51] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jegou, Julien Mairal, Patrick Labatut, Armand Joulin and Piotr Bojanowski. *DINOv2: Learning Robust Visual Features without Supervision.* 2nd Feb. 2024. arXiv: 2304.07193 [cs].
URL: http://arxiv.org/abs/2304.07193 (visited on 16/02/2024). preprint.

[52] Luca Pappalardo, Paolo Cintia, Alessio Rossi, Emanuele Massucco, Paolo Ferragina, Dino Pedreschi and Fosca Giannotti. 'A Public Data Set of Spatio-Temporal Match Events in Soccer Competitions'. In: *Scientific Data* 6.1 (1 28th Oct. 2019), p. 236. ISSN: 2052-4463. DOI: 10.1038/s41597-019-0247-7.
URL: https://www.nature.com/articles/s41597-019-0247-7 (visited on 09/10/2023).

[53] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai and Soumith Chintala.

‘PyTorch: An Imperative Style, High-Performance Deep Learning Library’. In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (visited on 27/03/2024).

[54] *Premium | Grammarly.* URL: https://www.grammarly.com/premium (visited on 14/05/2024).

[55] PricewaterhouseCoopers. *2024 AI Business Predictions.* PwC. URL: https://www.pwc.com/us/en/tech-effect/ai-analytics/ai-predictions.html (visited on 14/03/2024).

[56] Prajit Ramachandran, Barret Zoph and Quoc V Le. ‘Searching for activation functions’. In: *arXiv preprint arXiv:1710.05941* (2017). (Visited on 22/02/2024).

[57] Behnood Rasti, Danfeng Hong, Renlong Hang, Pedram Ghamisi, Xudong Kang, Jocelyn Chanussot and Jon Atli Benediktsson. ‘Feature Extraction for Hyperspectral Imagery: The Evolution from Shallow to Deep (Overview and Toolbox)’. In: *IEEE Geoscience and Remote Sensing Magazine* 8.4 (Dec. 2020), pp. 60–88. ISSN: 2168-6831, 2473-2397, 2373-7468. DOI: 10.1109/MGRS.2020.2979764. arXiv: 2003.02822 [cs, eess]. URL: http://arxiv.org/abs/2003.02822 (visited on 15/03/2024).

[58] *Rate of Growth for Clips and Highlights to Outstrip Live - Five-Year Projections.* Two Circles (GB). 18th Oct. 2019. URL: https://twocircles.com/gb-en/clips-highlights-value-growing-faster-live/ (visited on 14/03/2024).

[59] Olav A. Nergård Rongved, Steven A. Hicks, Vajira Thambawita, Håkon K. Stensland, Evi Zouganeli, Dag Johansen, Cise Midoglu, Michael A. Riegler and Pål Halvorsen. ‘Using 3D Convolutional Neural Networks for Real-time Detection of Soccer Events’. In: *International Journal of Semantic Computing* 15.02 (June 2021), pp. 161–187. ISSN: 1793-351X, 1793-7108. DOI: 10.1142/S1793351X2140002X. URL: https://www.worldscientific.com/doi/abs/10.1142/S1793351X2140002X (visited on 04/03/2024).

[60] Sebastian Ruder. *An Overview of Gradient Descent Optimization Algorithms.* 15th June 2017. arXiv: 1609.04747 [cs]. URL: http://arxiv.org/abs/1609.04747 (visited on 17/03/2024). preprint.

[61] David E Rumelhart, Geoffrey E Hintont and Ronald J Williams. ‘Learning Representations by Back-Propagating Errors’. In: (1986). (Visited on 17/03/2024).

[62] Mehdi Houshmand Sarkhoosh, Sayed Mohammad Majidi Dorcheh, Sushant Gautam, Cise Midoglu, Saeed Shafiee Sabet and Pål Halvorsen. *Soccer on Social Media*. 18th Oct. 2023. arXiv: 2310.12328 [cs]. URL: http://arxiv.org/abs/2310.12328 (visited on 18/03/2024). preprint.

[63] Mehdi Houshmand Sarkhoosh, Sayed Mohammad Majidi Dorcheh, Cise Midoglu, Saeed Shafiee Sabet, Tomas Kupka, Dag Johansen, Michael A. Riegler and Pål Halvorsen. 'AI-Based Cropping of Soccer Videos for Different Social Media Representations'. In: *MultiMedia Modeling*. Ed. by Stevan Rudinac, Alan Hanjalic, Cynthia Liem, Marcel Worring, Björn Þór Jónsson, Bei Liu and Yoko Yamakata. Cham: Springer Nature Switzerland, 2024, pp. 279–287. ISBN: 978-3-031-53302-0. DOI: 10.1007/978-3-031-53302-0_22. (Visited on 15/03/2024).

[64] *Sn-Spotting/Features/ExtractResNET_TF2.Py at 9842826f94e1419580a9d17219c11aca7225f7ce · SoccerNet/Sn-Spotting*. GitHub. URL: https://github.com/SoccerNet/sn-spotting/blob/9842826f94e1419580a9d17219c11aca7225f7ce/Features/ExtractResNET_TF2.py (visited on 19/04/2024).

[65] *Sn-Spotting/Features/VideoFeatureExtractor.Py at 9842826f94e1419580a9d17219c11aca7225f7ce · SoccerNet/Sn-Spotting*. GitHub. URL: https://github.com/SoccerNet/sn-spotting/blob/9842826f94e1419580a9d17219c11aca7225f7ce/Features/VideoFeatureExtractor.py (visited on 19/04/2024).

[66] *Sport*. In: *Wikipedia*. 3rd Mar. 2024. URL: https://en.wikipedia.org/w/index.php?title=Sport&oldid=1211530574 (visited on 10/12/2024).

[67] W. S. C. Sports. *How Changing Consumption Habits Are Fueling the Rise in Value of Sports Highlights*. WSC SPORTS. 5th July 2021. URL: https://wsc-sports.com/blog/editors-picks/how-changing-consumption-habits-are-fueling-the-rise-in-value-of-sports-highlights/ (visited on 14/03/2024).

[68] *Stats Perform Extends Long-Term Official Statistics Partnership for Premier League, EFL and SPFL*. Stats Perform. URL: https://www.statsperform.com/press/stats-perform-extends-long-term-official-statistics-partnership-for-premier-league-efl-and-spfl/ (visited on 12/03/2024).

[69] *Supervisely: Unified OS for Computer Vision*. URL: https://supervisely.com/ (visited on 04/04/2024).

[70] Selim Furkan Tekin, Arda Fazla and Suleyman Serdar Kozat. *Numerical Weather Forecasting Using Convolutional-LSTM with Attention and Context Matcher Mechanisms*. 4th Oct. 2023. arXiv: 2102.00696 [cs]. URL: http://arxiv.org/abs/2102.00696 (visited on 15/03/2024). preprint.

[71] T Thamaraimanalan, D Naveena, M Ramya and M Madhubala. 'Prediction and Classification of Fouls in Soccer Game Using Deep Learning'. In: (2020). (Visited on 06/05/2024).

[72] *THE IFAB - About The Laws | IFAB*. URL: https://www.theifab.com/laws/2022-23/about-the-laws/ (visited on 12/03/2024).

[73] *Top 10 Most Popular Sports In The World [Ranked]*. GeeksforGeeks. 21st Feb. 2024. URL: https://www.geeksforgeeks.org/most-popular-sports-in-the-world/ (visited on 14/03/2024).

[74] Joakim O. Valand, Haris Kadragic, Steven A. Hicks, Vajira Thambawita, Cise Midoglu, Tomas Kupka, Dag Johansen, Michael A. Riegler and Pål Halvorsen. 'Automated Clipping of Soccer Events using Machine Learning'. In: *2021 IEEE International Symposium on Multimedia (ISM)*. 2021, pp. 210–214. DOI: 10.1109/ISM52913.2021.00042. (Visited on 15/03/2024).

[75] Joakim Olav Valand, Haris Kadragic, Steven Alexander Hicks, Vajira Lasantha Thambawita, Cise Midoglu, Tomas Kupka, Dag Johansen, Michael Alexander Riegler and Pål Halvorsen. 'AI-Based Video Clipping of Soccer Events'. In: *Machine Learning and Knowledge Extraction* 3.4 (4 Dec. 2021), pp. 990–1008. ISSN: 2504-4990. DOI: 10.3390/make3040049. URL: https://www.mdpi.com/2504-4990/3/4/49 (visited on 18/03/2024).

[76] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser and Illia Polosukhin. *Attention Is All You Need*. 1st Aug. 2023. arXiv: 1706.03762 [cs]. URL: http://arxiv.org/abs/1706.03762 (visited on 26/04/2024). preprint.

[77] *VGG Image Annotator*. URL: https://www.robots.ox.ac.uk/~vgg/software/via/via_demo.html (visited on 04/04/2024).

[78] Li Wan, Matthew Zeiler, Sixin Zhang, Yann LeCun and Rob Fergus. 'Regularization of Neural Networks Using DropConnect'. In: (). (Visited on 15/03/2024).

[79] Junqing Yu, Aiping Lei, Zikai Song, Tingting Wang, Hengyou Cai and Na Feng. 'Comprehensive Dataset of Broadcast Soccer Videos'. In: *2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR)*. 2018 IEEE Conference on Multimedia Information Processing and Retrieval (MIPR). Miami, FL: IEEE, Apr. 2018, pp. 418–423. ISBN: 978-1-5386-1857-8. DOI: 10.1109/MIPR.2018.00090. URL: https://ieeexplore.ieee.org/document/8397046/ (visited on 14/03/2024).

[80] Xiaobo Zhang, Zhimin Li, Qian Zhang, Zegang Yin, Zhijie Lu and Yang Li. 'A New Weakly Supervised Deep Neural Network for Recognizing Alzheimer's Disease'. In: *Computers in Biology and Medicine* 163 (Sept. 2023), p. 107079. ISSN: 00104825. DOI: 10.1016/j.compbiomed.2023.107079. URL: https://linkinghub.elsevier.com/retrieve/pii/S0010482523005449 (visited on 15/03/2024).

[81] Hengshuang Zhao, Jiaya Jia and Vladlen Koltun. 'Exploring Self-Attention for Image Recognition'. In: *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). Seattle, WA, USA: IEEE, June 2020, pp. 10073–10082. ISBN: 978-1-72817-168-5. DOI: 10.1109/CVPR42600.2020.01009. URL: https://ieeexplore.ieee.org/document/9156532/ (visited on 26/04/2024).

Appendix A

Appendix for Chapter 3 - Published Papers

Through the creation of the TACDEC dataset created in Chapter 3, the desire to both share this research and make this dataset publicly available to other researchers was strong. This was made available through the ACM MMSys conference.

Title: TACDEC: Dataset of Tackle Events in Soccer Game Videos

Authors: Evan Jåsund Kassab, Håkon Maric Solberg, Sushant Gautam, Saeed Shafiee Sabet, Thomas Torjusen, Michael Riegler, Pål Halvorsen, Cise Midoglu

Published: ACM MMSys '24, April 15–18, 2024, Bari, Italy

DOI: 10.1145/3625468.3652166

TACDEC: Dataset of Tackle Events in Soccer Game Videos

Evan Jåsund Kassab
University of Oslo
Norway

Håkon Maric Solberg
University of Oslo
Norway

Sushant Gautam
SimulaMet & OsloMet
Norway

Saeed Shafiee Sabet
Forzasys
Norway

Thomas Torjusen
Norwegian Professional Football League
Norway

Michael Riegler
SimulaMet & OsloMet
Norway

Pål Halvorsen
SimulaMet, OsloMet & Forzasys
Norway

Cise Midoglu
SimulaMet & Forzasys
Norway

ABSTRACT

This paper introduces TACDEC, a dataset of tackle events in soccer game videos. Recognizing the gap in existing open datasets that predominantly focus on official soccer events such as goals and cards, TACDEC targets a comprehensive analysis of tackles — a critical aspect of soccer that combines technical skills, tactical decision-making, and physical engagement. By leveraging video data from the Norwegian Eliteserien league across multiple seasons, we annotated 425 videos with 4 types of tackle events, categorized into "tackle-live", "tackle-replay", "tackle-live-incomplete", and "tackle-replay-incomplete", yielding a total of 836 event annotations. The dataset offers an unprecedented resource for the development and testing of machine learning models aimed at understanding and analyzing soccer game dynamics. A proof-of-concept classification model demonstrates the dataset's utility, achieving promising results in automatic tackle detection, thereby validating TACDEC's potential to support not only advanced game analytics but also to enhance fan engagement and player development initiatives.

CCS CONCEPTS

• **Computing methodologies** → **Supervised learning by classification**; **Scene understanding**; **Activity recognition and understanding**; • **Information systems** → **Information retrieval**.

KEYWORDS

association football, video, event detection, classification

ACM Reference Format:
Evan Jåsund Kassab, Håkon Maric Solberg, Sushant Gautam, Saeed Shafiee Sabet, Thomas Torjusen, Michael Riegler, Pål Halvorsen, and Cise Midoglu. 2024. TACDEC: Dataset of Tackle Events in Soccer Game Videos. In *ACM Multimedia Systems Conference 2024 (MMSys '24), April 15–18, 2024, Bari, Italy*. ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3625468.3652166

1 INTRODUCTION

Soccer is one of the most popular sports in the world, attracting millions of fans and participants from various cultures and communities. Its global appeal and the significant commercial interests it generates have spurred extensive research and innovation in areas such as soccer analytics, player performance tracking, and fan engagement strategies. Recent studies, such as Pu et al. [36], highlight the evolving nature and potential of game analysis techniques, emphasizing the importance of data-driven decision-making in modern soccer.

Automatic content production for soccer has seen significant advances, driven by the integration of artificial intelligence and machine learning[10–13]. This evolution is reflected in the growing sophistication of algorithms capable of not only analyzing player and ball movements but also generating real-time statistics and predictive insights. Works such as [19, 27, 38] demonstrate how data-based analysis such as deep learning models have been effectively used to automate content production, offering faster and more accurate analyses than traditional methods. These innovations are transforming how soccer is broadcast, analyzed, and experienced by audiences worldwide.

Detecting events within video streams presents a significant challenge, given the diversity of methodologies developed for various applications. Specifically, in the sports domain, the conventional method of manually annotating video segments, where a team tags portions of the content in real-time during live broadcasts, is both time-consuming and costly. For soccer, this process involves individuals closely watching the game to identify and mark significant moments like goals and fouls, leading to delays in the availability of annotated content. To counter these limitations, there has been a surge in the development of automated systems over recent years, with several notable contributions making strides in this area [4, 14, 21, 25, 31, 34, 37, 39, 41, 42]. Furthermore, the SoccerNet challenge has featured action spotting as a key task [8], where recent competitions have seen the top-performing teams achieve an average mean Average Precision (mAP) of approximately 80% in 2023, demonstrating the progress in this field.[1]

Machine learning applications require well-curated datasets for training, which is particularly challenging in the context of sports

[1]https://www.soccer-net.org/tasks/action-spotting

Figure 1: Sample tackles.

video analysis. The quality and granularity of the data significantly influence the model performance. For soccer, datasets must accurately capture intricate details of the game, such as player movements, ball position, and game events. This demands high-resolution video footage as well as precise temporal and spatial annotations. Existing datasets in this domain, like SoccerNet, provide extensive annotations covering a wide range of events, but there is still a gap in terms of real-time processing and multi-dimensional analysis. The integration of advanced techniques in computer vision and deep learning promises to bridge this gap, offering the potential to revolutionize how sports analytics are performed, leading to more nuanced and real-time insights into games.

In this paper, we introduce TACDEC, a dataset designed to enable automatic tackle event detection in soccer games by leveraging video data from the Eliteserien league. TACDEC incorporates various class annotations based on a comprehensive understanding of a tackle and aims to fill the gap in soccer analytics and research concerning the technical, tactical, and physical aspects of tackles. Overall, our contributions are as follows:

- Creation of the TACDEC dataset, a novel resource for detailed tackle event analysis in soccer.
- Provision of a baseline classification model that validates the dataset's effectiveness for machine learning applications.
- Identification of potential enhancements to support advanced game analytics, fan engagement, and player development through open access to the dataset and model framework.

2 RELATED WORK

Several datasets already exist in the context of soccer and data analysis. For example, SoccerNet-v3 [5] is a comprehensive dataset that supports tasks like camera calibration, pitch localization, and player re-identification, with a focus on video data from soccer games. Another notable dataset, SoccerDB [20], offers a large-scale resource for video understanding, incorporating a substantial collection of images and extensive video footage of soccer games. These datasets serve as valuable resources for analyzing various aspects of soccer games, including player tracking and event spotting.

Despite the availability of soccer datasets, a gap exists in analyzing tackles. Current datasets mainly cover goals, cards, and set pieces, directly affecting game outcomes. Yet, they overlook tactical actions and specific player behaviors like tackling.

In other sports, tackling has been more in focus, and research in that direction shows the advantages of specifically focusing on tackling. In 2022, Nonaka et al. wanted to reduce the risk of severe injuries in sports where contact occurs; more specifically, they implemented a tackle detection system for rugby [32]. Using rugby games from the Japan Rugby Top League, deep learning was used together with pose estimation to detect potential concussion-leading tackles. A total of 293 tackles were extracted, with the subcategories "low-risk tackles" (238) and "high-risk tackles" (155). Some years earlier, in 2014, Gastin et al. explored how the use of wearable microsensors could help detect tackles in Australian football [9]. The detection accuracy for tackles was observed to be 78%. This accuracy was derived from two scenarios: a 66% correct detection rate when identifying the player executing the tackle and a 90% correct detection rate when identifying the player being tackled. Observations from actual videos were used as the criterion.

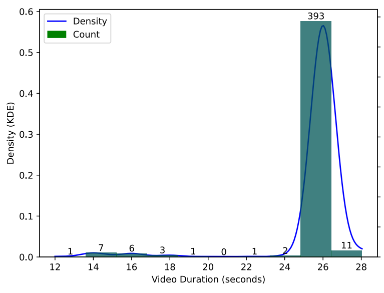
Back to computer vision tasks, Morra et al. [28] developed a two-level system that detects atomic and complex events using soccer data. With this, they used a synthetic dataset generated from a soccer game simulator, which included events such as goals, corners, and penalty kicks, but also events such as fouls. Although there were around 60,000 tackle events generated, simulated data might not capture the full complexity of real-game tackles.

Most of the work in the context of tackling is related to heavy-impact sports. Studies such as those by Klein et al. [24] show that tackling also contributes significantly to injuries in soccer players and that specific analysis and datasets can help reduce the burden on players and gain new insights.

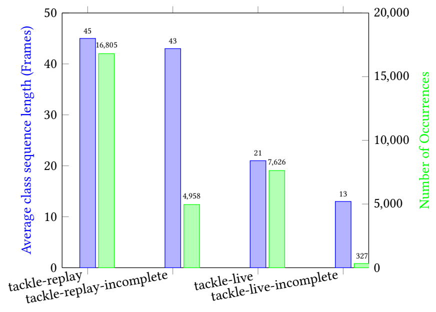
Recognition of a gap in existing soccer datasets, particularly in their lack of coverage of tactical and technical elements such as tackling, shows the need for our specialized tackle dataset. This gap is further highlighted by SoccerNet's incorporation of multi-view foul recognition in their annual challenges, demonstrating a growing interest in this research area [2]. Our dataset is not just an addition to the existing array of soccer data, but it represents a targeted approach with distinct advantages:

- Tackles are a key indicator of defensive abilities. By analyzing tackles, as depicted in Figure 1, we can obtain granular data on defensive actions, which allows for a more nuanced evaluation of player performances.
- Tackles, often dynamic and thrilling, significantly contribute to spectator experience. Highlighting these moments can enhance fan engagement, especially in digital media.
- Data on tackles can be instrumental in coaching and training, helping players refine their defensive skills with targeted feedback.
- Understanding the mechanics and context of tackles that lead to injuries can guide preventive measures, both in training and during games.
- Data on tackles can inform referees and soccer governing bodies about high-risk scenarios, leading to improved safety rules and enforcement.

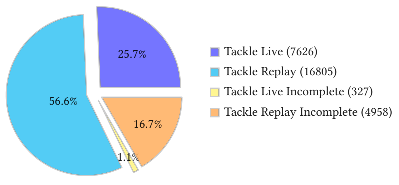
The focus on tackles in soccer and the creation of a specific dataset not only addresses a significant gap in existing soccer analysis methods but also offers a comprehensive view of the game's dynamics. It enriches tactical and performance analysis, enhances fan experiences, and supports player development initiatives.



(a) Histogram of video duration.



(b) Event occurence and duration.



(c) Distribution of event classes.

Figure 2: Dataset statistics.

3 DATASET CURATION

3.1 Sources and Data Collection

TACDEC was curated by capturing and annotating soccer videos from the Norwegian Eliteserien league. In partnership with the Norwegian Professional Football League (Norsk Toppfotball), we used APIs from their partner company Forzasys[2] and retrieved video clips with yellow card events, which present a higher probability of seeing tackles. Clips were extracted from 70 games in 2021, and 176 games in 2022, giving a total of 246 unique games. All yellow card events for 6 of the total 16 teams (Odd, Rosenborg, Sarpsborg, Brann, Viking, and Vålerenga) from the 2021 season and all yellow card events from the 2022 season were extracted. This returned a total of 1015 clips, which were inspected manually to determine whether a tackle occurred, assessed the presence of audio, and excluded clips where distinguishing between pushing/pulling and tackles was challenging. The total number of clips was then reduced to the current dataset size of 425. These 425 clips include a total of 836 annotated events, covering all teams involved in each season. Due to relegation, some teams only have entries in one season. Figure 2 depicts dataset statistics, including a histogram of video durations and the distribution of event classes by occurrence and duration.

3.2 Annotation

To label our dataset, we have defined a tackling sequence with start- and end-frames as follows:

Start frame: The moment the defender redistributes their weight onto the leg that is used to launch into the tackle, initiating the movement.

End frame: The tackle is considered complete when one of the two following conditions are met:

(1) The forward motion of the player tackling has stopped.
(2) If airborne, the tackled player makes contact with the ground.

Tackle classes (and corresponding class labels) are defined as follows:

I **tackle-live:** A "real-time" tackle in a broadcast video. A specific tackle can only be a live tackle once, even though there could be multiple unique live tackles in one clip.

II **tackle-replay:** When a tackle is replayed during the same clip. Could be in slow-motion or zoomed, and there could be multiple replays for each live tackle.

III **tackle-live-incomplete:** A live tackle, for which the start or end frame is not visible according to the definition of a tackle. This could be due to clipping artifacts or players being off-screen.

IV **tackle-replay-incomplete:** A tackle replay, for which the start or end frame is not visible according to the definition of a tackle. This could be due to clipping artifacts or players being off-screen.

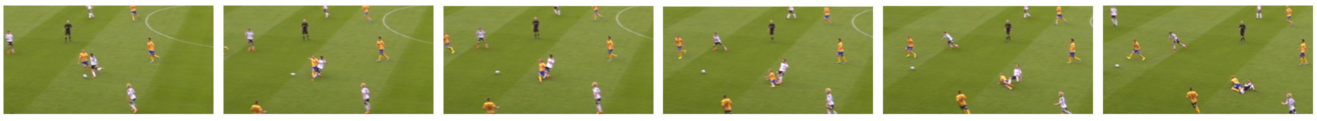
V **background:** For when none of the 4 other class labels are applicable.

LabelBox [1] was used to manually annotate frames due to its feature allowing for double review or verification of all annotations. Here, we iterated through the frames of each video and labeled them according to the classes above. Labels were then exported from LabelBox and post-processed in terms of cleaning and removing excess information. Overall, among the 425 labeled videos, each video contains at least one event, often multiple. Figure 3 visually illustrates two tackle events. Approximately 68% of the videos feature labeled replay tackles, while 60% have tackles occurring in real-time. 23% of the videos contain incomplete replay sequences, and 5% showcase incomplete live tackles. It is important to note that despite the presence of incomplete tackles within a video, multiple tackle sequences are typically observed.

3.3 Final Output

The curated dataset comprises a total of 425 labeled videos. The average length per clip is 0.43 minutes translating to 25.8 seconds. Table 1 presents the number of samples in the final dataset per team and season. To prevent each sample from being counted twice, we exclusively associated each sample with the home team. Each video in the dataset is accompanied by a separate JSON file (Listing 1) that contains detailed information about the labeled sequences within the video (e.g., class label, start and end time for each sequence). The TACDEC dataset can be found on Hugging Face (https://huggingface.co/datasets/SimulaMet-HOST/TACDEC).

[2]https://forzasys.com/


(a) (b) (c) (d) (e) (f)

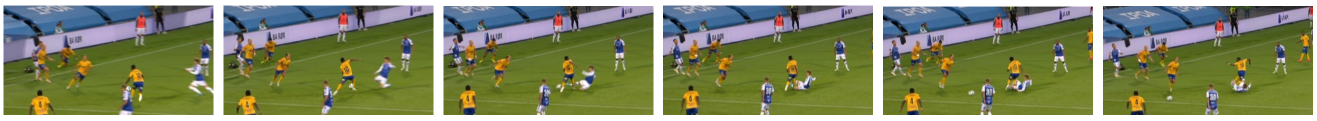

(g) (h) (i) (j) (k) (l)

Figure 3: Sample frames from a `tackle-live` event with a side block tackle (top), and sample frames from a `tackle-replay` event with a sliding tackle (bottom). Starting frames in (a) and (g), ending frames in (f) and (l).

Team	Season	tackle-live	tackle-replay	tackle-live-incomplete	tackle-replay-incomplete	Total
Aalesund	2021	-	-	-	-	-
	2022	10	18	1	5	34
Bodø/Glimt	2021	3	2	1	2	8
	2022	20	16	-	4	40
Brann	2021	8	2	1	2	13
	2022	-	-	-	-	-
HamKam	2021	-	-	-	-	-
	2022	22	29	2	13	66
Haugesund	2021	2	2	-	-	4
	2022	35	29	2	5	71
Jerv	2021	-	-	-	-	-
	2022	28	16	2	6	52
Mjøndalen	2021	1	-	-	-	1
	2022	-	-	-	-	-
Kristiansund	2021	1	2	-	2	5
	2022	9	20	-	1	30
Lillestrøm	2021	6	8	-	3	17
	2022	20	12	1	10	43
Molde	2021	1	1	-	-	2
	2022	19	17	-	2	38
Odd	2021	4	10	-	2	16
	2022	20	18	1	11	50
Rosenborg	2021	1	9	-	2	12
	2022	11	13	2	7	33
Sandefjord	2021	2	1	-	1	4
	2022	18	14	2	7	41
Sarpsborg 08	2021	9	10	2	1	22
	2022	7	17	-	2	26
Stabæk	2021	6	4	1	-	11
	2022	-	-	-	-	-
Strømsgodset	2021	4	4	1	2	11
	2022	13	22	2	2	39
Tromsø	2021	4	4	-	3	11
	2022	10	14	0	5	29
Viking	2021	9	7	-	-	16
	2022	16	21	0	6	43
Vålerenga	2021	3	2	1	1	7
	2022	19	17	0	5	41
All	**All**	**341**	**361**	**22**	**112**	**836**

Table 1: Number of annotated events in the dataset, per home team and season.

Listing 1: Template for the JSON files in the dataset.

```
{
  "id": "1234_qwerty",
  "metadata": {
    "game-id": 1234,
    "date": 2023-02-01,
    "team-home": {"name": "TeamH, "id": 12}
    "team-away": {"name": "TeamA", "id": 34}
    "clip-id": "qwerty"},
  "media_attributes": {
    "height": 720,
    "width": 1280,
    "mime_type": "video/mp4",
    "frame_rate": 25,
    "frame_count": 650},
  "events": [
      {"start_frame": 200,
       "end_frame": 300,
       "type": "tackle_live"},
      {"start_frame": 325,
       "end_frame": 450,
       "type": "tackle_live"},
      {"start_frame": 520,
       "end_frame": 600,
       "type": "tackle_replay"}]
}
```

4 DATASET USE CASE DEMONSTRATION

In this section, we aim to demonstrate the validity and utility of our dataset by applying it to a straightforward classification task. As a proof of concept, we focus on the automatic detection of tackle actions within video frames. While recognizing that this application represents a relatively narrow scope of the potential of the dataset, it serves as a practical entry point due to the comparative ease of evaluating classification tasks. Through this methodological approach, we aim to validate the robustness of the dataset and establish a foundation for its application in more complex and varied research domains.

The initial stage of our pipeline involves the pre-processing of video frames, such as resizing and transforming, to ensure a standardized format and dimensionality across all inputs. This is crucial for maintaining consistency when feeding data into the feature extraction model.

4.1 Feature Extraction

Following pre-processing, each video is processed on a frame-by-frame basis using the DINOv2 [35] model to extract a set of rich, descriptive features. This model is adept at distilling significant visual information from individual frames, producing a feature vector for each frame. To accommodate videos of varying lengths, a fixed-length feature matrix is constructed for each video. This matrix is populated with the extracted features, and padding frames are utilized as necessary to standardize the matrix size across all videos, ensuring that each video, regardless of its original length, is represented by a feature matrix of uniform dimensions.

Using the DINOv2 model, we extract characteristics from video frames, where each frame is individually processed to produce a characteristic vector of dimensions [257,1024]. The vector has a 1024 representation for 256 regions within each frame, as well as a CLS token to it. The extraction process capitalizes on the model's capacity to distill comprehensive visual representations, where we specifically retain the output of the last layers to harness rich features conducive to various research domains. This approach facilitates a robust foundation for feature representation, accommodating a broad spectrum of downstream tasks beyond our primary focus. The version used during the feature extraction for this work is the HuggingFace implementation of Facebook's DINOv2-large, with around 302M parameters[3].

4.2 Multi-Class Classification

Once the feature extraction phase is complete, we construct a frame-classification dataset comprising 35,609 (13.10%) randomly sampled frame instances across 3 distinct classes ("background", "tackle-live", "tackle-replay"), leaving 236,293 of 271,902 frames out. For the classification, we utilize only the initial token (CLS) in the feature sequence, excluding 256 features from the total of 257 available. It is also worth noting that instances of the "incomplete" classes were substituted for their respective others as a frame-by-frame classification would not be able to learn the temporal features, hence distinguishing between a frame present in a tackle could be problematic. The division of the dataset into training and testing subsets adheres to an 80:20 ratio, with an allocation strategy that ensures that frames from an individual clip are exclusively assigned to either subset. This partitioning scheme is pivotal in preserving the integrity of the evaluation process, preventing informational leakage between the training and testing phases, and ensuring a realistic assessment of the model's performance on unseen data.

The resulting subsets are used to train and evaluate a multi-layered neural network designed for classification characterized by a minimalist architecture, incorporating dropout mechanisms and L1 regularization [40] as a measure against over-fitting, reinforcing the model's generalization capability. The architecture of this neural network is chosen to optimize performance on the classification task, incorporating three linear layers and ReLU activation functions [30] suited to handling the complexity and variability of the input features. This network uses the extracted features to make per-frame predictions, identifying specific actions or characteristics defined by the classification task.

[3]https://huggingface.co/facebook/dinov2-large

Event	precision	recall	f1-score	support
Background	0.93	0.70	0.80	7500
Tackle-live	0.52	0.90	0.66	1627
Tackle-replay	0.84	0.94	0.88	5068
Macro avg	0.76	0.85	0.78	14195
Weighted avg	0.85	0.81	0.81	14195
Accuracy			**0.81**	14195
Accuracy (Dummy Classifier)			0.35	14195

Table 2: Evaluation metrics for the classifier and comparison with a dummy classifier.

To ensure versatility for other potential downstream applications, the full DINOv2 features for all 271,902 frames in the dataset are made available as part of the public dataset, along with the dataset generation code, feature extraction code, the full specification of the baseline classification architecture, training process, and the model weights, enhancing transparency and facilitating reproducibility[4].

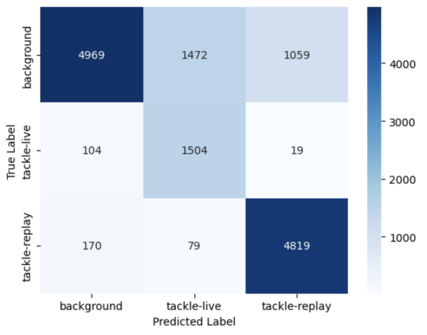
4.3 Evaluation

Performance Evaluation: The classifiers are evaluated using a range of standard metrics and visualization methods to ensure a comprehensive assessment of their performance [15]. These metrics include *accuracy* which measures the overall correctness of the classifier; *precision* which assesses the classifier's ability to avoid false positives; *recall* which evaluates the classifier's capability to find all relevant instances; *F1 Score* which provides a balance between precision and recall; *Confusion Matrix* which visualizes the classifier performance in different categories; *Area Under the Receiver Operating Characteristic curve (AUC-ROC)* which reflects the trade-off between sensitivity and specificity [16]; and *Precision-Recall (PR) curve* which shows the the relationship between precision and recall for various threshold settings. Each metric offers insight into different aspects of classifier performance, allowing a holistic evaluation [7]. The following section outlines the rigorous evaluation of the classifier. The classifier was evaluated based on various metrics that are essential to understanding its strengths and weaknesses in the classification of soccer events, as shown in Table 2. The trained classification model outperformed the stratified dummy classifier baseline with random guessing by a significant margin.

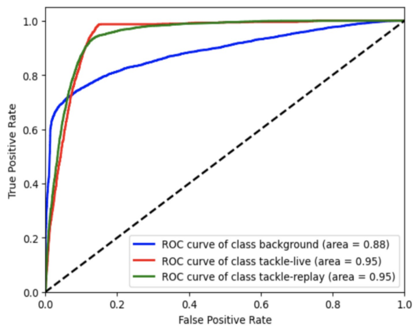
Overall Performance: The classifiers showed varied effectiveness in different types of events. As seen in Table 2, the lowest F1 score was observed for the "tackle-live" of (0.66), with a relatively low precision of 0.52, even though 90% is captured. For "background", we see a high precision, indicating that it predicts the background correctly 93% of the time, with an F1 score reflecting a balanced trade-off between precision and recall. The "tackle-replay" class has an F1-score of 0.88, which is the overall highest, shining back on great precision and recall with 0.84 and 0.94, respectively.

Overall Accuracy: The classifiers achieved an overall accuracy of 0.81, indicating that the classifier with a more complex model and more fine-tuning, can achieve good results.

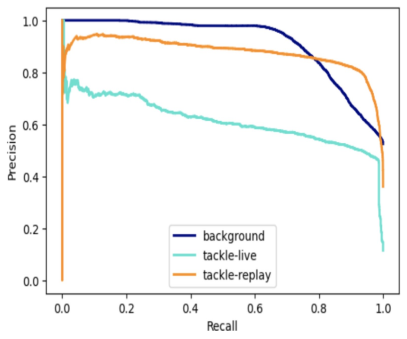
[4]https://huggingface.co/datasets/SimulaMet-HOST/TACDEC



(a) Confusion matrix.



(b) ROC curve.



(c) PR curve.

Figure 4: (a) Confusion matrix showing the performance of the classifier in predicting action events from the soccer game. (b) Multi-class ROC curve for all event types. (c) Multi-class PR curve for all event types.

Confusion Matrix: The Confusion Matrix seen in Figure 4a shows that the background has the highest number of false negatives, indicating that it can be difficult to differentiate between the background and the tackle events. As for "tackle-live", we see a lower number of false positives and false negatives compared to "background" suggesting the model distinguishes better here than for "background". "tackle-replay" shows the best performance with the high true positives and far less misclassification, meaning the classifiers is most effective at identifying "tackle-replay". These results might be due to "background" and "tackle-live" sharing similar camera settings, whereas "tackle-replay" often features closer zooms.

ROC Curve: A multi-class ROC curve was plotted to analyze performance across various classes and is shown in Figure 4b. The figure shows that the (blue) line, denoting "background", is the most challenging one to distinguish from the rest, while for the other two classes, we move closer toward the top left corner.

Precision-Recall (PR) Curve: A multi-class PR curve was plotted to analyze the performance between various classes and is shown in Figure 4c. The precision for "background" (blue) remains high until a sharp decline at high recall, indicating stable precision until it struggles with positive instances. "tackle-live" (cyan) shows lower and inconsistent precision, dropping with increased recall. "tackle-replay" (orange) maintains high precision across recall levels, highlighting the classifier's effectiveness in distinguishing this class.

4.4 Discussion

The TACDEC dataset, centered on tackle events in soccer, heralds numerous possibilities for research and practical applications in various domains [3, 22, 23, 26, 29, 33]. It serves as a key resource for advanced sports analytics, enabling a deeper understanding of defensive strategies and player dynamics, which are crucial for player performance evaluation and team tactics [18]. The dataset on tackle events facilitates injury prevention research by identifying patterns linked to player injuries, informing safer training practices and potential rule changes [3, 26, 33]. For the computer vision and machine learning community, TACDEC provides a rich ground for developing and testing algorithms aimed at tackling recognition and enhancing automated sports analysis technologies. Media companies can leverage tackle recognition models to create engaging fan content, while coaches and players could benefit from it as an educational tool to improve defensive skills and game understanding [6, 29]. Additionally, it offers insights for social and psychological studies on player behavior and decision-making [17, 29]. Lastly, the dataset could assist soccer governing bodies in refining rules and officiating standards, ultimately contributing to the sport's safety and integrity. Overall, the potential of TACDEC to impact a wide array of fields underscores its value as a significant contribution to soccer analytics and the broader sports technology landscape.

5 CONCLUSION

The development and evaluation of the TACDEC dataset represent a significant step forward in the domain of soccer analytics, specifically in the automatic detection of tackle events. By focusing on a previously under-explored aspect of soccer game analysis, this work fills a gap in sports research. Our proof-of-concept classification model, applied to tackle detection, underscores the dataset's practical value and offers an example of its broader applicability for various research purposes beyond the initial demonstration. As TACDEC becomes available to the research community, we anticipate that it will lead to a wide range of studies in computer vision, game understanding, and analytics, thus contributing to the enhancement of tactical analysis, player performance evaluation, and the overall experience of soccer fans. As future work, we aim to expand the dataset, incorporate additional event types, connect resulting injuries to tackle events, and explore more sophisticated models for event detection.

ACKNOWLEDGMENTS

This research was partly funded by the Research Council of Norway, project number 346671 (AI-storyteller). We would like to thank the Norwegian Professional Football League ("Norsk Toppfotball") for making videos available for the research.

REFERENCES

[1] 2024. Labelbox | Data-centric AI Platform for Building & Using AI. https://labelbox.com [Online; accessed 2. Feb. 2024].
[2] 2024. *SoccerNet - 2024*. https://www.soccer-net.org/challenges/2024

[3] Nicholas Burger, Michael I Lambert, Wayne Viljoen, James C Brown, Clint Readhead, and Sharief Hendricks. 2016. Tackle Technique and Tackle-Related Injuries in High-Level South African Rugby Union under-18 Players: Real-Match Video Analysis. 50, 15 (2016), 932–938. https://doi.org/10.1136/bjsports-2015-095295

[4] Anthony Cioppa, Adrien Deliège, Silvio Giancola, Bernard Ghanem, Marc Van Droogenbroeck, Rikke Gade, and Thomas B. Moeslund. 2020. *A Context-Aware Loss Function for Action Spotting in Soccer Videos.* IEEE Computer Society. https://doi.org/10.1109/CVPR42600.2020.01314

[5] Anthony Cioppa, Silvio Giancola, Adrien Deliège, Le Kang, Xin Zhou, Zhiyu Cheng, Bernard Ghanem, and Marc Van Droogenbroeck. 2022. SoccerNet-Tracking: Multiple Object Tracking Dataset and Benchmark in Soccer Videos. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).* IEEE, 19–20. https://doi.org/10.1109/CVPRW56347.2022.00393

[6] Joshua Coleclough. 2013. Soccer coaches' and referees' perceptions of tackle incidents with respect to the laws of the game. *International Journal of Performance Analysis in Sport* 13, 2 (2013), 553–566.

[7] Jesse Davis and Mark Goadrich. 2006. The relationship between Precision-Recall and ROC curves. In *ICML '06: Proc. of the 23rd international conference on Machine learning.* Association for Computing Machinery, New York, NY, USA, 233–240.

[8] Adrien Deliège, Anthony Cioppa, Silvio Giancola, Meisam J. Seikavandi, Jacob V. Dueholm, Kamal Nasrollahi, Bernard Ghanem, Thomas B. Moeslund, and Marc Van Droogenbroeck. 2021. SoccerNet-v2: A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).* IEEE, 19–25.

[9] Paul B. Gastin, Owen C. McLean, Ray V. P. Breed, and Michael Spittle. 2014. Tackle and impact detection in elite Australian football using wearable microsensor technology. *J. Sports Sci.* 32, 10 (2014), 947–953. https://doi.org/10.1080/02640414.2013.868920 arXiv:24499311

[10] Sushant Gautam. 2022. AI-based Soccer Game Summarization: From Video Highlights to Dynamic Text Summaries. https://doi.org/10.13140/RG.2.2.24627.91681 Master's Thesis, Tribhuvan University.

[11] Sushant Gautam. 2023. Bridging Multimedia Modalities: Enhanced Multimodal AI Understanding and Intelligent Agents. In *ICMI '23: Proc. 25th Int. Conf. on Multimodal Interaction.* ACM, New York, NY, USA, 695–699. https://doi.org/10.1145/3577190.3614225

[12] S. Gautam, C. Midoglu, S.S. Sabet, D.B. Kshatri, and P. Halvorsen. 2022. Assisting soccer game summarization via audio intensity analysis of game highlights. In *Proc. 12th IOE Graduate Conference.* Inst. of Eng. Tribhuvan University, Nepal, 25–32. https://conference.ioe.edu.np/publications/ioegc12/IOEGC-12-004-12009.pdf

[13] S. Gautam, C. Midoglu, S.S. Sabet, D.B. Kshatri, and P. Halvorsen. 2022. Soccer Game Summarization using Audio Commentary, Metadata, and Captions. In *NarSUM '22: Proc. 1st Workshop on Narrative Summarization.* ACM, New York, NY, USA, 13–22. https://doi.org/10.1145/3552463.3557019

[14] Silvio Giancola, Mohieddine Amine, Tarek Dghaily, and Bernard Ghanem. 2018. SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW).* IEEE, 18–22. https://doi.org/10.1109/CVPRW.2018.00223

[15] Margherita Grandini, Enrico Bagli, and Giorgio Visani. 2020. Metrics for Multi-Class Classification: an Overview. arXiv:2008.05756 [stat.ML]

[16] Jan Grau, Ivo Grosse, and Jens Keilwagen. 2015. PRROC: computing and visualizing precision-recall and receiver operating characteristic curves in R. *Bioinformatics* 31, 15 (Aug. 2015), 2595–2597.

[17] Sharief Hendricks, Steve den Hollander, Nicholas Tam, James Brown, and Michael Lambert. 2015. The relationships between rugby players' tackle training attitudes and behaviour and their match tackle attitudes and behaviour. *BMJ OPEN SP EX MED* 1, 1 (Dec. 2015), e000046. https://doi.org/10.1136/bmjsem-2015-000046

[18] Sharief Hendricks, Brad Roode, Bevan Matthews, and Michael Lambert. 2013. Defensive Strategies in Rugby Union. 117, 1 (2013), 65–87. https://doi.org/10.2466/30.25.PMS.117x17z6

[19] Yudong Jiang, Kaixu Cui, Leilei Chen, Canjin Wang, and Changliang Xu. 2020. Soccerdb: A large-scale database for comprehensive video understanding. In *Proc. of the 3rd International Workshop on Multimedia Content Analysis in Sports.* 1–8.

[20] Yudong Jiang, Kaixu Cui, Leilei Chen, Canjin Wang, and Changliang Xu. 2020. SoccerDB: A Large-Scale Database for Comprehensive Video Understanding. In *MMSports '20: Proc. of the 3rd International Workshop on Multimedia Content Analysis in Sports.* Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3422844.3423051

[21] Andrej Karpathy, George Toderici, Sanketh Shetty, Thomas Leung, Rahul Sukthankar, and Li Fei-Fei. 2014. Large-Scale Video Classification with Convolutional Neural Networks. In *IEEE Conference on Computer Vision and Pattern Recognition.* IEEE, 23–28. https://doi.org/10.1109/CVPR.2014.223

[22] Hamish A Kerr, Eric H Ledet, Ashar Ata, Jennifer L Newitt, Matthew Santa Barbara, Milan Kahanda, and Erin Sperry Schlueter. 2018. Does Instructional Video Footage Improve Tackle Technique? 13, 1 (2018), 3–15.

[23] Doug King, Patria A Hume, and Trevor Clark. 2010. Video Analysis of Tackles in Professional Rugby League Matches by Player Position, Tackle Height and Tackle Location. (2010).

[24] Christian Klein, Patrick Luig, Thomas Henke, Hendrik Bloch, and Petra Platen. 2021. Nine typical injury patterns in German professional male football (soccer): a systematic visual video analysis of 345 match injuries. *British journal of sports medicine* 55, 7 (2021), 390–396.

[25] Tianwei Lin, Xiao Liu, Xin Li, Errui Ding, and Shilei Wen. 2019. *BMN: Boundary-Matching Network for Temporal Action Proposal Generation.* IEEE Computer Society. https://doi.org/10.1109/ICCV.2019.00399

[26] Zubair Martin, Sharief Hendricks, and Amir Patel. 2021. Automated Tackle Injury Risk Assessment in Contact-Based Sports - A Rugby Union Example. In *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (Nashville, TN, USA). 4589–4598.

[27] Cise Midoglu, Saeed Shafiee Sabet, Mehdi Houshmand Sarkhoosh, Mohammad Majidi, Sushant Gautam, Håkon Maric Solberg, Tomas Kupka, and Pål Halvorsen. 2024. AI-Based Sports Highlight Generation for Social Media. In *Proc. of the ACM Mile-High Video Conference (MHV).* 7–13. https://doi.org/10.1145/3638036.3640799

[28] Lia Morra, Francesco Manigrasso, Giuseppe Canto, Claudio Gianfrate, Enrico Guarino, and Fabrizio Lamberti. 2020. Slicing and Dicing Soccer: Automatic Detection of Complex Events from Spatio-Temporal Data. In *Image Analysis and Recognition.* Springer, Cham, Switzerland, 107–121. https://doi.org/10.1007/978-3-030-50347-5_11

[29] Paul H. Morris and David Lewis. 2010. Tackling Diving: The Perception of Deceptive Intentions in Association Football (Soccer). 34, 1 (2010), 1–13. https://doi.org/10.1007/s10919-009-0075-0

[30] Vinod Nair and Geoffrey E. Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *ICML'10: Proc. of the 27th International Conference on International Conference on Machine Learning.* Omnipress, Madison, CT, USA, 807–814.

[31] Olav Andre Nergård Rongved, Markus Stige, Steven Alexander Hicks, Vajira Lasantha Thambawita, Cise Midoglu, Evi Zouganeli, Dag Johansen, Michael Alexander Riegler, and Pål Halvorsen. 2021. Automated Event Detection and Classification in Soccer: The Potential of Using Multiple Modalities. *Machine Learning and Knowledge Extraction* 3, 4 (2021), 1030–1054. Issue 4. https://doi.org/10.3390/make3040051

[32] Monami Nishio, Naoki Nonaka, Ryo Fujihira, Hidetaka Murakami, Takuya Tajima, Mutsuo Yamada, Akira Maeda, and Jun Seita. 2022. Objective detection of high-risk tackle in rugby by combination of pose estimation and machine learning. In *JSAI International Symposium on Artificial Intelligence.* Springer, 215–228.

[33] Naoki Nonaka, Ryo Fujihira, Monami Nishio, Hidetaka Murakami, Takuya Tajima, Mutsuo Yamada, Akira Maeda, and Jun Seita. 2022. End-to-End High-Risk Tackle Detection System for Rugby. In *Proc. of IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (New Orleans, LA, USA). 3549–3558. https://doi.org/10.1109/CVPRW56347.2022.00399

[34] Olav A. Norgard Rongved, Steven A. Hicks, Vajira Thambawita, Hakon K. Stensland, Evi Zouganeli, Dag Johansen, Michael A. Riegler, and Pål Halvorsen. 2020. Real-Time Detection of Events in Soccer Videos Using 3D Convolutional Neural Networks. In *2020 IEEE International Symposium on Multimedia (ISM)* (Naples, Italy). IEEE, 135–144. https://doi.org/10.1109/ISM.2020.00030

[35] Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, et al. 2023. DINOv2: Learning Robust Visual Features without Supervision. *arXiv* (April 2023). https://doi.org/10.48550/arXiv.2304.07193 arXiv:2304.07193

[36] Zhiqiang Pu, Yi Pan, Shijie Wang, Boyin Liu, Min Chen, Hao Ma, and Yixiong Cui. 2024. Orientation and decision-making for soccer based on sports analytics and AI: A systematic review. *IEEE/CAA Journal of Automatica Sinica* 11, 1 (2024), 37–57.

[37] Olav A. Nergård Rongved, Steven A. Hicks, Vajira Thambawita, Håkon K. Stensland, Evi Zouganeli, Dag Johansen, Cise Midoglu, Michael A. Riegler, and Pål Halvorsen. 2021. Using 3D Convolutional Neural Networks for Real-time Detection of Soccer Events. *International Journal of Semantic Computing* 15, 02 (2021), 161–187. https://doi.org/10.1142/S1793351X2140002X

[38] Melissa Sanabria, Frédéric Precioso, Pierre-Alexandre Mattei, and Thomas Menguy. 2022. A Multi-stage deep architecture for summary generation of soccer videos. *arXiv preprint arXiv:2205.00694* (2022).

[39] Du Tran, Heng Wang, Lorenzo Torresani, Jamie Ray, Yann LeCun, and Manohar Paluri. 2018. *A Closer Look at Spatiotemporal Convolutions for Action Recognition.* IEEE Computer Society. https://doi.org/10.1109/CVPR.2018.00675

[40] Diego Vidaurre, Concha Bielza, and Pedro Larrañaga. 2013. A Survey of L1 Regression. *Int. Stat. Rev.* 81, 3 (Dec. 2013), 361–387.

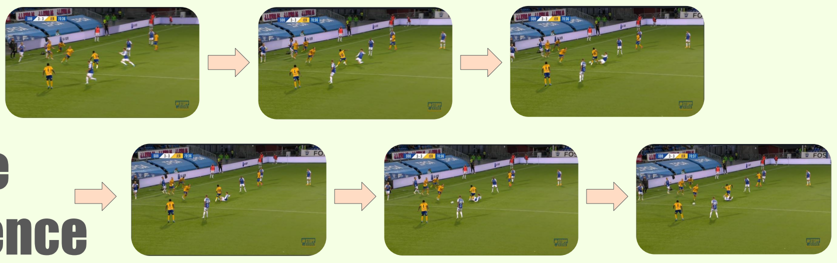
[41] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2016. *Temporal Segment Networks: Towards Good Practices for Deep Action Recognition.* Vol. 9912. Springer Verlag, Heidelberg, Germany.

[42] Shuangjie Xu, Yu Cheng, Kang Gu, Yang Yang, Shiyu Chang, and Pan Zhou. 2017. *Jointly Attentive Spatial-Temporal Pooling Networks for Video-Based Person Re-identification.* IEEE Computer Society. https://doi.org/10.1109/ICCV.2017.507

TACDEC: Dataset for Automatic Tackle Detection in Soccer Game Videos

Evan Jåsund Kassab
University of Oslo, Norway
evanjaasund@hotmail.com
UNIVERSITETET I OSLO
simulamet
Simula Metropolitan Center for Digital Engineering AS

A Tackle Sequence



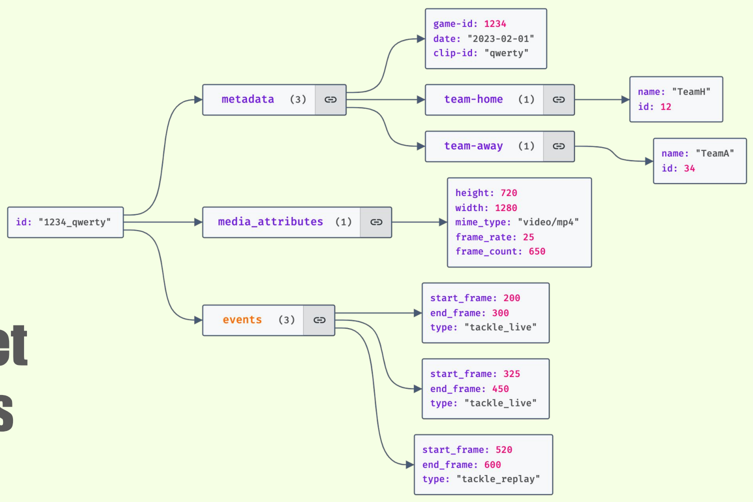
What is a Tackle?

The start frame is marked by the tackler's weight shift onto the initiating leg, indicating the tackle's start. The end frame occurs when either the tackler's forward motion ceases or, if the tackled player is airborne, when they contact the ground.

Abstract

This paper introduces TACDEC, a dataset focusing on tackle events in soccer videos, addressing the lack of data on such critical interactions in existing datasets. Utilizing Norwegian Eliteserien league footage, we annotated **425 videos**, capturing **836 tackle events** across four categories. TACDEC enables the development and evaluation of machine learning models for soccer video analysis.
An initial model tested on the dataset showed promising results in automatic tackle detection, underscoring TACDEC's value in advancing game analytics, fan engagement, and player development.

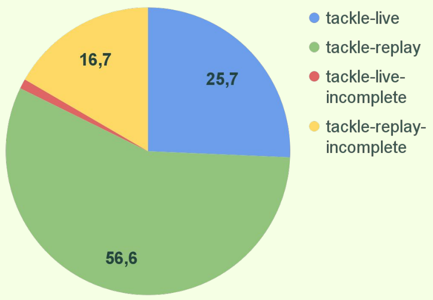
Dataset Labels



Clips per Team

Teams	SEASON	tackle-live	tackle-replay	tackle-replay-incomplete	tackle-live-incomplete	Total
Aalesund	2022	10	18	1	5	34
Bodø/Glimt	2021 2022	23	18	1	6	48
Brann	2021	8	2	1	2	13
HamKam	2022	22	29	2	13	66
Haugesund	2021 2022	37	31	2	5	75
Jerv	2022	28	16	2	6	52
Mjøndalen	2021	1	-	-	-	1
Kristiansund	2021 2022	10	22	-	3	35
Lillestrøm	2021 2022	26	20	1	13	60
Molde	2021 2022	20	18	-	2	40
Odd	2021 2022	24	28	1	13	66
Rosenborg	2021 2022	12	21	2	9	55
Sandefjord	2021 2022	20	15	2	8	45
Sarpsborg 08	2021 2022	16	27	2	3	48
Stabæk	2021	6	4	1	-	11
Strømsgodset	2021 2022	17	26	3	4	50
Tromsø	2021 2022	14	18	-	8	40
Viking	2021 2022	25	28	-	6	59
Vålerenga	2021 2022	22	19	1	6	48
All	All	341	361	22	112	836

Dataset Classes



A Simple Classifier

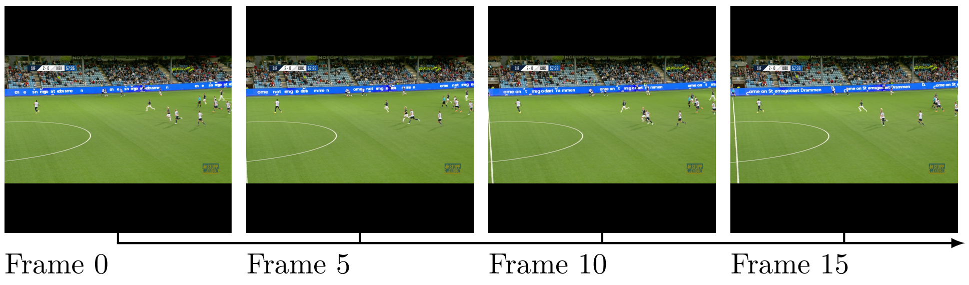
Event	precision	recall	f1-score	support
Background	0.93	0.70	0.80	7500
Tackle-live	0.52	0.90	0.66	1627
Tackle-replay	0.84	0.94	0.88	5068
Macro avg	0.76	0.85	0.78	14195
Weighted avg	0.85	0.81	0.81	14195
Accuracy			**0.81**	14295
Accuracy (dummy classifier)			0.35	14195

References

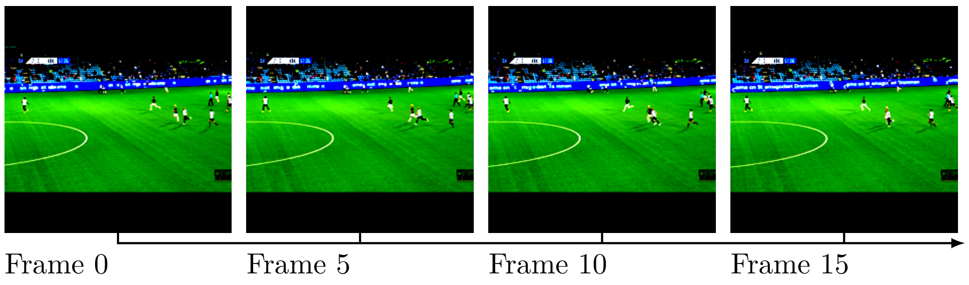
[1] 2024. Labelbox | Data-centric AI Platform for Building & Using AI. https://labelbox.com
[2] 2024. SoccerNet. https://www.soccer-net.org/challenges/2024
[3] Midoglu et al., *AI-Based Sports Highlight Generation for Social Media*.
ACM Mile-High Video Conference (MHV), 2024. https://doi.org/10.1145/3638036.3640799

Appendix B

Appendix for Chapter 7 - Discussion



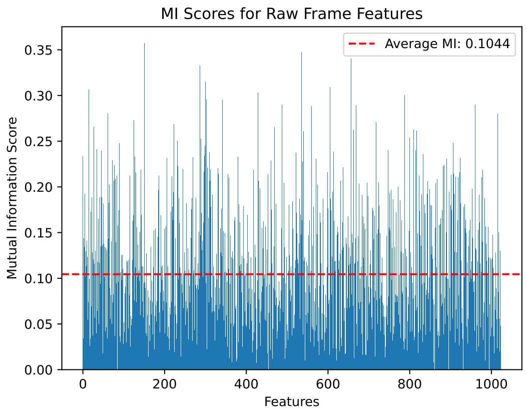
(a) Frames prior to processing.



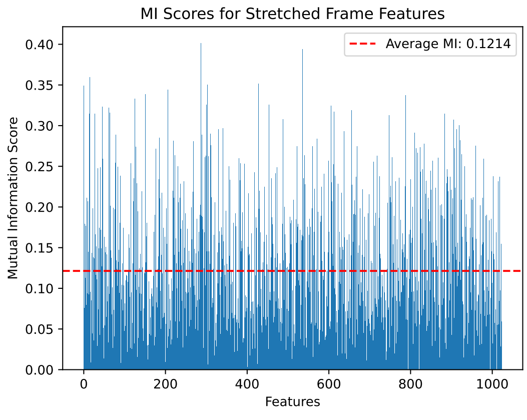
(b) Frames post processing (zero-padded).

Figure B.2: Comparison of 'zero-padded' frames before and after processing.

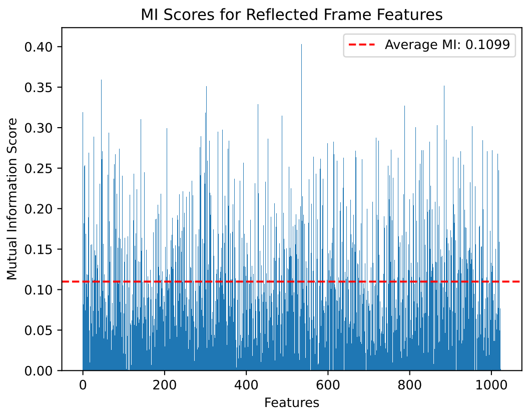
Figure B.1 displays the mutual information scores for 4 different variations of alterations done before feeding the frames to DINOv2's image processor. Mutual information score is a measurement of the mutual dependence between variables, most commonly used when there are two variables. Despite this, it can still be applied to high-dimensional data like the CLS-tokens utilized in this thesis. When applied in this context, it measures the mutual information score of each of the 1024 features against the target variables individually. This allows for the evaluation of how much information each feature contributes to predicting the target variable, aiding in the selection of the most informative features for model building. The higher the score, the stronger the dependency between the feature and the target.



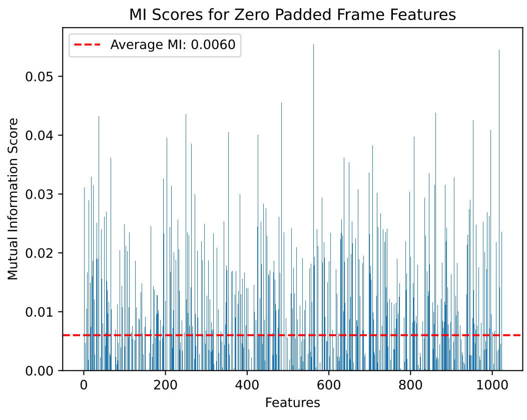
(a) Raw



(b) Stretched



(c) Reflected padding



(d) Zero padding

Figure B.1: Mutual Information Plots for the feature vector for the different alterations : (a) Raw, (b) Stretched, (c) Reflected padding, (d) Zero padding

Figure B.1a displays the mutual information scores of CLS-tokens from the images that were not altered prior to using the image processor.

After trying CLS-tokens for raw and stretched frames, a padded approach was tried. This approach involved padding black borders (zero-padding) on both the top and bottom of the image in order to create an 1:1 aspect ratio, see Figure B.2a. By doing so, the relationships between the pixels themselves remain intact, in contrast to being distorted when stretching the images. The results were less effective than anticipated, as it was impossible to pass on any learning to the models. Further examination revealed an average MI-score between features, as illustrated in Figure B.1d, indicating insufficient informational content in the features for effective class differentiation with only around $\frac{1}{20}$ of the scores compared to the average mutual information scores for that of the raw frames seen in Figure B.1a. Due to this, the strategy of using reflective padding was instead opted for. As we can from the average MI-score in Figure B.1c, it is higher than for the unaltered frames in B.1a. Also reflected in the results from both Chapter 5 and Chapter 6, an average higher MI-score did often lead to increased performance, with both alterations used in this thesis having a higher average MI-score than those not altered.