Machine learning-based approach for automated clipping of soccer events

Using scene boundary detection and logo detection

Joakim Olav Valand and Haris Kadragic

Thesis submitted for the degree of
Master in Programming and System Architecture
60 credits

Department of Informatics
Faculty of mathematics and natural sciences

UNIVERSITY OF OSLO

Spring 2021

Machine learning-based approach for automated clipping of soccer events

Using scene boundary detection and logo detection

Joakim Olav Valand and Haris Kadragic

© 2021 Joakim Olav Valand and Haris Kadragic

Machine learning-based approach for automated clipping of soccer events

http://www.duo.uio.no/

Printed: Reprosentralen, University of Oslo

Abstract

Today, highlights in soccer matches are manually annotated and clipped by human operators. This is a time-consuming, tedious, and expensive task. The clips are often a preset time interval instead of a tailored interval that fits the specific event. The editors might not even have time to clip it as it can often be important to distribute it as close to the live event itself. It could be edited later, but in many cases, this is too expensive. In this thesis, we experimented with automating the process of highlight generation using Scene boundary detection, logo detection, and a production-based algorithm. Through experimentation, we concluded that the VGG inspired CNN using grayscale input of 54×96 achieving a 100% F1-score was the best fit for our logo detection module on Eliteserien. For the more complex Premier League logo dataset, we concluded that the ResNet CNN using RGB input of 108×192 achieving an 0.997 F1-score was the best fit for our logo detection module. We trained and evaluated TransNet-V2 [64] on the SoccerNet shot boundary dataset, and compared the performance to the pre-trained version, and concluded that the pre-trained version was sufficient for the Scene boundary detection model of our system. Further, we combined these modules and implemented two different configurations of our system, one including full celebration scenes, and the other removing certain celebration scenes. We compared these to the already existing model in Eliteserien. Based on the qualitative and quantitative evaluation through a user study, we showed that Our model - Short and Our model - Full consistently produces more compelling highlight clips compared to the original model used in Eliteserien today. Upon inspection of the preferences of the participants we discovered that due to the random nature of the original model (using a set time interval for highlight extraction), it achieves low scores when it "misses", while in the cases where it "hits", the preference of model is more even. The results showed that this is a complicated task and there is a variety of which model is preferred impacted by several different factors such as background, real-world factors, mood, etc.

Acknowledgments

We would like to thank our supervisors, Pål Halvorsen, Steven Hicks, and Michael Riegler for their help. We would also like to thank Olav Rognved and Vajira Thambawita for help along the way. Finally, we would like to thank our families and friends for all their patience and support.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Non-linear TV and video clips on the internet are becoming an increasingly bigger part of our everyday life. Videos' incorporation with social media, smaller devices, cheap cellular data, and high-bandwidth internet at all times has made videos highly accessible and shareable. The competition for the users' attention is high with many video streaming services such as Amazon Prime, HBO, Netflix, Disney+, and video sharing platforms such as TikTok, Twitch, and YouTube. YouTube alone has over 1 billion hours of content watched daily, most of it through mobile screens [77].

Sports play a huge part in society today, both culturally and commercially. From 2016 to 2017, watch time for sports highlight videos grew by more than 80% on YouTube. In a survey of people who identified as sports fans, 80% said they used multiple devices to search for additional information such as player stats, live scores, and related videos [65]. Therefore, providing consumers with near to real-time replay options for use on a second device could be of big interest during a game. Previous clips related to the teams, league, or players are of interest too and should be available. To meet these demands, we want to make compelling clips of more events while providing a good technical standard and make them available fast. This way, online sports streaming providers and betting companies can provide good pregame content and live replay accessible on the fly.

Soccer is maybe the world's most popular sport, played by 250 million players in over 200 countries as reported by FIFA[25]. A combined 3.572 billion viewers – more than half of the global population aged four and over – tuned in to the 2018 FIFA World Cup, according to audience data for the official broadcast coverage[46]. FIFA also report 1.25 billion views on their content on YouTube and 87 million clicks on their live blogs during 2018[3]. Soccer is popular on TV, but as the world becomes more mobile than ever with accessible internet, there is a higher demand for instant updates on our mobile devices.

In recent years, we have seen the trend of consumers wanting to consume as much video as possible in the shortest amount of time. In 2015, a study by Microsoft showed that the human attention spawn decreased

from 12 seconds (2000) to 8 seconds (2013) [43]. With these trends, we have seen platforms such as TikTok having enormous growth in recent years. They report 800 million active users and 2 billion downloads in App Store, bypassing YouTube, Instagram, and Facebook with 33 million downloads as of Q1 2019 [22, 45]. Further, we see a rising trend in providing events in both research [7, 36, 79] and real systems [1]. In the context of soccer highlights, short and concise summaries have been the standard in sports news coverage, but these do not cover all games or events and each highlight can not be watched separately. It is fair to assume that it would be of benefit to make more events available with short and concise clips of soccer highlights .

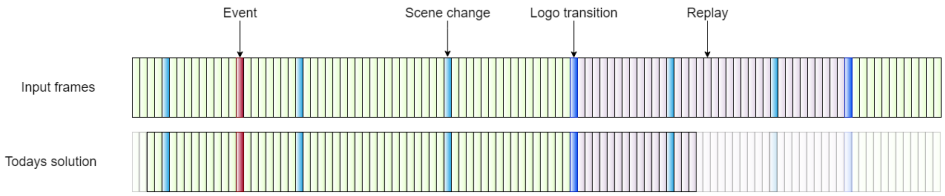

Figure 1.1: Visualization of how the solution used today can cut when using a fixed interval. This is an example of the cutting in the middle of the replay.

Today, highlights in soccer matches are manually annotated and clipped by human operators. This is a time-consuming, tedious, and expensive task. The clips are often a preset time interval instead of a tailored interval that fits the specific event as shown in Figure 1.1. The editors might not even have time to clip it as it can often be important to distribute it as close to the live event itself, like on a betting site. It could be edited later, but in many cases, this is too expensive. Due to all this, the highlights are in many cases of poor quality. The clips often start way too early or in the middle of the event of interest. It often ends abruptly in the middle of a replay as well. The celebration is part of the sport and should in many cases be included, so the end timestamp is also an important aspect. It is important to keep the momentum going if the highlights are being played back to back. Good highlights should be short and exciting, while still giving enough context to understand what is going on. It should start and end on reasonable timestamps.

Automatic event detection and clipping can increase the availability of user consumption. Video files that contain such data are valuable in themselves, providing statistics from events that can be useful for fans, gambling companies, coaches, or fans reading text-based summaries of matches. A system like this can be especially useful for teams in lower divisions with limited funds and save a lot of time for people who have little knowledge about editing by making the clipping automatic by the press of a button. You can find a lot of papers online about event detection and video summarization [5, 17, 27, 41, 49, 54], but they mostly focus on the part of spotting a goal, card, substitutions, and other events. While

[1] highlights.eliteserien.no

most papers focus on the task of spotting, we will focus on a task that has received less attention, i.e., using machine learning techniques to find the best start and stop for clipping the highlights in soccer matches. For example, the current clips clipped by Forzasys[2] in the Swedish Allsvenskan and Norwegian Eliteserien are initially clipped using a static value of seconds before and after a highlight based on "averages", and is only edited manually if resources are available. Often, these static clippings are just fine, however, sometimes they are completely off by for example stopping the clip in the middle of a replay. Thus, We want to use machine learning to make the clipping function dynamic, less expensive, and much faster.

In summary, with the growing demand for sports highlights combined with most people having multiple and portable devices with internet access, we want to research an intelligent system that can automatically produce highlight clips from a timestamp with the help of machine learning and video processing. Research on event spotting is already a popular [17, 27, 41, 49] focus in the field of machine learning, and combined with our task, it would be completely automatic. We see that it is beneficial to be able to distribute highlights fast, as sports fans often use a second device to look at complementary content in parallel to a match. Given the drop in human attention spawn [43], we also want the clips to be concise, showing only the relevant action. A system like that would save time and money and produce more content and more compelling highlights than the existing solution today [3].

1.2 Problem Statement

Addressing the manual, tedious task of performing accurate clipping of events as described above, we want to research a high-performance method to extract compelling highlights in soccer. We want the system to extract compelling highlight clips using an already annotated timestamp of an event taking place. To do this we build on existing machine learning state-of-the-art solutions of shot classification, clipping, and summarization of sporting events, and make our own proposed model that will work on our specific problem. We also want to explore how good our clips are compared to the already existing clips in the Norwegian Eliteserien in a scalable manner. Because what defines a good clip is a rather complicated matter, that comprises both technical, more objective truths, while also being a subjective question. We perform a user survey where we compare our clips to the already existing clips. This way, we can evaluate not only the technical performance but also get quantitative and qualitative data that can give insight into the quality of the highlights in the eyes of consumers. Based on this, the research question we aim to answer is:

> *Can a machine automatically extract compelling highlight clips from soccer videos?*

[2] http://forzasys.com
[3] http://forzasys.com

To answer this question and narrow down the tasks into smaller parts, we have defined 3 research objectives that each will bring us closer to a final conclusion:

Objective 1 Research and design a system to automatically extract highlight clips from soccer videos. Identify and prepare the necessary data needed for development and final evaluation.

Objective 2 Implement a system for clipping highlights and perform an objective evaluation of the different modules used, i.e., logo detection and scene boundary detection.

Objective 3 Perform a qualitative and quantitative evaluation of the system through a user study that evaluates the subjective nature of high-quality soccer highlight clips.

1.3 Scope and limitations

This thesis will focus on the specific event type goal in the sport of soccer, but the solution was designed as a more general system, meaning that it can be adapted to other events with few adjustments. We limit ourselves to the Eliteserien dataset collected by Forzasys (season 2018) and the Premier League (season 16/17) subset from SoccerNet [18] dataset. The reason we do not use the full dataset of SoccerNet [18], is the fact that we have to make our own logo detection datasets, and SoccerNet originally a very large dataset with 500 matches covering six different leagues. Our computational ability is limited by the hardware we have available. This limits the amount of training, training duration, and storage space available. Due to the length of the thesis, we decide to only include video in our scope of work and leave out audio and commentaries, though it is considered during evaluation as it is still part of the finished highlight clips.

The number of participants and their diversity are limited due to the reach of our network. This impacts the subjective evaluation in the sense that most of the participants fall into our age group of 18-29 years old. We would also have liked to have had the participants view a much higher number of videos, but in a realistic setting, it is hard to find participants willing to use hours and hours watching soccer highlights. Therefore, not all types of goals are represented in the subjective evaluation and the number of comparisons shown to the participants is limited.

1.4 Research method

We have based our research method upon the report "computing as a discipline" written by the task force on the core of computer science which was established by the ACM (Association for Computing Machinery) education board in 1989 [52]. In this report, three paradigms are described which we will describe in general and how they link up to our thesis.

- *Theory paradigm* The theory paradigm is rooted in mathematics and consists of four different steps. These are (i) characterize objects of study (definition), (ii) hypothesize possible relationships among them (theorem), (iii) determine whether the relationships are true (proof), and (iv) interpret results.

- *Abstraction paradigm* The second paradigm, abstraction (modeling), is rooted in the experimental scientific method and consists of four stages. These are (i) form a hypothesis, (ii) construct a model and make a prediction, (iii) design an experiment and collect data, and (iv) analyze results.

- *Design paradigm* This thesis is mostly applying the third paradigm, design which is rooted in engineering and consists of four steps. These are (i) State requirements, (ii) state specifications, (iii) design and implement the system, and (iv) test the system.

Our work mainly falls under the design paradigm as we state requirements, design, implement, and test the system. For our system to be useful the system needs to reach a certain performance for logo detection, Scene boundary detection, and consumer satisfaction. We also fall under the theory paradigm as we have a theory that certain modules will be faster and fit better for our use case, and we also have certain hypotheses about how well some of the models will perform based on the participant's background. Furthermore, we collect the data and analyze the results in-depth to either confirm or discard our hypothesizes. Finally, we touch upon the abstraction paradigm through the use of machine learning concepts and different type of hyperparameter optimizations for the different models.

1.5 Main contributions

Based on the problem statement described in Section 1.2, we want to make a machine learning model that provides a soccer highlight of a high standard, and this involves objective evaluation of key modules and a subjective evaluation of the final system. We will here restate the objectives set in Section 1.2, and our main contributions in association with each of them.

Objective 1 Research and design a system to automatically extract highlight clips from soccer videos. Identify and prepare the necessary data needed for development and final evaluation.

To meet this objective, we research machine learning approaches for video summarization, Scene boundary detection, and logo detection. Based on soccer broadcast production, we propose a highlight clipping system based on logo recognition tailored for a specific league and season and a shot boundary detection.

We design our logo detection as a binary image classification task. We analyze state-of-the-art approaches in the field of image recognition.

We settle on VGG [62] and ResNet [32, 33] architectures, both reaching impressive performance on the ImagNet ILSVLC dataset [19, 58]. Our candidate logo recognition models are ResNet50V2 [33], a lightweight CNN based on the VGG architecture [62], a simple CNN architecture, and an SVM using VGG16 [62] as a feature extractor.

We create a frame logo recognition datasets for two different leagues, Eliteserien season 2018 containing $1,025$ logo and $7,025$ background frames, and Premier League season 2016 - 2017 extracted from SoccerNet-v2 [18] containing $23,194$ logo and $43,260$ background frames. Both with high quality with respect to the sampling and labeling quality, but differ in size and complexity of logos. To compensate for insufficient data from Eliteserien, we supplement with synthetic data using a script adding extra logo frames.

Shot boundary detection is a popular field of research and has shown great performance results in the recent years [39, 63, 64, 70]. For our shot boundary detection task, we use TransNet-V2 [64], a state-of-the-art model with great performance on the shot boundary benchmark datasets ClipShots [70], RAI [11], and BBC [10]. We will test TransNet-V2 with its complimentary pre-trained weights, trained on ClipShots [70] and generated transitions using clips from TRECVid IACC.3 [8], as well as do our training on soccer clips only.

To train and evaluate, we extract over $150,000$ clips of 100 frames containing transitions from the full SoccerNet-v2 dataset with labels suitable for TransNetV2 [64]. Finally, we prepare a subjective evaluation for our system and the current system used in Eliteserien, on the Eliteserien dataset.

Objective 2 Implement a system for clipping highlights and perform an objective evaluation of the different modules used, i.e., logo detection and scene boundary detection.

To meet this objective, we implement the candidate models for logo detection, using SVM and CNN. We experiment on the Eliteserien dataset and Premier League dataset and assess the performance using several metrics. We show that for the Eliteserien dataset both the SVM and CNN achieved satisfactory results for the task at hand and the VGG model with a grayscale input of 54×96 pixels achieves the best result with a 100% F1-score. We also show that with a larger and more complex dataset such as the Premier League dataset, the CNN still performs well, while the SVM models failed to reach satisfactory results. We further improve the CNN models by adding more variety of backgrounds, including hard samples extracted by our classifiers, which proves to be effective. We find that the ResNet model with an RGB input of 108×192 reaches the best scores with a precision of 100% and a recall of 95.5% for logo transition detection on five full-length matches.

We evaluate the state-of-the-art shot boundary detection model TransNetV2 [64] on the SoccerNet-v2 [18] dataset. We show that a

pre-trained version trained on regular video clips performs well on soccer videos for gradual and abrupt transitions. We experimented with training the model specifically on soccer clips, which show potential but does not reach the levels of the pre-trained model. We find the model to be frame-accurate and therefore a sufficient model for our scene boundary detection module.

We combine logo detection and shot boundary detection in order to form a full system that outputs highlight clips, with high technical performance. We implement two different clipping protocols. The first configuration of the system includes all the celebration scenes between the event and the replay, and the other configuration of the system excludes a number of celebration scenes.

Objective 3 Perform a qualitative and quantitative evaluation of the system through a user study that evaluates the subjective nature of high-quality soccer highlight clips.

For this objective, we perform a qualitative and quantitative evaluation through a user study for Our model - Short, Our model - Full, and the Original model used today in Eliteserien. 64 participants rate highlights of five goals generated by our system and the existing solution and compare them with each other. The rating goes from 1 (worst) to 10 (best). Based on the results from the survey, we find the following ranking of the models:

1 Our model - Short achieves an average score of 7.40

2 Our model - Full achieves an average score of 6.84

3 Original model used in Eliteserien today achieves an average score of 5.89.

We find that due to the random nature of the Original model using a fixed interval for highlight extraction it achieves low scores when it "misses", while in the cases where it "hits", the original model achieves decent results compared to the other models.

Further, we group the participants by soccer fans, sports fans, gender, age, and editing experience, and find that the ranking of the models remains the same for all the groups, but the preferences, scores, standard deviation, and median vary.

Finally, we identify possible biases for the different groups of participants and discuss possible biases and real-world factors that could impact the results.

Our contributions are interesting in the context of the problem statement, and the presented results are valuable as for how much impact a good highlight clip has on consumer satisfaction. We showed that the machine was able to provide highlight clips of reliable technical standards based on the technical results and empirical evaluation. From the gathered quantitative results from the online survey, we showed that the technical

performance in conjunction with our two different clipping protocols leads to better results than the solution of the fixed interval used today. We also identified that what is considered a compelling highlight is subjective, and there are differences in what production strategy the potential users prefer. Our work gives a strong foundation for further work with using machine learning to generate automatic highlight clips in soccer.

1.6 Outline

Chapter 2 - Background In the Background chapter we introduce key concepts and terminology in machine learning that will be used further thorough the thesis. We also discuss already existing approaches for the problem at hand and relevant concepts that could apply to our problem statement. This chapter lays the foundation for the ideas this thesis will build upon.

Chapter 3 - Methodology In chapter 3, we describe the datasets and their respective task. We address the differences and weaknesses, and how it is pre-processed. Further, we discuss our proposed solution and introduce the different candidate models for the different tasks. This includes architecture and hyperparameters which will be used for experimentation in the next chapter. We discuss how to evaluate the system through objective data as well as subjective data gathered from an online survey.

Chapter 4 - Experiments and Results In the Experiments and Results chapter, we discuss the training iterations before we present the results for our experiments. The strengths and weaknesses of our models are analyzed and we try to understand why the models perform as they do. We present and analyze results for logo detection and Scene boundary detection based on objective quantitative data. Finally, we make prototypes of our system, and evaluate their performance based on their outputting highlight clips, and continue addressing the results of the online survey.

Chapter 5 - Conclusion In chapter 5, the work of our thesis is summarized, and the contributions are presented. Furthermore, we discuss possible future work that can be done in the context of our task and this field of research in general, to improve today's solutions.

Chapter 2

Background

Based on the challenges described in the previous chapter, we aim to develop a system that can automatically extract high-quality highlights in soccer. To understand the problem and the solution, there is a need to understand the concepts on which it builds. This chapter tries to explain some of the basic underlying technologies and some related works.

We start by defining the events we are using in this thesis, then we define some key concepts in machine learning relevant to the problem we are trying to solve. Next, we explore related works using different approaches related to the task of image processing, video summarizing, and other machine learning research that could be used for our solution.

2.1 Event definition

An event in our context is a goal, goal attempt, or a card. An event is defined as a thing that happens, especially something important. It can be hard to quantify an event, as the duration of an event is not clear. Sigurdsson, Russakovsky and Gupta [60] found in an experiment that there is mostly consensus of the center of the event. Therefore, defining an event as instantaneous on the time of the main action in the center of the event would be reasonable. In this thesis, we work with events being defined as Norgård Rongved et al. [49] defines it. Goals are defined by when the ball crosses the goal line, goal attempt as when the player makes an attempt, and cards when the card is given by the referee. It is consistent with the annotations provided by SoccerNet, as well as the spotting provided by online match reports, which also defines events at one exact point in time [18].

2.2 Machine Learning

Machine learning (ML) is an enormously expansive field in data science. Its ability to learn through experience has been useful in many fields such as health, entertainment, and science. Machine learning at its most basic is the practice of using algorithms to parse data, learn from it, and then make a

determination or prediction about something in the world [72]. Therefore, we will in this chapter explain some key concepts in machine learning that lay the foundation for what we use in this thesis.

2.2.1 Supervised, Unsupervised and Reinforcement Learning

We often split ML into three categories, supervised, unsupervised, and reinforcement learning. Supervised learning is the most common form of machine learning. In supervised learning, we have a set of true labels $Y = \{y_1, y_2, y_3.., y_n\}$ which usually are annotated manually by a human. We also have an input dataset $X = \{x_1, x_2, x_3, .., x_n\}$ corresponding to the true labels. Supervised machine learning utilizes the known data to learn the mapping function from input variable X to output variable Y, finding the best suitable function $Y = f(X)$ such that mapping new unknown input data X yields correct Y. We use supervised learning in this thesis.

As opposed to supervised learning, we have unsupervised. Unsupervised machine learning uses data without labels and tries to find hidden patterns. We also have reinforcement machine learning which does not need labeled data. It uses software agents, which are programs or algorithms that have a set of rules to follow. These rules are set to maximize the result of the learning.

2.2.2 Classification

Supervised machine learning is often used for classification. The output is categorical. It is used to identify a specific class, e.g. classify pictures of animals to a specific animal. An example in the context of our problem is running our labeled data through a machine learning algorithm that identifies if there is a logo present or not. This is an example of a binary classification problem.

2.2.3 Regression

Regression is a type of supervised machine learning. Instead of outputting a class like in classification, it outputs a real number (score). The training data is a mapping from input to a goal target. Its goal is to identify the relationship between the input features and making a function that can accurately predict the correct score. An example of this could be to predict the temperature tomorrow based on the temperature of the previous days.

2.2.4 Dataset

Whatever the use case for your algorithm is, it needs data to learn from and evaluate performance on it. Datasets help you to organize unstructured data from different (the same) sources to get the target outcome. Your dataset must be of good quality and relevant to the use case because it is the foundation of your model [30].

In the context of supervised learning, we have our input data X and our corresponding labels Y that make up our dataset. To prepare the dataset for training, we usually split our dataset into three sets depending on our problem and solution. We have the training set, validation set, and test set.The training set is what is used directly to update the algorithm to make it fit better. The training set is most often the biggest subset of our total dataset.

The validation set is another subset we use during training to validate the results. This dataset represents the current state of the algorithm, as it is a more accurate measurement than the training set because it is not directly updating the weights. We often use the metrics on the validation to tune the hyperparameters of the model to evaluate how well our model generalizes and to prevent our model from overfitting to the training set. Even though we do not use the validation set directly on the algorithm, the tuning makes the model biased toward it as we tweak the model into what gives the best results on the validation dataset. This is why this is not used for the final evaluation. The test set serves the purpose of evaluating how well our model performs on unseen data and is never touched during training to keep it independent and general. It is used in the end to see if the algorithm is generalized and in a scientific context this is the set you use to give a final evaluation of your model. This should never be used before a final evaluation.

Overfitting

Overfitting is a problem to be aware of in Machine Learning. Machine learning tries to make an optimal solution based on the data we use in the learning phase. There is a risk of overfitting to this data, i.e. the algorithm works great on the training data, but fails to generalize to new data. Separating the data into train-, validation- and test sets are one way of minimizing this risk. It is also important that the data for the training is representative of the real data. The ultimate goal for a model is to be able to predict well on new unlabeled data the model has never seen before [57, 76].

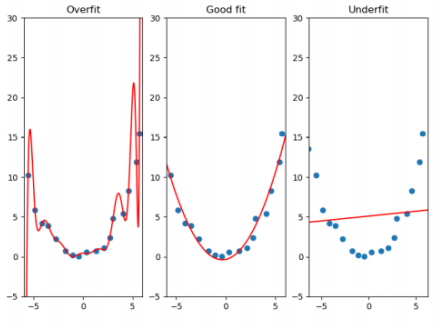

Figure 2.1: Illustration of 3 different functions (red line) used to fit the training set, Taken from [57].

2.2.5 Gradient Descent

Gradient descent is widely used to estimate optimization for a model. It is used to update the function iteratively, updating it little by little in the right direction until an optimal solution is found. A cost function, $C(X, w)$, is an estimate of how far off the model is from the optimal solution. By finding the gradient of the cost function with respect to the weight, $\nabla_w C(w_t)$, we can find the direction in which the weights should be updated to reduce the cost [57, 76].

$$w_{t+1} = w_t - \mu \nabla_w C(w_t) \tag{2.1}$$

Where t is time, μ is learning rate and $C(w_t)$ is the cost function. The learning rate is a hyperparameter that decides how big of a step the iteration will take. Higher values make it update faster, but it might not converge due to overshooting the local minimum. If it is too low, it will update slow. There needs to be a compromise between the two.

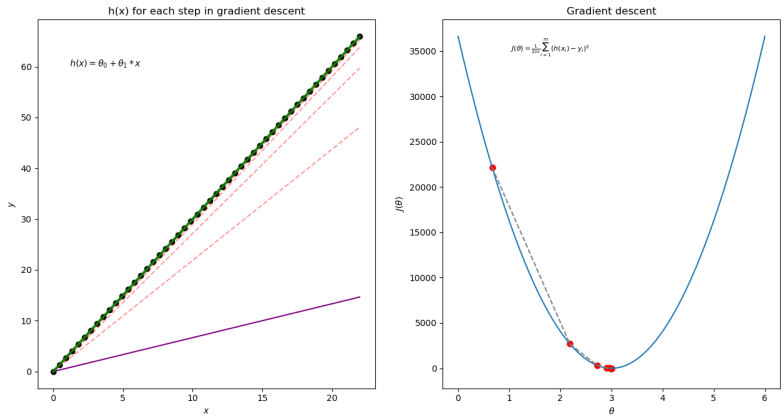

Figure 2.2: An example of gradient descent used to find the local minimum. On the left, we see an example of a linear regression line fit during each iteration; on the right, we see the loss for corresponding iterations of gradient descent. Taken from [57].

2.2.6 Convolution

Convolution is the operation of an element-wise multiplication and sum between a filter and a region of the same size of the input. With a 2D input, such as a frame, the filter 'slides' over the input image, outputting a 2D feature map, where each element corresponds to one application of the filter on a specific part of the frame. The filter is essentially a matrix of learnable weights that are trained to identify specific features [31, 76].

For one convolution, we often specify the filter size, stride, and dilation.

The filter size decides the local receptive field, meaning that it decides how much information we look at simultaneously. Today, we usually go

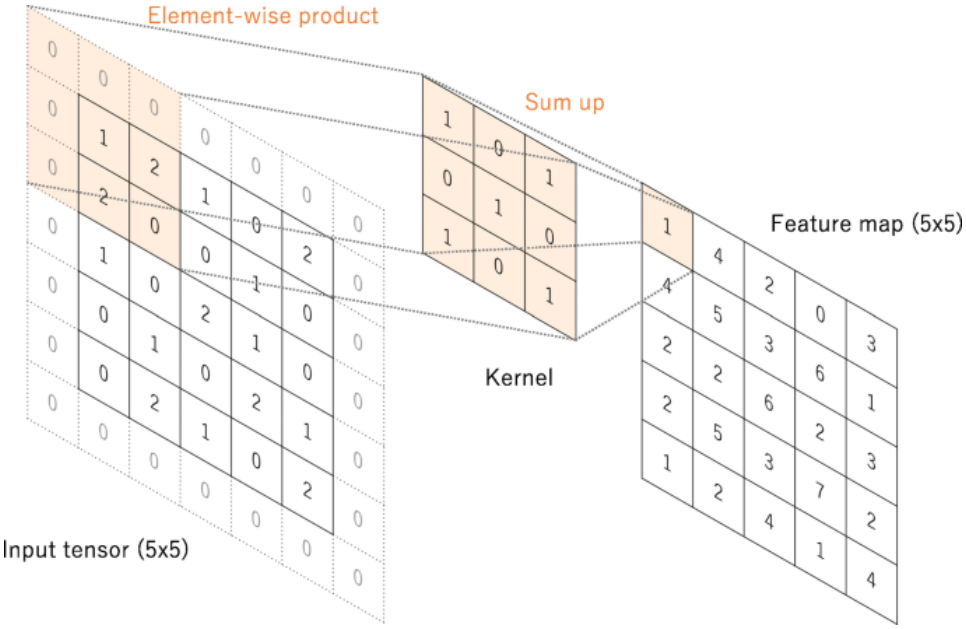

Figure 2.3: Illustration of a simple convolution using a 3x3 kernel, zero padding and a stride of 1. Figure pulled from [76].

for small filter sizes and instead go deeper which widens the receptive field. 3x3 is the most common, as it is cost-friendly. A 1x1 filter will only reduce the dimensionality, for example, map an image with three channels to a 2D feature map. 2x2 and 4x4 are generally not used because we need the symmetry we get from odd-numbered size filters. Each element in the feature map would not point directly to one anchor point. 5x5 or bigger is very costly to train, and in most cases, it is better to use the 3x3 filter size with a deeper model.

Stride is the steps we take between each application of the filter. With a stride of 1, we apply the filter on every element. With a stride of 2, we skip every other element. This also increases the receptive field. Dilation decides the width and height of the kernel. If the filter size is 3x3 with a dilation of 1, the filter will look at the neighboring elements to the central element. If the dilation is 2, it will skip over one element on each axis. For filters bigger than 1x1, we also specify if we want to zero pad the edges. This is because the filter can not fit, leading to some lost information. Zero paddings are often used if the edges contain important information or to preserve the input size.

All this translates directly to 3D convolution, such as a video input which is a series of images. The only difference is that there is one more axis to move along.

2.2.7 Neural Network

A Neural Network (NN) is inspired by biological neurons of our brain. Its building blocks are perceptrons, which are interconnected nodes, which

can be over multiple layers. The perceptrons send the signal produced to an activation function, where the function is to decide if that perceptron is to "fire" or not, which is an analogy of how our neurons in the brain work. The activation function is usually non-linear, making the resulting function of the network non-linear and a universal approximator. Two popular activation functions are ReLU as shown in Equation 2.2 and Sigmoid as shown in Equation 2.3.

$$ReLU(x) = max(0, x) \tag{2.2}$$

$$\sigma(x) = \frac{e^x}{1 + e^x} \tag{2.3}$$

The patterns recognized by the Neural network are stored numerically in vectors and could represent images, sound, time, words, and so on. We can think of neural networks as a model that helps us cluster or classify our data [31, 57].

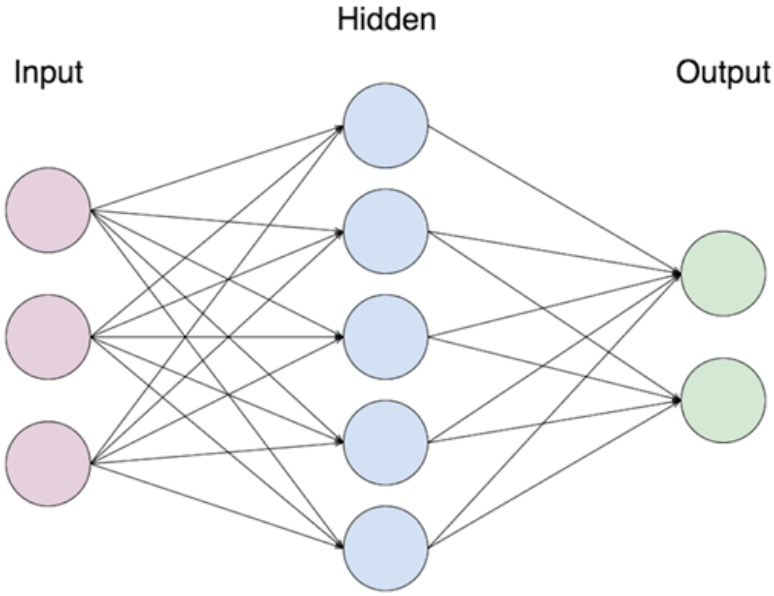

Figure 2.4: A illustration of a neural network where the pink nodes illustrate the input nodes, blue nodes illustrate the nodes at the hidden layer, the green nodes illustrate the nodes at the output layer and the lines illustrate the learn-able weights of the neural network. Figure pulled from [35].

The neural network also has a loss function. This loss function is used to give the model a state during training of how close it is to the goal. The goal would be to find a function approximation that most accurately maps input X to correct output Y for all data. If we look at the neural network as a function $f(X) = \tilde{Y}$, the loss function would be a function $f(Y, \tilde{Y}) = loss$, where Y is the ground truth mapping from X. Given this function, we can find the gradients with respect to the weights in the network and update them according to gradient descent. This way, the loss will become

less, and we will be closer to the target function. For binary classification problems, the most common loss function is binary cross-entropy. This is described in Section 2.4.

2.2.8 Convolutional Neural Network

Convolutional Neural Networks (CNN) combines convolution and neural networks. It often combines multiple convolutions as showed in Figure 2.3 and neural network layers as showed in Figure 2.4, each taking the output activations of the previous layer as input. The convolution uses multiple filters in each layer, each learning different features. The filters in the earlier layers, i.e. the layers closer to the input, interpret simpler features like edges, while deeper layers combine these layers and find more complex features like circles and squares, and eventually very complex combinations such as faces, hands, wheels, etc. We often add more filters to deeper layers because there are more combinations of features to learn. The output features of the convolutional layers are fed to the neural network. The neural network learns to separate the samples based on these [31, 76].

2.2.9 Pooling

Pooling uses a pooling operator to downsample a feature map. The pooling operator works almost exactly like a kernel in the convolution operation, except that the pooling operator either chooses the highest value or the average of the patch instead of taking the dot product. The pooling operator is almost always of size 2x2 with a stride of 2, meaning that each 2x2 region (with no overlapping) maps directly to one activation [76].

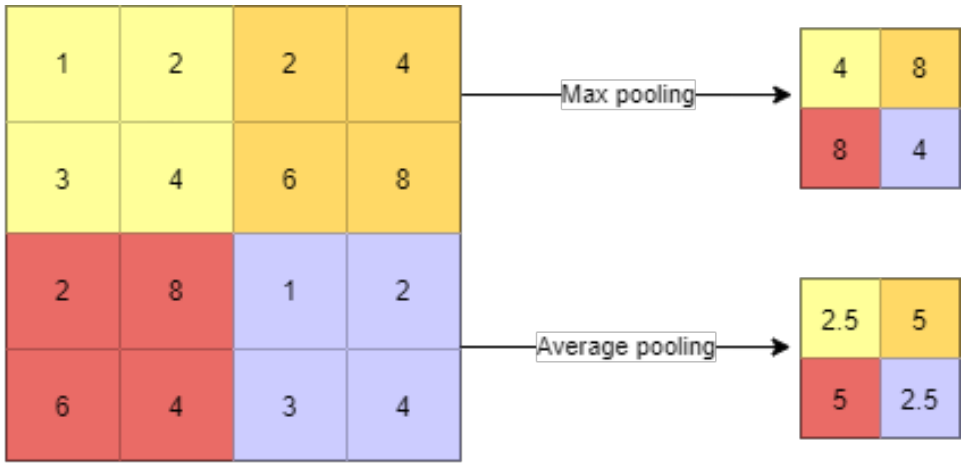

Figure 2.5: A illustration of max pooling and average pooling with a stride of 2 and filter size of 2x2.

The two main operators used are max or average pooling. Max pooling chooses the highest value in the active region, while average pooling takes the average as illustrated in Figure 2.5. This reduces the computational cost by reducing the number of learnable parameters without losing too much

information. It also makes the network less sensitive to the location of the features.

2.2.10 SVM

SVM is a supervised machine learning algorithm that can be used for regression and classification challenges. The SVM algorithm plots the data in a N dimensional space where N represents the N number of features you have. Where the value of each feature represents a coordinate. Then SVM performs classification by finding the best hyperplane to separate the classes. A good rule of thumb is to select the hyperplane that segregates the classes better [51, 78].

The SVM chooses the "best" hyperplane by maximizing the distance between the nearest data points and the hyperplane to help select the right hyperplane. This distance is called **Margin**, the SVM will choose the hyperplane that maximizes the margin, if you have a low Margin you have a higher chance of miss classification. One thing that makes the SVM so robust is that it contains a feature to ignore outliers and finds the hyperplane that maximizes the margin. Now that we have looked at linearly separable data, how does SVM handle data that is not linearly separable? We can solve this problem easily by introducing additional

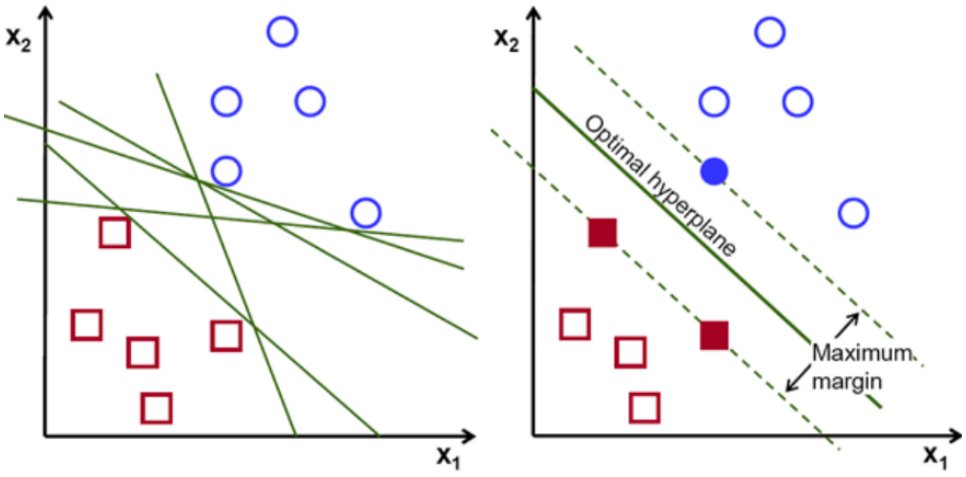

Figure 2.6: On the left we see potential hyperplanes for the SVM, on the right we see the optimal hyperplane that maximizes the margin, Taken from [51].

features manually before letting the SVM do its magic. But introducing additional futures manually also makes the computational cost of the SVM more expensive.

To solve this the SVM uses a kernel function to map the feature space to a higher dimension. This can be computationally expensive to transform all the data to a higher dimension, therefore the kernel figures out what the dot product in the space looks like instead of transforming all the data (this is computationally cheaper). It is important to note that this is still

an expensive and complex operation, so this is something to have in mind when choosing a model for your dataset. One of the kernel tricks we will be using in this thesis is the radial basis function kernel (RBF kernel) which is commonly used to separate non-linearly separable data.

2.2.11 Weight initialization

Weight initialization refers to the initial values of the weights. A network can be sensitive to the initial weight values [42]. Earlier, it was normal to initialize the weights between small numbers, such as $+/-$ 0.01, with a uniform distribution (all values are equally likely). The problem with this is that it can be hard to know what values to use. Reproducing other scientists' work can also be hard if these values are not documented. In 2010, Glorot and Bengio [28] proposed a method now known as Glorot uniform initialization (also known as Xavier initialization). They proposed to initialize the weights based on the number of input nodes and the number of hidden layers. The Glorot uniform initialization initializes the weights between $-s$ and s if $s = \frac{\sqrt{6}}{\sqrt{ni+nh}}$m where $ni + nh$ is number of input nodes plus the number of hidden layers. The bias is commonly initialized to 0.

2.2.12 Binary cross-entropy

For binary classification problems, it is common to use binary cross-entropy. The reason for using this is that it gives an exponential increase of loss the more off the predictions are. The formula looks like this:

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i * log(p(y_i)) + (1 - y_i) * log(1 - p(y_i)) \tag{2.4}$$

Where $H_p(q)$ is the loss over q elements, y is the true class (0 or 1), $p(y)$ is the predicted probability of the positive class (between 0 and 1) and N is the total number elements. Easy explained, we sum the log of the distance distance from the true class to the predicted probability over all elements, and then divide on the negative total number of elements, because log of values between 0 and 1 are negative. This leads to an exponential increase of loss the further from the true class the prediction is. The loss can be seen in Figure 2.7.

2.2.13 Exploding and vanishing gradient problem

When gradient descent is used for training a network, we calculate the derivative of a given loss function with respect to the weights and bias. We do this in what is called forward propagation. In the backpropagation, we use this to calculate the gradient and update the weights in the right direction according to the gradient. The more hidden layers, the more the gradients are multiplied. This is the reason for a problem referred to as the exploding or vanishing gradient. Small values will exponentially get

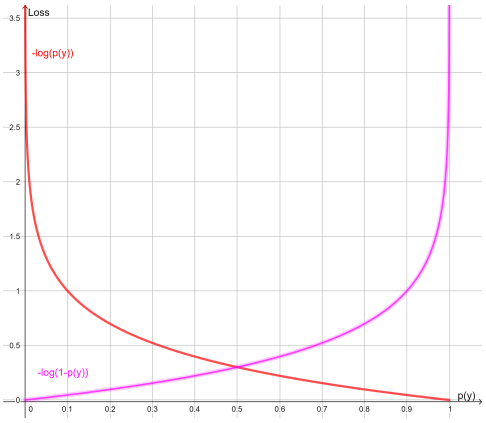

Figure 2.7: Illustrates the loss for y=1 (red) and y=0 (purple).

smaller and big values will get very big until they eventually overflow. This leads to the earlier layers being unable to learn.

2.2.14 Transfer Learning

Training a network takes a lot of time and resources. Therefore, using pre-trained weights as part of the network might be a good idea, leveraging already known knowledge to another problem. To continue on the analogy of our brain, our brain uses former knowledge from different scenarios when learning something new. Transfer learning uses learned patterns from other similar tasks to initialize the weights to kick start the initialization, and then train the model to generalize on our new specific problem.

2.2.15 Spatial and temporal features

Machine learning algorithms are very good at extracting spatial features. Classifying images has become extraordinarily accurate. Classifying events in videos is however a much harder task. Temporal features are features spanning over multiple frames or time. This is important to catch an event like a goal in a soccer video or to make a weather forecast where earlier conditions are important. The technology can also be applied to other inputs like MRI scans, helping us diagnose patients.

2.3 Definition of metrics

We use many different metrics when analyzing the results. It is very important to understand what the different metrics mean, and to know what metrics should be used to measure success. In machine learning, we look at a prediction as either true positive, false positive, true negative or false negative. Positive or negative refers to the predicted value, while true/false refers to if it is correct or incorrect. The metrics are objective data [29].

In many domains and specific problems, we use accuracy as measurement. Accuracy shows us how many of the true positives are found out of the total.

$$Accuracy = \frac{True\ Positives}{Total} \tag{2.5}$$

Precision tells us how many of the positives we trust actually are positives in reality. This unit measurement is important if we need all true cases to be correct.

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{2.6}$$

Recall tells us how many of the positive class is found. This is important if it is important to not leave out any positives. An example would be to fail to find cancer in a patient, as it would be much better to have false positives than to have false negatives.

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{2.7}$$

Sometimes we combine these two scores into one, by finding the harmonic mean between them. This is called the F1-score

$$F1 = 2 * \frac{Recall * Precision}{Recall + Precision} \tag{2.8}$$

Average gives us the average S over all the classes and tell us how consistent our model is across all the classes. here S is the calculated recall, precission or F1-score and i denotes the i'th class for $i\epsilon\{1, 2, 3, .., C\}$ for C number of classes.

$$Avg(S) = \sum_{i=1}^{C} S_i \tag{2.9}$$

One crucial factor is imbalance in the dataset. Therefore, we calculate the average and the weighted average. in this equation N_i is the number of samples in the i'th class.

$$WeightedAvg(S) = \frac{\sum_{i=1}^{C} S_i * N_i}{N_{Total}} \tag{2.10}$$

2.4 Related Works

Many papers and articles looking at machine learning to resolve and automate video-related problems [10, 11, 60, 70] and sport is a common topic. However, few papers focus on our specific task of clipping, but rather on the problem of finding the relevant event [27, 41, 49]. There are still many relevant works that offer possible solutions to different aspects of our solution [7, 17, 54, 55]. In this section, we discuss already existing work that is relevant to this thesis. We will first describe important work that has great success in the field of object detection. Then we move on and describe work done in the field of action recognition and how our system

can be tied to this. Finally, we describe several concepts and research done in the field of Camera shot classification, Replay detection, Audio, Sport summarization systems and, Temporal and Motion segmentation and how this research ties to the problem we are trying to solve throughout this thesis.

2.4.1 Object detection

Object detection has had a lot of success in recent years, and one of the main reasons is the availability of large datasets such as ImageNet[19], and their ImageNet Large-Scale Visual Recognition Challenge (ILSVRC). ImageNet contains over 15 million high-resolution images labeled in 22 000 categories. It uses the WordNet hierarchy (only the nouns). Each node in the tree is a category with subcategories, meaning that we for example have a category vehicle, with subcategories of boats and cars, etc. ILSVRC uses a subset of ImageNet with 1000 categories and 1.2 million images, 500 000 of which have bounding boxes for object localization [58].

In 2012, the winner of the ILSVRC was AlexNet [40]. It achieved a top-5 error of 16.4%, almost 10 % less than the second place[58]. This was a groundbreaking result and the beginning of large-scale deep neural networks. This architecture has around 62 million parameters, uses 5 convolution layers and 3 fully connected layers. The filter sizes used are 11×11, 5×5, and 3×3. The first 2 convolutional layers are followed by overlapping max pooling, and the last 3 are connected to the fully connected layers. The output layer uses softmax activation distributing the output probability of the 1000 classes.

In 2014, Szegedy et al. [66], a team from Google, entered the ImageNet challenge with a deep convolutional network called GoogleNet[58]. To overcome the problem of overfitting, the authors proposed making the system 'wider', by letting different filter sizes operate on the same level. It is then followed by max pooling. This is illustrated in Figure 2.8. The architecture is 22 layers deep and demands expensive calculations. This is why the authors also added 1x1 convolutions reducing the dimensions, meaning that each RGB pixel (consisting of 3 values) is reduced to one value. GoogLeNet achieved 6.67% top 5 error in the ILSVRC challenge of 2014[58] and was also the winner (image classification challenge).

The same year as GoogLeNet in ILSVRC, Visual Geometry Group of Oxford University submitted their architecture called VGGNet (VGG11, VGG16, VGG19). Simonyan and Zisserman [62] presented it in the paper 'Very Deep Convolutional Networks for Large-Scale Image Recognition'. This architecture had a huge impact on the community and has inspired many other architectures. The paper has been cited over 55000 times. This is probably due to the performance, the network achieved a top 5 error rate of 7.4%, which is the first time a deep neural network has gotten under the 10% mark. It was enough for the second place ILSVRC challenge of 2014 [58], behind GoogLeNet, as mentioned above. It is also fairly simple, and easily available with pre-trained weights on ImageNet.

The architecture uses VGG-blocks, which is a series of consecutive

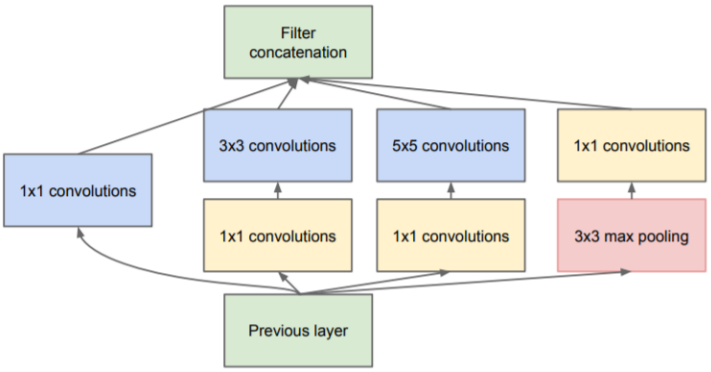

Figure 2.8: Inception module with dimension reductions. Notice the width (compared to VGG in Figure 2.9, and the 1×1 convolution used for dimension reduction. Figure taken from [66].

convolution layers with 3x3 filters and a ReLu activation function, followed by a 2x2 max pooling with a stride of two. For each block, the number of filters present in the convolution layers is increased. VGG16 has three blocks connected to two fully connected layers of 4096 channels each, using the ReLu activation function. The last fully connected layer has 1000 channels to fit the ImageNet challenge classes and uses the softmax activation function.

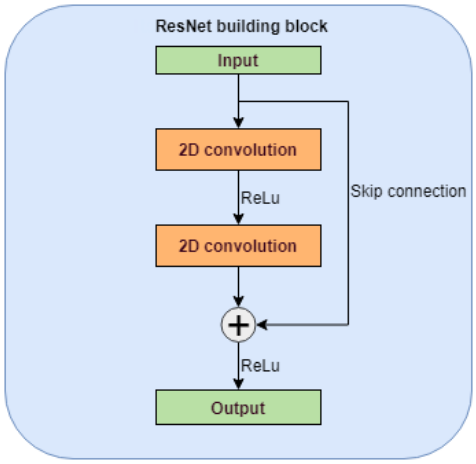

Figure 2.9: A residual block with two convolution layers.

Residual Network, or ResNet in short, was introduced in 2015 in "Deep Residual Learning for Image Recognition"[32]. It is a type of neural network that introduced residual blocks 2.9, where there is a skip connection (or shortcut connection). Deeper architectures attain more complex features but are also more prone to the vanishing gradient problem, described in Section 2.2.13. ResNet alleviates this problem with the skip connection giving the gradient a path to flow through. This allows ResNet to reach a great depth of 152 layers. ResNet achieved a 4.49 top 5

error rate and was the winner of ILSVRC 2015 in the image classification, detection, and localization task.

These are some of the most revolutionary architectures since ImageNet started its challenge in 2010. There have been many improvements making the performance on ImageNet even better. These includes SENet [34], ResNetV2 [33], InceptionV3 [68] and Inception-ResNet [67], the last of which combines deep networks with residual connections.

Enabled by high quality and big datasets, the mentioned architects have each made a huge impact in the field of object classification, recognition, and localization. For our task, we can use this to differentiate frames with and without a logo as part of a transition. These models are designed for bigger classification tasks, but we can take inspiration and possibly use smaller versions, as we do want to keep computational costs low. Comparing the performance of simpler networks to the more complex systems like these can give insight into what fits best in our system.

2.4.2 Action recognition

The results for the task of action spotting in soccer are getting better and better as time goes by, and eventually, these events need to be clipped into nice highlights. In 2018 SoccerNet 3.1.2 released the challenge of action spotting introducing a baseline model scoring an Average-mAP of 49.7% regarding the spotting task [27]. 2 years later the paper "A Context-Aware Loss Function for Action Spotting in Soccer Videos" [17] was released reporting an Average-mAP of 62.5%. But, this year in February RMS-Net[41] was released significantly improving the results, reporting an Average-mAP of 75.1% on the task of action spotting. These models all were tested on a more complex dataset such as SoccerNet for a lot of different actions. But, in the paper "Real-Time Detection of Events in Soccer Videos using 3D Convolutional Neural Networks" [49] a model achieving Average-mAP of 32.0% on the task of spotting in SoccerNet. The interesting part is that the model was tested on Eliteserien and Allsvenskan reporting 87% accuracy for Allsvenskan and 95.0% on Eliteserien when considering classification for the event Goal, which is inside the scope of this thesis.

The task of action spotting is highly related to our objective of automating highlight clipping. Our proposed system's aim is to make the production of highlight clipping automatic by transforming a video clip combined with one event timestamp into a high-quality highlight clip. To make the whole process fully automatic, we can combine our model with the task of event spotting for soccer.

2.4.3 Shot boundary detection

A popular strategy for making highlight clips in sports is to separate the video into smaller clips, where the video is cut on each transition from one camera view to another. Koumaras et al. [39] presents a shot detection algorithm using discreet cosine transform (DCT). Tabii and Thami [69] use this algorithm with soccer footage, first extracting the dominant color

and comparing them frame by frame. If the difference is above a certain threshold, it means there is likely a shot transition. they achieved good results with high resolution on the DCT, but with lower resolution, the recall suffered heavily. The results were 100% recall and precision of 94.23%. Zawbaa et al. [80] [79] followed this strategy and in addition, implemented a more tailored algorithm to handle cuts that transitioned gradually over several frames (smooth transition). This was done by skipping 10 frames to simulate an instant cut transition. They achieved 97.2% recall and 93.9% precision. These are good results but are trained and tested on very small datasets.

SoccerNet-v2 [18, 27] is a big dataset of broadcast soccer videos, which include over 500 full-length matches. We describe the dataset in more detail in Section 3.1. It is used as a benchmark for soccer event action spotting, camera segmentation, and shot boundary detection. Deliège et al. [18] reports the benchmarks for four approaches of shot boundary detection on this dataset. The results can be seen in Table 2.1. CALF [17], an abbreviation of Context Aware Loss Function, is a deep learning network originally made for the action spotting of SoccerNet-v2 but was moderated to fit this task. Using the scikit-video library [21], they made two boundary detectors, Histogram and Intensity. Histogram reports a scene change when the histogram intensity difference of two consecutive frames hits above a threshold, while Intensity uses color intensity in the same manner. Content uses the scene detection library PySceneDetect [14]. It uses the content-aware option, which detects boundaries based on changes in the HSV color space (hue, saturation value).

Method	Bound det.	Abrupt	Transition fading	Logo
CALF [17] (det.)	59.6	59.0	58.0	61.8
Intensity [21]	64.0	74.3	57.2	28.5
Content [14]	62.2	68.2	49.7	35.5
Histogram [21]	78.5	83.2	54.1	82.2

Table 2.1: Leaderboard for and Boundary Detection (mAP %), reported in [18].

Souček and Lokoč [64] proposes TransNetV2, a model that uses series of convolutions, RGB histogram and learnable similarities between frames to detect scene boundaries [63, 64]. The learnable similarity is a learned function that gives two frames a score according to how similar they are. They report state-of-the-art performance on the shot boundary benchmark datasets ClipShots [70], RAI [11], and BBC [10].

As seen in Table 2.1, there are some variable results. In order to use shot boundary for clipping, it would need to be more stable predictions. TransNetV2 reports very good results, although it has not been tested on a large-scale soccer database such as SoccerNet. Soccer frames are very similar in colors (much green) for the vast majority of scene changes, making it harder to spot the changes. It would still be interesting to see if it could outperform some of the reported models in Table 2.1, and help

produce better quality highlight clips.

2.4.4 Camera shot classification

Extracting high-level features can be important when low-level features do not map on certain classifiers. Shot classification is one example of higher-level features, which is often used when analyzing sports events with machine learning. Labeling each shot with a class could play a major role in where to clip a highlight. It would be intuitive that where on the playing field the camera is viewing has a correlation of the importance of the shot of the event being highlighted.

In the paper "Algorithms And System For Segmentation And Structure Analysis In Soccer Video"[6] published in 2001, the authors proposed an algorithm for classifying soccer-segments as play/brake based on a set of rules and camera-zoom. Because of the structure of a soccer game, the author used the grass-to-color ratio to label a camera shot as either global, mid-view, or zoom-in, based on the labeling they were classified as play/break with respect to some set rules (looking at neighboring classes). The system was tested across 4 different leagues and reported an average global accuracy (correctly classified duration's of play/break) of 76,35%, and 84,5% for the task of camera-view classification [6]. Zawbaa et al. [80] [79] classified the soccer shots as long, medium, close-up, and audience/out of the field, with good results of above 85% for precision and recall for all classes except for the audience class getting a 59.6% recall. They used low-level features such as the grass ratio which they extracted when finding the shot boundaries. Each class had different thresholds. The black color ratio was also used to distinguish between audience and close-ups. Minhas et al. [44] used the AlexNet CNN model to classify the shots from sports videos. They used a deep-learning model, and it demonstrated good accuracy of 94% accuracy. Rafiq et al. [54] proposed a model for classifying scenes in cricket. They used AlexNet CNN deep-learning model as well, but took advantage of transfer learning. It was pre-trained on ImageNet, a database of images with associated nouns. They achieved an impressive 99.27% precision and a recall of 99.26%, which gives an F1 score of 99.26%

We see that other papers have achieved good results regarding scene classification [6][79][54]. The idea of classifying scenes as play/break, attack, and audience could help us find relevant scenes for our highlights and identify which type of rules should apply when clipping, We also see that a pre-trained model on ImageNet achieved impressive results of 99.26% F1 score[54], so this opens up the possibility of using pre-trained weights instead of doing all the training from scratch.

2.4.5 Replay detection

Clips of events may contain replay clips, both before and after the event. Replay is of great interest for highlight clips, showing the important moments, while also having good quality shots. Replay detection can also

help with filtering out irrelevant replay for event detection. Zawbaa et al. [80] [79] implemented two different logo-based replay detection, one using a support vector machine (SVM) and the other using an artificial neural network (ANN). The SVM algorithm achieved 98.1% recall and 92.8% precision, while the ANNs recall only achieved 69.6%. In 2005, the authors of "Football Video Segmentation Based on Video Production Strategy"[56] built further upon the idea of using the play/brake, grass-to-color-ratio and camera-shot [6]. The authors used these previous ideas to introduce the class labels play, focus, replay, and breaks (using logo detection). Using these labels to classify segments as attacks. The authors also proposed a new indexing scheme built on "attack" and on this new indexing scheme, they introduced a "related video browser" (looking at nearest neighbors) and "summary browser" (show all proposed video segments and the ability to remove or insert segments in the summary) [56].

The safe assumption that a given league follows a standard production pattern after a specific highlight is perhaps the most important thing to keep in mind when approaching our problem statement. Detecting the replay in soccer using a logo-based approach has been proven to be effective using the SVM algorithm and not so effective using an ANN [79, 80]. With the prior knowledge of a production pattern and having the ability to detect a logo and a replay, this could be used for the task of finding cut points of highlights.

2.4.6 Audio

Raventos et al. [55] used audio features to give an importance score to the highlights. This could potentially be used for clipping as well, as audio from audiences is often a reaction to what is happening on the field. They use a change in audio power level as one of the audio features. A shot of the audience, for example, could be relevant for a highlight clip by giving some context. Using the audio could give us information if it should be included or not. In 2003, the authors of "Sports Video Summarization using Highlights and Play-Breaks" [7] wrote a paper about a more audio-focused summarization method to reduce computational cost and to generalize across different sports. The authors designed an algorithm for detecting whistle detection for finding highlights and based on the frequency and pitch of the whistle sound they would set a threshold to determine if it is an important event or not. The authors also measured the level of excitement of the audience and commentators (commentators tend to speak faster and with a higher pitch during important events). They also utilized the visual aspect by analyzing text display (for example scoreboard), and by combining all these aspects come up with a framework for detecting and clipping highlights [7].

2.4.7 Sports summarization systems

Based on the papers discussed in the paragraphs above [6, 7, 56], the authors of "Machine Learning-Based Soccer Video Summarization

System" [80] proposed a video summarization system consisting of six phases: pre-processing, shot-processing, replay detection, scoreboard detection, excitement event detection, event detection, and summarization. In the replay phase, the authors used logo detection for the segmentation of the video, making the assumption that a logo appears at the start of a highlight and the end of a highlight. They trained an SVM classifier and a Neural Network classifier for this task They concluded that the SVM classifier was better suited for this task with a 98.5% recall rate and a 93.1 % precision rate, whereas the Neural Network had a 93.3% recall rate and 69.5% precision rate. It is important to note that these results were achieved on a relatively small dataset consisting of 5 videos from the World Cup Championship 2010, Africa Championship League 2010, Africa Championship League 2008, European Championship League 2008, and Euro 2008 [80].

It would be interesting to take the ideas mentioned above and see if it could generalize to different leagues and see if we can achieve these kinds of results on a bigger dataset since it is hard to determine if this is a good model that would work in a realistic setting when they only had a dataset of 5 videos.

2.4.8 Temporal and Motion segmentation

Capturing temporal features can be a hard task. It is important for a number of machine learning problems, such as action recognition, weather forecast, and for example analyzing MRI scans. For general action recognition, Simonyan and Zisserma [61] proposed a CNN architecture using two streams, one extracting spatial features pre-trained on ImageNet, and the other extracting temporal features with the optical flow as input. Carreira and Zisserman [13] added 3D convolution, and Feichtenhofer, Pinz and Zisserman [24] looks at 3D pooling in addition. C3D used 3D Convolution to learn spatio-temporal features compared to 2D filters [74]. Two-Stream Inflated 3D ConvNet (I3D) [13] using kinetics-400 [37] showed that inflating pre-trained 2D filters into 3D filters improved the results. Extracting temporal information from the video may be useful when clipping as well. The authors of "Motion Entropy Feature and Its Applications to Event-Based Segmentation of Sports Video" [4] used an entropy-based motion approach towards the problem of video segmentation in sports events. By computing the EMV (entropy motion value) as a function of time, the author formulated the task of segmentation as a change point detection problem where the author divides the EMV curve into segments by change points.

Extracting temporal information could be very interesting to use for clipping the highlights. It could for example be used with audio to prevent clipping in the middle of a word. It could also be used to capture the ball and player movement, game pace, and crowd reactions, maybe indicating the value of interest for a highlight clip. Using already used techniques to extract these features, and finding new effective methods for features revolving around the particular problem of soccer highlights will

be important for the final result.

2.5 Summary

The annotation operation of videos is expensive, boring, and tedious work. In this chapter, we started by defining the different events we can expect to face when working with soccer matches. Then, we moved on to describe how datasets are often split into training, validation, and test sets, and the reason for doing this. We described some key concepts in Machine learning relevant to our thesis, such as gradient descent and the concepts of SVM, CNN, and NN. Finally, we define some metrics to be used further in this thesis. The problem of annotating video is an active field of research. With increasing amounts of video data, there is a need for an effective and accurate annotation. Sports video annotation is a time-consuming and tedious process. The community is making big steps in research regarding automatic event annotation, but few focus on the production side of the clipping process. We think the next step is automating this as well, in order to show the result of finding the events in a subjective appealing clip. Therefore, by using earlier research and experimentation, we aim to improve the quality of the highlights.

The object classification problem has seen great improvements in recent years, for easier training and better performance. For image and video processing, we find that it is common to extract high-level features such as play/brake, shot transitions, replay and logo. Replay is of special interest when it comes to making a highlight. Combining these high level features with prior knowledge of production strategies regarding the event at hand (our main focus is on goals), could be used for annotation of the highlight start/end interval. We further see that CNN and SVM have been proven to yield good results finding these high-level features and that pre-training has shown to yield good results. Sound is also a feature that should not be overlooked as clipping in the middle of a sentence could be annoying and the sound itself based on noise from the crowd, commentator voice, and speech speed could provide valuable information about the event itself. We also discover that temporal information for a task such as ours could be very useful for our task as soccer is a sport with a lot of movement, pace, and scene changes when an exciting highlight is taking place. Using all this knowledge and ideas gained, we want to make our own model based on some of these ideas and experiment with different configurations. In the next chapter, we propose a selection of different machine learning models used for high-level feature extraction to find a good performing model for soccer videos, before putting them together to our final highlight system.

Chapter 3

Methodology

We have talked about the manual annotation operation used today, and how the research of automatic event detection has progressed 2.4.2. We have looked at research regarding highlights, and how these papers' main focus is often on the event detection itself. The segmentation provided is either segmentation of the replay scenes, a predetermined cut, or a cut based on the scores of the events itself [56, 79, 80]. While these papers have provided highlights, they have only provided an evaluation of how well the model can classify replays, events, and so on, without providing any subjective evaluation of these highlight clips. We have also looked at how shot boundary detection has been used to segment the input of different models to further classify them separately into classes such as play/break, camera shot classification, and replay [17, 55, 56, 79, 80].

We will in this chapter introduce our solution for automatic clipping, focusing on the production quality. We will also provide a qualitative and quantitative evaluation of the highlights produced by our model to further help us understand what makes a good clip in the eyes of a consumer. Furthermore, we want to set our model up against the solution provided today in Eliteserien that makes a predetermined cut at 10 seconds before the event and 25 seconds after the event, which has multiple problems. It can start too early, start a few frames before a shot boundary, start in the middle of a replay scene of a prior event, and so on. It also typically ends in the middle of a replay scene, or even before the replay has started.

To build a system that can automatically clip highlight moments from a soccer match, we need to identify the important parts surrounding an event. For example, for goals, we want to show the live footage of the goal attempt in addition to the replay. A preliminary screening of our datasets shows that there is almost always a logo transition before and after a replay. In SoccerNet, 1,693 out of 1,703 goals has a logo transition before a replay within 100 seconds after the annotated goal. 1,609 of these include a logo at the end before 100 seconds as well. This is also the case in our logo Eliteserien dataset. This is why we propose to make a logo recognition module that will help us annotate the ending of a highlight and ensure that the event is shown from different angles. We will also use shot boundary detection to improve the production quality of automatic

clipping. By knowing where there is a scene change, we can avoid clipping a few frames before it and instead decide on some rules on where to clip to get the quality we want. We can also use the scenes to determine where to cut if we do not find a logo transition or only one of them. Due to the decrease in human attention spawn and the rising trend of platforms providing short video clips to be consumed fast and in large quantities, we want to see if these trends generalize to soccer highlights 1.1.To keep the consumer more engaged, we also want to test out shortening the clips by cutting out some of the scenes between the goal and the replay. The input of our system will be the tag for the goal event and enough of the video clip to be able to make a better beginning and include the replay after.

3.1 Dataset description

To train and evaluate our proposed system for automatic highlight clipping, we use soccer match video clips from Eliteserien (2018) and SoccerNet [18]. From these, we have made two separate logo recognition datasets containing frames separated into a logo or background class. Eliteserien is a small dataset with annotation of events only. This makes the collection of quantitative data to evaluate our system hard. During the work on this thesis, SoccerNet-v2 [18] was released and provided us with the ability to expand the scope of this thesis. SoccerNet provides a lot of annotations as well as a huge amount of soccer footage, enabling us to gather much more objective data automatically. The differences between the two datasets can also provide more insight into how our system can be more generalized across leagues.

3.1.1 Eliteserien

The Eliteserien dataset consists of 300 clips of goals from Norwegian Eliteserien. These clips start 25 seconds before the annotated goal and end 50 seconds after, lasting a total of 1 minute and 15 seconds. The goals are annotated as the ball crosses the goal line, and we observe a +/- 2 seconds inaccuracy, though it is mostly accurate. All clips have a resolution of 960×540 at 25 frames per second and audio with commentaries.

One problem with this dataset for our application is that many of the clips are too short. The logo transition is not always present. There is also no other annotation than the time of the event, which means we can only evaluate the performance of our system by manually examining the predicted annotations.

3.1.2 SoccerNet

SoccerNet [27] has 500 annotated games from different professional soccer leagues. The dataset consists of untrimmed broadcast videos from each half of the game, meaning the full dataset has 1,000 videos, each half containing about 45 minutes each (plus 0 to 8 minutes of added time), adding up to

a total of 764 hours of video. The dataset contains videos with audio and 506,137 commentaries at 1-second resolution from online sources. The clips are available in high-quality (resolution of $1,920 \times 1,080$) and low-quality versions (resolution of 398×224). 6, 637 action events were annotated from parsed online match reports and manually refined to a 1-second resolution.

Furthermore, in the SoccerNet-v2 dataset [18], they added 14 additional action events with over 100,000 annotations, in addition to camera labels describing camera view and scene changes as a single temporal anchor. There is a total of 158 493 scene change timestamps with the additional information of type, such as a close-up corner, public, inside the goal, close-up player, and so on. The dataset also provides the type of transition. These can be abrupt changes (71.4%), fading transition (14.2%), or logo transition (14.2%). The final category for the annotations in SoccerNet-v2 is replay scenes linked to their associated action event. They make up a total of 32 932 scenes.

The videos were manually annotated. Where annotations traditionally have been an interval indicating that the event lies within, these are single temporal annotations instead. The three most relevant action events are defined as follow by the authors of SoccerNet:

- **Goal** is defined similarly to the IFAB rules [1], which is when the ball crosses the goal line.
- **Card (separately annotated as red and yellow)** Is defined as the moment the referee shows the yellow/red card to the player.
- **Substitution** Is defined as the moment the new player enters the field.

	Season			
League	14/15	15/16	16/17	Total
EN-EPL	6	49	40	95
ES-LaLiga	18	36	63	117
FR-League 1	1	3	34	38
DE-Bundesliga	8	18	27	53
IT-Serie A	11	9	76	96
EU-Champions	37	45	19	101
Total	81	160	259	500

Table 3.1: Overview of the SoccerNet dataset with respect to different leagues and seasons.

Since all the videos are from popular high-level leagues we can expect professional broadcast videos containing multiple views, replays, and a variety of standard video production techniques such as slow-motion, scoreboard, logos, animations, and so on. SoccerNet provides so much data, both in video and complimentary annotations, which we can get

[1] https://www.theifab.com/laws/chapter/30/section/82/

Class	Total
Goal	1643
Substitution	2849
Card	2145
Total	6637

Table 3.2: Distribution of the "main" events annotated in SoccerNet.

good amounts of quantitative data from. We can measure the technical performance of the modules on full videos using the annotations for scene changes and logo transitions, making it easier to evaluate.

Dataset	Videos	Logo transition	Shot boundaries
Train	29	1,999	9,321
Validation	6	393	2,464
Test	5	359	1,897

Table 3.3: Distribution of the full dataset compared to the expected input of 120 seconds * 25 frames per second, where two logo transitions of 20 frames each are present.

3.1.3 Logo recognition dataset

We have two separate logo recognition datasets with images of a logo and background class, made from frames in the Eliteserien and SoccerNet Premier League 2016/2017 datasets. They will be used to train and evaluate the frame logo classifiers. Making datasets can be a time-consuming task, and for our datasets, most of the job has to be done manually. The SoccerNet dataset 3.1.2 has annotations that help us to extract the data, but it is still much manual work to get the quality and quantity we want.

Eliteserien logo dataset

The Eliteserien dataset is made of images from 50 randomly selected clips from the Eliteserien dataset. We extract images from every 15th frame using the FFmpeg tool, described here 3.3.2. We further extract all frames around the ones containing a logo transition. The images are 108×192 pixels. We ended up with 1,025 logo images and 7,025 background images.

The logo transition frames, shown in Figure 3.2, are very similar to each other, considering 1/4 of the transition graphic cover up the whole screen, resulting in identical frames. Most display the league logo as well. The ones with a team logo are very similar to one another as they are all in a white box. A weakness in the background set is that it lacks diversity with regards to what we can expect in a full match because all our data is obtained from a few clips that only revolve around goals. The background/logo ratio does not correspond to the ratio we

Figure 3.1: Eliteserien: Random images from the background class (left) and logo class (right).

Figure 3.2: Figure shows the type of logo transition we can expect in the Eliteserien dataset. It lasts for 20 frames in total, 10 of which are fade-in, 5 fully covering, and 5 are fade-outs.

expect from our use case. With an expected input of 120 seconds, we get $120 seconds \times 25 frames per second = 3000$, with 2 expected logo transitions of 20 frames each, we get a background/logo ratio of 74, while our dataset has 6.85.

	Background frames (B)	Logo frames (L)	Ratio B/L
Dataset	7,025	1,025	6.85
Expected input	2,960	40	74
Synthetic[2]	N/A	896	N/A

Table 3.4: Distribution of logo transition and shot boundaries in SoccerNet Premier League season 2016 - 2017

SoccerNet Premier League Season 16/17 logo dataset

From SoccerNet, we choose Premier League season 2016/2017 (SoccerNet PL16/17). Using the camera segmentation annotations, we extract 50 frames around each logo transition. We find 5 different logotypes, shown in Figure 3.4. We also find two unique transitions. We manually categorized each logo transition into one of these, both for value when analyzing later, and to split between logo and background. The annotated anchor points

[2]Dataset with additional augmented logo images, described in Section 3.1.5

are similarly tagged by the annotators (within +/- 3-5 frames) for each type, making it an easier task to further separate the frames into logos and background. We count the number of frames for each type, take into account the uncertainty, and make a script to divide them into classes. We have also manually looked over the set, eliminating the errors that are exceptions to the rules. Because of the uncertainty for the start/end time of the frames, the start and end of the logos are less represented. Example images are shown in Figure 3.4

This leaves us with a dataset of over 85,000 images extracted from 1,999 logo transitions, of which 36,319 images are classed as logo and 49,947% are of backgrounds. During training, we notice a bias in the set due to the lack of diversity in the background, further described in Subsection 4.1.4. We expand the training set with 7,812 backgrounds from the training set matches from every 500*th* frame. We use a classifier to extract over 6000 hard samples to make the training set sturdier. All frames are at least 2 frames apart. No frames are duplicates from the old dataset. We also find 954 logo frames that are wrongly classified, and add them to the training set. All the frames are manually quality-checked. The training set now contains a total of 43,260 background images and 23,194 logo images. We refer to this training set as Train Medium. We experiment with an even bigger set, made by extracting every 100*th* frame. We call this the Train Max set. Due to worse performance, we settled on the medium-sized background set for training. The validation set is kept the same for comparability.

	Background frames (B)	Logo frames (L)	Ratio B/L
Train Max[3]	74,432	23,191	3.21
Train Medium[4]	43,260	23,191	1.87
Train Initial[5]	29,378	22,240	1.32
Validation	9,302	6,938	1.34
Test	9,102	7,004	1.30
Total[6]	61,664	37,113	1.66
Expected input	2,960	40	74.00
Full matches	130,960	1,380	94.90

Table 3.5: Distribution in the full dataset compared to the expected input of 120 seconds$\times$25 fps, where two logo transitions of 20 frames each are present.

The reason we choose to focus on only one season, the Premier League season 2016/2017, is because we want to keep the data at a manageable level because of how time-consuming it can be. Both considering processing videos and manually processing and categorizing frames. We want to ensure good quality. Premier League is the most watched football

[3]Training set with mined hard samples and every 100th background frame
[4]Training set with mined hard samples and every 500th background frame
[5]The first version of the train dataset
[6]Including the final version of the train dataset (Train Medium)

Figure 3.3: SoccerNet: Random images from the background class (left) and logo class (right).

Figure 3.4: The different types of logo transitions we can expect in the PL 2016/2017 dataset (from SoccerNet PL16/17).

league in the world, reaching over 978 million homes with live coverage in the 2018/2019 season [23, 53]. The 2016/2017 season is the newest season available on SoccerNet. This dataset is still much more diverse compared to our Eliteserien dataset, with several different types of transitions, infographics, and stadiums. With a total of 40 games, this is still a very large dataset.

Augmentation

Augmentation can be used to make a model more robust, reduce overfitting, build invariance, and level out imbalanced datasets. We want to make our models as robust as possible towards changes in the frames or unexpected patterns when it comes to logo/background appearance.

During training, each image in the training set gets a random degree of shear between 0 and 0.2. Shear distorts the image along an axis to rectify the perception angles and can represent looking at an object from different angles. The images also get a random value between 0 and 20% of zoom for each axis independently. The lost pixels are filled with the nearest pixels

value. There is also a 50% chance of horizontal flip. These augmentations happen on the fly, with the use of Keras ImageDataGenerator [16]. Because we use such a small value for shearing, it mostly just adds some small noise to the image, helping to prevent overfitting. Flipping will make the dataset virtually bigger, but it may also help the model to be more generalized concerning the positioning of a logo, as well as where the graphics of the live scenes are located.

Figure 3.5: Example of augmentations, original picture, zoomed, sheared and horizontally flipped.

3.1.4 Dataset for shot boundary detection

Our shot boundary dataset (SoccerNet SBD dataset) is made from all shot boundary transitions in SoccerNet, using FFmpeg 3.3.2 to make the clips. It is made for TransNetV2, which takes 100 frames as input in 48×27 resolution. The system only needs to train on 100 frame clips containing shot boundaries as it can learn what is not a boundary by all the frames that are not a boundary. Therefore, we extract 100 frames from each shot boundary. To make it more robust towards the variable placement of the boundary frame, we randomly select a frame between 30 and 60 which are to be the shot boundary. We also made sure that close shot boundaries were also annotated for each of our clips.

Dataset	Logo	Abrupt	Smooth	Other
Train	25,920	48,745	16,426	60
Validation	8,648	19,019	6,286	73
Test	8,637	17,844	6,027	20

Table 3.6: Distribution of the different transition types from the full SoccerNet-v2 [18] dataset.

The dataset contains over 150 000 shot boundaries, with 43,000 logo transitions, 85,000 abrupt transitions, 28,000 smooth transitions and 153 labeled 'other'. We also use a subset containing clips from Premier League season 2016/2017 (SBD PL16/17 dataset), which has a total of 12,323 transitions. We used a python script to process all videos using FFmpeg [73], and it was run on the DGX-2 server 3.3.1. All labels from SoccerNet (temporal anchors for the transition) were converted into the format used by TransNet V2 (frame number of start/end of each scene). We will also use the full videos from the test set of SoccerNet V2 [18] for evaluation. For training, TransNetV2 relies on two output heads, one of

which is used to find all transition frames. Because SoccerNet does not provide the number of transition frames, the transitions are classified as abrupt, smooth (gradual), and logo. In order to train this head, we label the logo as +/- 5 frames, and smooth as +/- 3. Most logos we have looked at is more than 10 frames. The smooth transitions are usually 3-5 frames, with some exceptions.

3.1.5 Data preparation

Both datasets are split into train, validation, and test sets. For Eliteserien, we split each of the 50 clips into 60% train, 20% validation, and 20% test. On the SoccerNet dataset, we split by what game the frames are extracted from and use the split recommended by the SoccerNet team. This results in 29 games for training, 6 for validation, and 5 for test. To try to lessen possible bias, we split the sets on the full games for SoccerNet PL16/17 and on clips for Eliteserien. This can mitigate the problems of the same game or clip having consecutive frames, or more general similarities, such as the team colors, digital graphics, grass, stadium, and lighting conditions. This way, we can better test the generalization of our models. This split is the same for the overall datasets, logo recognition dataset, and shot boundary dataset.

The training sets are used to directly train our model, validation evaluates the models during training, and the test sets are only used to evaluate finished models. For the logo frame classifier and logo detection module, presented in Section 3.8, we will use both the logo frame datasets as well as the full-length matches (SoccerNet PL16/17) and clips (Eliteserien) to evaluate, and we will do so in the context of each league separately. For Eliteserien models, we will additionally evaluate on some of the unused clips. The shot boundary detection system, described in Section 3.5, will be evaluated on the SoccerNet PL16/17 full-length videos. We will evaluate the system in the context of each dataset separately before evaluate the comparing and concluding the system as a whole. The test set for Eliteserien logos is used to evaluate the logo classifier, while some of the unused 250 clips of Eliteserien will be used for evaluating the whole system.

One thing to note about the dataset split, is that some team logos might not be encountered during training, such as in the goal logo in Figure 3.4 and the team logo transition shown in Figure 3.2. We can use this to evaluate if the features learned are very specific (overfitted) or if they learn the more general features present in all of the corresponding transitions. If this proves to be a vital flaw in the Eliteserien dataset, as it is very small to begin with, we have made a synthetic logo class dataset that can act as a supplement to the existing logo frame training set of Eliteserien. This way we can boost the performance even though we lack a complete dataset. Our python script generates a desired number of images by pasting manually cropped logos onto random backgrounds. The logo is randomly inserted and randomly scaled to fit the random background. This helps prevent overfitting. This script is useful to serve as a supplement for the training

set if more training data or a more balanced distribution of logos in the training set is desired.

Figure 3.6: 4 images from Eliteserien randomly inserted logo with random size.

This type of generating synthetic training images would be useful in the realistic scenario of a league implementing a new logo transition where no video or images of this exists. Because the soccer background will look very similar no matter the season or league, this script can be used to update the logo training data only. By only needing to insert the different transition frames into the script, this is a very effective method to update the model on the new logotype. This way the model will be up and ready to go when the new season starts.

3.2 Data preprocessing

Before we feed our data to the model, we apply normalization to the data. We scale the pixel values to values by zero centering it by subtracting half of the maximum pixel value of 255. We then divide it by 2/255. By doing this, we squeeze all our data between -1 and 1 while still keeping their relative value with respect to the other pixels in the image. The reason for doing this type of normalization is to prevent large weight values which can lead to an unstable model that will not generalize well. It also helps prevent the exploding gradient problem and makes our data less sensitive towards outliers. It also implies that our features should be weighted equally, and prevent higher values to change the gradient too drastically. We used the implementation of this method from the Keras library [16]

$$Normalization(p_c) = \frac{p_c - 127.5}{127.5} \tag{3.1}$$

p is pixel and c is channel. 127.5 is half of the max color intensity, which is between $0 - 255$

Because the input size of our models is all smaller than the original images, we have to downsize them. We use Nearest neighbor interpolation (nearest) to achieve this, a non-adaptive algorithm, meaning that the pixels are treated equally across the image. The intensity is chosen from the nearest pixel in the original image, preserving sharpness, but losing all information in between. When performing changes from the original 16:9 aspect ratio to 1:1, for example with the input of 72×72, the images are resized using the same interpolation. We can see how the image is resized in Figure 3.7.

Figure 3.7: Aspect ratio 1:1 compared to 16:9 .

3.3 Implementation

The models were implemented using python version 3.7.10 (gcc version 7.3.0) with numpy 1.19.2, Keras 2.4.3, Sklearn 0.24.1 and Tensorflow 2.4.1, and executed on DGX-2 server 3.3.1. The development of the models in this thesis is done locally on 2 different computers with the following specs:

- Computer 1 16GB RAM and NVIDIA GeForce MX350 GPU
- Computer 2 16GB RAM, Intel(R) Core(TM) i7-6700HQ CPU @ 2.60 GHz 2.59GHz and a Nvidia GTX 950M, 4 GB GPU

3.3.1 DGX-2

DGX-2 is the deep learning cluster we use for training and testing in this thesis when it comes to computationally heavy operations or memory heavy operations. It holds 16X NVIDIA Tesla V100 GPUs with a total of 512G memory. Using the DGX-2 servers significantly sped up the training time of one of our dummy models by about 200% using only one GPU compared to training on our machine [50].

3.3.2 Tensorflow

Tensorflow is an open-source library for numerical computation and machine learning. TensorFlow uses python to provide a convenient front-end API for the user while executing the applications in high-performance C++.

Keras

We use Keras 2.4.3 to implement and train all our machine learning models, and for loading and preprocessing our images. Keras is a high-level API of TensorFlow 2 and serves as a highly productive interface for solving

machine learning problems, with a focus on deep learning. The key abilities of Keras are:

- efficiently executing low-level tensor operations on GPU, CPU, or TPU.
- Scaling computation to many devices
- computing the gradient of arbitrary differentiable expressions.
- Exporting programs to external run times such as servers, browsers, mobile, and embedded devices.

Using Keras for the implementation allows us to use a high level API for building our machine learning models efficiently, with good performance when combined with the DGX-2 cluster 3.3.1. It includes a number of popular architectures, such as VGG16 [62], VGG19 [62], InceptionV3 [68], Xception [15], ResNet50 [32], ResNet50V2 [33], ResNet152 [32] (and more ResNet architectures) and many more. They come pre-trained on ImageNet [19].

Sklearn

Sklearn is a useful machine learning library in python. The library contains a lot of efficient tools for machine learning and analysis, such as regression, SVM, Gridsearch, confusion matrices, and so on. Sklearn was used for the development of the SVM models.

FFmpeg

FFmpeg is a powerful multimedia framework able to decode, encode, transcode, and more, with all common file formats, and many more. It is compatible across Linux, Mac OS X, Microsoft Windows, and more [2]. We use this tool to extract frames from our video datasets and save them as JPG files as well as load video as a NumPy array in python.

3.4 Logo transition detection

Our function of the logo transition detection module is to recognize the full logo transitions with one temporal anchor point without any false positives. Our strategy is to make a frame logo classifier that predicts a frame as either a background or a logo and then use a sliding window approach to determine a logo transition as a temporal point. That means that the frame logo detector has to recognize enough consecutive logo frames while not find too many false positives consecutively so that we can predict a logo transition with high confidence. The module will take the frames after the goal event as input, classify the frames separately, and then annotate a logo transition if there are multiple frames in a window present. The window size, stride, and frame rate will depend on the results of the frame logo classifier.

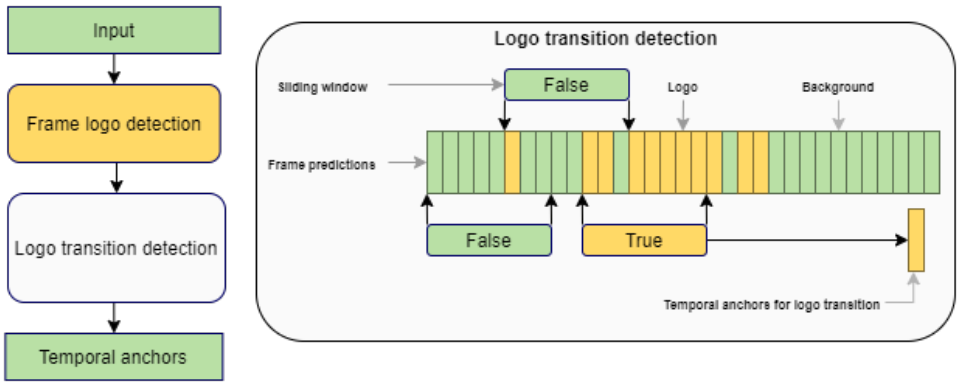

Figure 3.8: Our approach to find start and end of replay. Different window size, stride and frame rate will be determined by the performance of the selected frame logo detector.

For the module to predict a transition correctly, we need the frame logo classifier to perform well enough so that enough actual logos are predicted as such, while the number of clustered backgrounds incorrectly classified as logos is low. To evaluate the logo frame classifiers during the design and training stage of our proposed architectures, we will use the validation frame dataset. The two important things to look at are how many of the actual logos are found, and how many of the backgrounds are miss classified. For the SoccerNet Premier League models, we will also use the full soccer matches in the validation set to evaluate the module as a whole and further analyze the performances, before concluding training. We will then move on to test the performances on the test sets.

3.4.1 Feature extraction

Since we are working with images that contain a lot of data, we want to be able to reduce the dimensionality while still accurately and completely describing the original data. We want our model to extract relevant features for the task of recognizing logos, such as shapes, edges, motions, and complex patterns. The challenge here is to reduce the computational cost and memory usage of extracting these features while still keeping all the relevant information for our model. For this task, we have chosen more complex state-of-the-art models which have performed well on similar tasks such as VGG16 [62] and ResNet [32]refObject Detection. We have also chosen to implement our own smaller CNN and a lightweight VGG which is computationally cheaper to see how well it performs compared to these state-of-the-art models. Our goal is to find a sufficient model that provides good predictions while still computationally cheap.

3.4.2 Model selection for logo classifier

To find a model that best fits our system, we test different architectures and input sizes. We want to measure the performance of each model in relation

to the execution time. Our two datasets have different properties, such as size and logo features, and the best fit for each may differ.

A simple CNN

A small and simple CNN is fast to implement and train. It is interesting to see the results of this compared to deeper CNN and residual networks. The CNN logo detector is a shallow CNN model that consists of only two 2D convolution layers with 32 3×3 filters and ReLU activation. Both layers are followed by 2×2 max pooling with a stride of 2. It is then flattened, and sent to a fully connected layer with 128 neurons with ReLU activation. Last, predicting the binary outcome with a fully connected layer with a sigmoid output function. The total number of parameters with an input of $108 \times 192 \times 3$ is $4,720,801$.

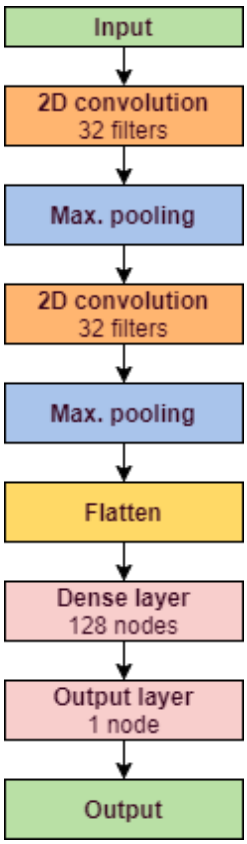

Figure 3.9: CNN model architecture.

The model is extremely fast compared to the others in Table 4.19. It achieves great scores on the Eliteserien, with an F1-score of 0.994 for the logo class with an RGB input of 72×72, running at $9,063$ fps. The with RGB 27×48 input, this architecture achieves the best precision of 0.995 and a recall of 0.983, running at $15,964$ fps. On the SoccerNet dataset, however, the shallow architecture has a harder time performing. The results are presented in Chapter 4.1

Residual Network

ResNet uses residual connections in order to be able to train deep networks, as illustrated in Figure 2.9. The network used in the ImageNet challenge (ILSVRC) achieved a 4.49 top 5 error rate. More about ResNet in Section 2.4.1. This is a very complex model which is probably well suited to the SoccerNet dataset. The drawback is the size and number of parameters.

We use ResNet50V2 [33] from the Keras library [16], which is based on the ResNet architecture described in Section 2.4.1. It is designed for the classification of a thousand different object classes from the ImageNet [19] database. We have swapped out the last two dense layers with a dense layer of size 128 with ReLU as activation function and an output layer with sigmoid activation function to fit the model to our problem. It is a very deep network with lots of parameters. With an RGB input of size 108×192, we end up with a total of $23,827,201$ parameters, $23,781,761$ of which are trainable. This is more than 5 times that of the simple CNN with the same input. ResNet50V2 comes with pre-trained weights trained on ImageNet [19], and we test using these as initial weights, with both fine-tune training and normal training. We only run RGB images on this network, as this is what it is designed for.

VGG

We use a model inspired by VGG architecture. This is a deeper convolutional model than our previous simple CNN. It has more layers and

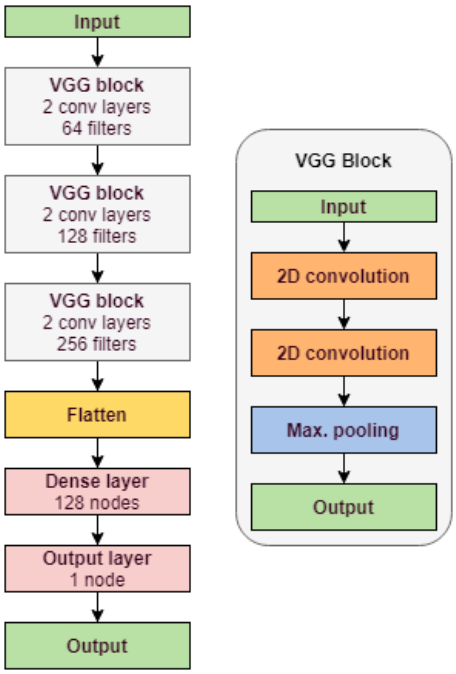

Figure 3.10: Architecture inspired by VGG.

more filters. VGG uses more filters for each convolution layer the deeper

they are positioned, and more consecutive convolution layers before max-pooling are performed. Inspired by this, we use a model with 3 VGG-blocks, seen in Figure 3.10. The first block has two convolutions with filters 64 filters each, the second block with two convolutions and 128 filters each, and the last block have four convolution layers with 256 filters each. All blocks end with a 2×2 max pooling with a stride of 2. The blocks are followed by a fully connected layer with 128 neurons and ReLu activation, and an output layer using sigmoid. With an input of 108×192, it has $12,549,441$ parameters.

SVM

The SVM algorithm achieved promising results for the task of logo recognition in soccer on a relatively small dataset, discussed in 2.4.5, so we want to see how well the SVM performs using our feature extractors on a small dataset (Eliteserien) and a large dataset (SoccerNet). Pre-trained CNN that have achieved good results on similar datasets are commonly used as feature extractors throughout machine learning [9, 20]. Therefore we want to explore how well the SVM performs using a pre-trained state-of-the-art feature extractor such as VGG16, which has achieved 92.7% top-5 test accuracy in ImageNet [19]. VGG16 is a relatively large and computationally consuming CNN. To compare performance to a computational cheap model, we use Simple CNN to extract features as well. VGG architecture is described in 2.4.1, and we explain SVM in Section 2.2.10.

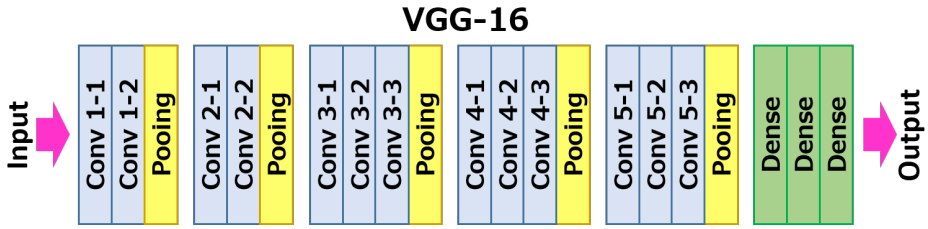

Figure 3.11: VGG16 architecture, figure taken from [48].

VGG16 is part of the VGG family of deep convolutional architectures, and follows the concepts of VGG as described in Section 2.4.1 and mentioned in the above section. The VGG16 architecture consists of 16 layers in 5 VGG-blocks. Each blocks convolutional layers uses 64, 128, 256, 512 and 512 filters for block 1, 2, 3, 4 and 5 respectively. It has three dense layers, two consisting of $4,090$ nodes and the last one consisting of $1,000$ nodes for each of the $1,000$ classes its weights were originally trained for in ImageNet. It is available with pre-trained weights (ImageNet) in the Keras library, which is described in Subsection 3.3.2

As we can see from Figure 3.12 our model starts by preprocessing the image using the pre-trained (on ImageNet) VGG16 but cuts out the dense layers. After the model gets the output from the VGG16 network, it flattens the output into a feature vector which is fed into the SVM. The SVM uses

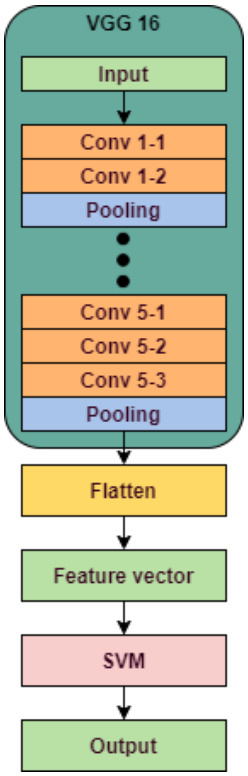

Figure 3.12: SVM architecture.

the decision boundary line to make a prediction based on the given feature vector. Based on the result the SVM outputs 0 for class background or 1 for class logo. The SVM CNN model works in the same way as the SVM VGG16 model, except here the flattening is done in the CNN model and not manually on the output as in the SVM VGG16 model.

3.4.3 Training and evaluation

For training of the CNN models, we initialize the weights using the Glorot uniform initialization [28] with Adam optimizer [38] and use binary cross-entropy as the loss function. Read more about initialization, optimizers, and loss-function in Section 2.2. Because ResNet50V2 comes with pre-trained weights on ImageNet [19], we will use these for initializing. All data is pre-processed as described in Section 3.2. We use the hyperparameters as described below.

The training of the SVM models starts by taking in the normalized data of the images in a given input size and feeding them into the CNN. After we get the output from the CNN and have our feature vector. Here we will perform a grid-search which is described below in the hyperparameters section. Through experimentation, we also define either a linear kernel or RBF kernel depending on if the data is linearly separable or not.

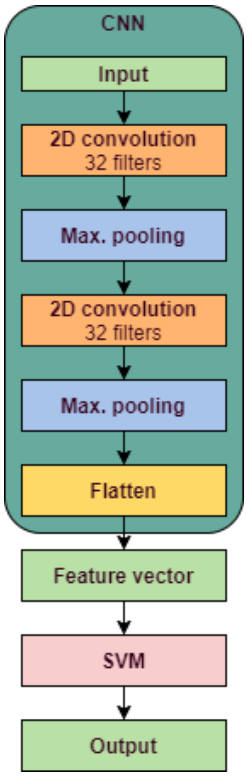

Figure 3.13: SVM architecture.

Hyperparameters

In general, we use a learning rate of 0.001 and 32 for batch size. We run training with early stopping using patience of 10, meaning we stop if the loss of the validation set does not improve (loss) for 10 epochs. We also reduce the learning rate by a factor of 10 with a patience of 7 epochs on plateau, meaning there is no improvement for validation loss. This is to fine-tune the model in the last epochs. We run training for a maximum of 40 epochs. We change some of these settings depending on the model we train. We use the Keras built-in methods for all our training of the CNNs [16]. These settings are used except for when we fine-tune the pre-trained ResNet50V2 model.

For the SVM we use grid search which is a type of hyperparameter tuning where we pass in a grid of parameters and grid-search will return the best estimator. The different parameters we pass in to the grid-search are "C": [0.01, 0.1, 1, 10, 100] (the regularization parameter) and 'gamma':[1,0.1,0.001,0.0001] (learning rate), so the model will estimate for "C=0.01, gamma=1", "C=0.01, gamma=0.1", "C=0.01, gamma=1" and so on. We define a max iteration of 100 epochs. After the grid-search is done we choose the estimator with the best score on the validation set.

To choose a window size and the number of frames required, we will use a grid search to find the best settings. Using the results of these, we can also determine the lowest frame rate of which are necessary to get the same

result. From this, we can calculate the theoretic computation performance based on the classifier's performance, measured in FPS.

Fine-tune ResNet50V2

To fine-tune ResNet50V2, we start by only training the dense layers. To do this, we freeze the weights of the base model containing the pre-trained model, only training the head containing two fully connected layers. We trained using the hyperparameters described in the section above. After this stage, we got a weighted F1-score of 0.9082 and 0.962 on the SoccerNet and Eliteserien validation dataset respectively, with an RGB input with 108×196 resolution. On SoccerNet, we get a recall above 97 % for all logotypes except for the Simple PL logo, shown in Figure 4.4. This got a recall of 0.6050. The results are described in Subsection 4.1.3 and presented in Tables 4.11 and 4.3 and

For further fine-tuning, we 'unfreeze' all the layers, and continue training. We set the learning rate to 0.0001 to help preserve the learned features from ImageNet [19]. We reset the optimizer with the new learning rate. We run for another 40 epochs (with early stopping). For the RGB input with 108×196 resolution, all scores for recall on the logotypes improved, but the recall for the Simple PL logo only reached 0.7776, while the others got 0.99 or better. Logo precision improved from 0.9505 to 0.9809.

Evaluating models

We want to be able to evaluate the different components of the system during and after training. Correct evaluations lead to better decisions for tweaking the design and selection of models. This can be a tricky task, as there are many aspects to take into considerations when analyzing the results. First and foremost, it is important to understand the meaning of the metrics we use and analyze them with the dataset and the actual use case in mind. We will use the metrics defined in Section 2.3.

To evaluate the classifier during training, we will use precision, recall, and F1-score for both classes from the logo frame validation datasets. We will also use full videos from the SoccerNet PL16/17 validation set to analyze the performance in a more realistic use case. Finally, we will use logo frame test sets to evaluate the final classifiers' performance, followed by a test on full videos. Testing on the full video matches will give us a more realistic overview of how the models will perform on real data. Comparing data from testing on frames and full videos can also give us a clearer idea of how well the logo frame dataset performs.

Measure computational performance

To calculate the execution time, we measure the prediction time of predicting $1,000$ random frames, including normalization. We use a seed when picking the random images to make sure we use the same for each measure. For each model, we run the measurement 11 times in a for-loop,

sort the results and report the average of the 5 middle values. This way, we do not let outliers influence too much. We do not measure the time of loading and handling resizing of the images, or take into account the memory usage. We used 500 images as batch size. We used the model's call function (opposed to the model's predict function) for predicting, as this is optimized for predicting images that fit in memory, according to the documentation [71]. This fits our use case. The results are reported in Section 4.1.

3.5 Shot Boundary Detection

Our shot boundary model will be used to detect scene boundaries in order to make decisions for the start point, possibly the start- and endpoint of the trimmed out part, and possibly the endpoint if a logo transition does not appear or are not detected.

In Subsection 2.4.3 we introduced some related works regarding shot boundary detection. In Table 2.1 we see some results reported on SoccerNet-v2 dataset [18]. We see that the mAP results for fading transitions (smooth, gradual) are not that great with the proposed models used. We, therefore, want to test TransNetV2, which reported a good performance on the more universal datasets. We want to see how it performs on soccer videos, and if it can be used to improve the quality of our highlights.

3.5.1 TransNetV2

To find camera shot boundaries, we will use TransNet V2, a state-of-the-art scalable architecture for shot boundary detection. The model comes with pre-trained weights trained on ClipShots [70] and generated transitions using clips from TRECVid IACC.3 [8]. In general, the network takes a sequence of 100 consecutive video frames and applies a series of convolutions, RGB histogram and learnable similarities between frames, returning a prediction for every frame in the input [63].

Stacked Dilated Deep CNN

TransNet V2s main layer component is called Stacked Dilated Deep CNN (SDDCNN), shown in Figure 3.15. Multiple DDCNN cells on top of each other, followed by spatial average pooling, form a Stacked DDCNN block. A DDCNN cell contains four $1 \times 3 \times 3$ (spatial) convolution, each followed by a $3 \times 1 \times 1$ dilated (temporal) convolution with the dilation rates of 1, 2, 4 and 8. With a dilation rate of D in the first cell, the kernel looks at the Dth frame to the left and right of the middle frame. TransNet V2 consists of three SDDCNN blocks with two DDCNN cells in each.

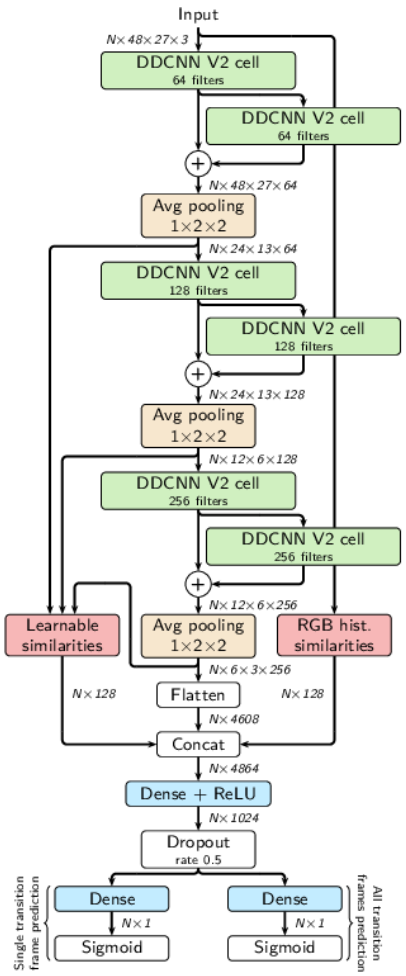

Figure 3.14: TransNet V2 Architecture taken from [63].

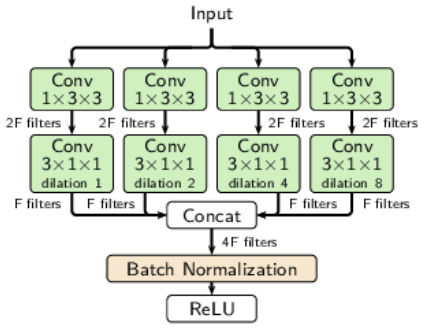

Figure 3.15: TransNet V2 DDCNN V2 cell with 4F filters, taken from [63].

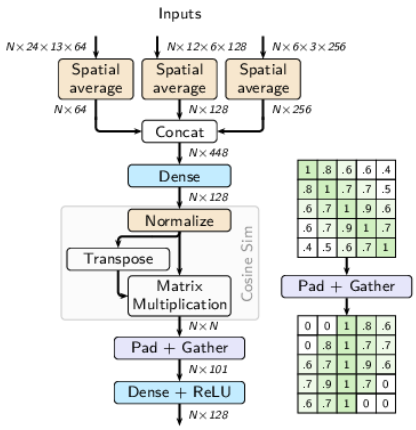

Figure 3.16: TransNet V2 Learnable frame similarities computation with visualization of Pad + Gather operation (right), taken from [63].

RGB histogram and learnable similarities

RGB histogram similarities and learnable similarities between the frames are also used by TransNet V2. RGB histograms, as well as learned features, are computed by spatially averaging activation of each average pooling [64]. See Figure 3.16. RGB histogram looks at the intensity and frequency in the red, green, and blue color space, giving each frame a similarity score. The learnable similarities are extracted from the three averaging pool layers. Both these similarities are projected by a single dense layer, followed by calculating the cosine similarity matrix. The output is a similarity score, representing the similarity of the frame compared to all the input frames on either side.

Multiple classification heads

In Figure 3.14, we can see that it has two output heads. One is for single frame predictions, used for predicting the middle frame of the prediction. This is the one used in inference. The second head, the all transition frame head, is used to predict all transition frames. Its purpose is to update the weights during training to help the models 'understanding' of the full transition [64].

Training

TransNet V2 model comes pre-trained, 15% of which are 'real' transitions extracted from ClipShots [70] and the rest from synthetically generated transitions made from clips from TRECVid IACC.3 [8] (35% hard cuts and 50% dissolves). ClipShots is a database with 166,756 manually annotated transitions from over 4000 online videos. 77% of the transitions are hard cuts while the rest are gradual transitions such as dissolves and wipes. TECVid IACC.3 is a database of 4600 internet archived videos (1800 hours of content) [8, 75]. We will compare the pre-trained model with the model trained on our shot boundary dataset.

We will start with the SBD PL16/17 dataset, a subset of the SoccerNet SBD dataset, as a preliminary experiment to see how the dataset and labels perform. We then continue training the same weights on the full SoccerNet SBD dataset in order to see if the performance increases. These experiments are described in Section 4.2. We run training on the small dataset for 50 epochs and continue training on the bigger set for 30 epochs. We train using the same configurations as Souček and Lokoč [64] uses. The positive class in the first single-frame head is weighted by a factor of 5. The second all-frame head's contribution to the loss is discounted by 0.1. L2 regularization is added to the loss weighted by 0.0001. We use Stochastic gradient descent with momentum set to 0.9 with a fixed learning rate of 0.01 [64].

To evaluate the two models, we will look at precision, recall, F1 score, mainly for abrupt and smooth transitions, as this is what this module will encounter the most. The metrics are described in Section 2.3. When evaluating, we use a tolerance parameter δ, which is the size of an interval centered on the ground truth frame, in which a transition must be predicted to be considered correct. $\frac{\delta}{2}$ is the maximum distance from the ground truth. This hyperparameter is important, as we are not sure how accurate the annotations are. Because the module is going to be used for clipping, we need the module to be very accurate, as more than a few frames inaccuracy can lower the quality of the editing. It can be hard to derive if inaccuracies are in the annotations or predictions from only looking at the quantitative data we gather from the evaluation of the models using different tolerance (δ) values, and we will therefore build an analyzing tool to look at a hand full of the predicted transitions manually. This way we can evaluate if it finds the transitions if false negatives and false positives are due to the tolerance and get a sense of how frame-accurate it is.

3.6 Our model

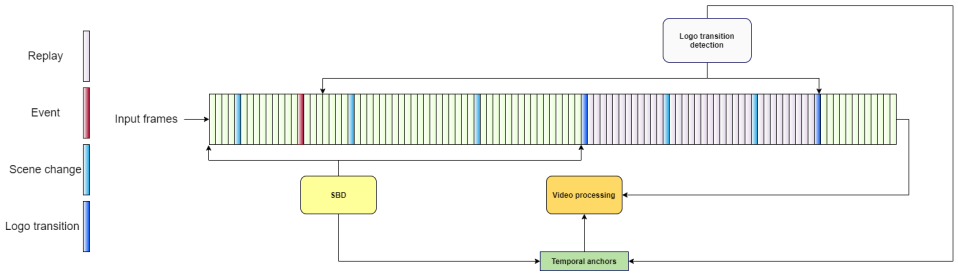

Figure 3.17: Our model.

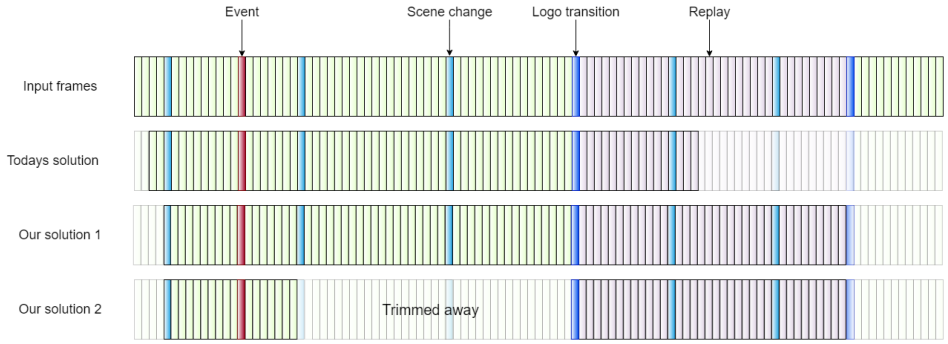

Figure 3.18: We want to make the highlight clips include all replay. We also want to experiment with shortening down the clips without losing the replay.

The full model is shown in Figure 3.17. The output of the system is visualized in Figure 3.18. The input of the system is a sequence of frames starting 15 seconds before the annotated goal event and 120 seconds after. The system first identifies the logo transitions, and then the scene changes between the first input frame up to the first logo transition. These timestamps are then used by the video processing module to process the input frames into a final highlight clip based on a clipping protocol. This protocol determines how the system cuts the output of the logo transition module and shot boundary detector. The protocol is available in the appendix in Algorithm 1.

The protocol determines the start point based on the scene starts present 12 to 5 seconds before the goal. It chooses the scene furthest from the goal. If there is no scene change, a default value of 10 seconds is used. The end is chosen in the middle of the last logo transition. We have added the option of cutting out some of the scenes in between the goal and replay. If a scene change is found between 5 and 10 seconds after the goal we cut. If not, we use the default value of 8 seconds. The protocol includes the last scene if it ends before 5 ahead of before the first logo transition, or 5 seconds if no scene change is present. This is to avoid a transition with poor quality. We designed these protocols by examining the production patterns in the

soccer broadcasts. Clipping at scene changes can improve the quality by avoiding starting a clip a few frames before a scene change. This utilizes the broadcast production as well, as a scene change often happens when something exciting happens. We will evaluate both protocols using an online survey, described in the next section.

3.7 Subjective evaluation

The end goal of our final model is to provide high-quality soccer clips in the eyes of a consumer. Considering this end goal does not have an objective truth to it, but is a rather subjective opinion that will vary from person to person, we need some form of measuring our final model other than technical reports. To evaluate our system, we have made an online survey. The goal of the survey is to get sufficient data to be able to better understand what makes a good clip in the eyes of a consumer.

To gather participants, we invite family, friends, and colleagues to participate in the survey without providing any information other than what is provided in the survey itself. In addition, we post the survey publicly in the IFI, UIO (Department Of Informatics, University of Oslo) Facebook group with 3,3 thousand members and sent the survey to OSI3 (Soccer team for students in Oslo).

We used Google form to create and host the survey. The survey is described in the following sections.

3.7.1 Background

The survey starts with an introduction part where we inform the participants that we are trying to make a machine learning model for the automatic extraction of highlights in soccer. During the form, they will be asked to rate different clips. In other words, we give some background information about the survey without revealing information about our model to avoid as much bias as possible as seen in Figure 3.19. The survey takes an estimated time of 10 - 12 minutes.

After the initial introduction, we want to gather some background information about the participants to be able to group the participants by different variables and see if there is any impact on the results based on which group the participants get categorized in. We also want to group the participants to analyze the group's representation with respect to the actual users of online highlight clips.

We first want to know some general information about the participants by asking them to fill in their age and gender. We want a diverse group, but also be able to take it into account if that is not the case. Furthermore, we want to see if there is some bias across genders or age groups when it comes to how they rate the clips.

In the next section, we ask general questions about the participants relation to sport in general and soccer as seen in Figures 3.20 3.21. We start by asking the participants if they consider themselves a sports fan.

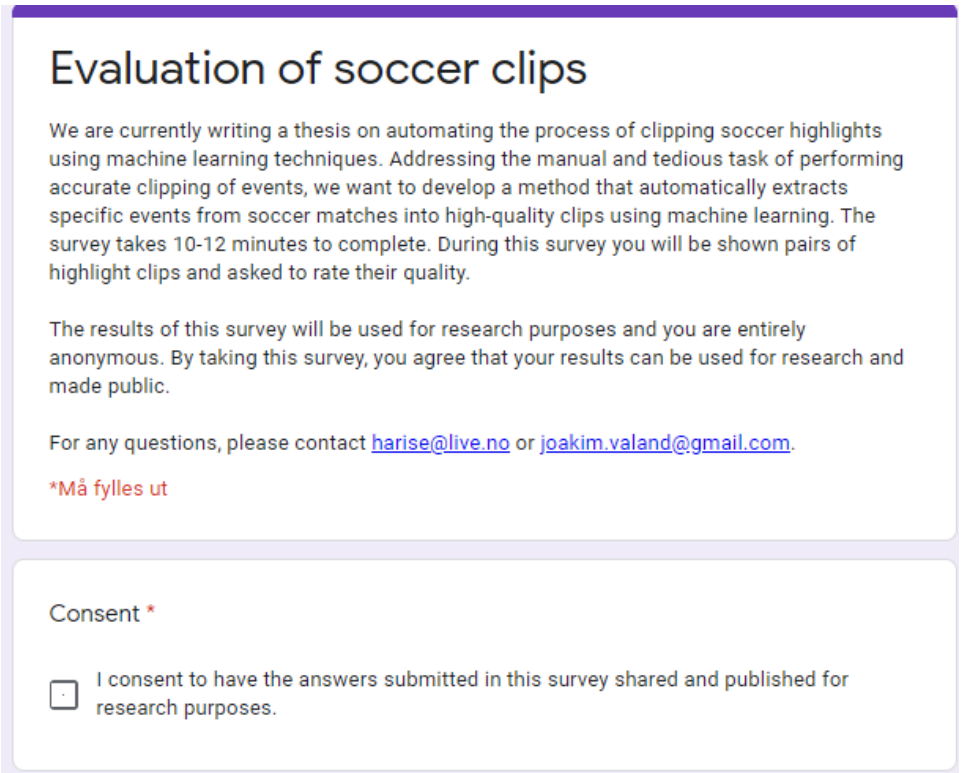

Figure 3.19: General information about the survey(the first page presented to the participants).

The reason for asking this is to see if general sports fan rates the videos differently. It is a reasonable assumption that familiarity with sports gives you a better foundation to know what to look for and what you want to see in a highlight, even on soccer specifically. Most sports with TV coverage have a similar pattern when it comes to highlights, for example, the pacing of how the exciting and crucial events play out. Further on, we want to consider the participant's foundation for rating a clip. We map how often the participants watch sports broadcasts and online highlights weekly. This can indicate the participant's understanding of the production quality and interest in sports content.

The next section is about soccer specifically. This is the important section where we want to filter out our target group from the rest. Therefore we ask these questions, "How often do you watch soccer matches on average?" and "How often do you watch soccer highlights on the web?". These questions provide us with data that will let us filter out the target group who will most likely be viewing these clips in a realistic scenario and probably has the strongest opinions on what they want to see or what annoys them about certain highlights.

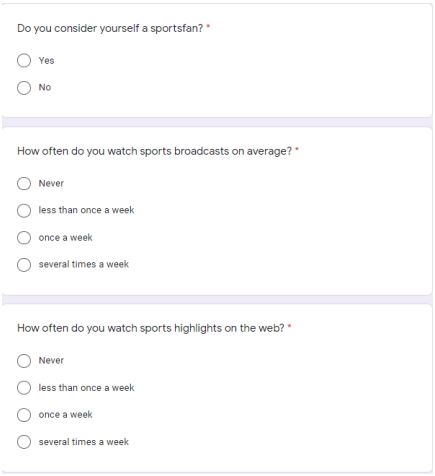

Figure 3.20: The general questions about sports presented to the participants.

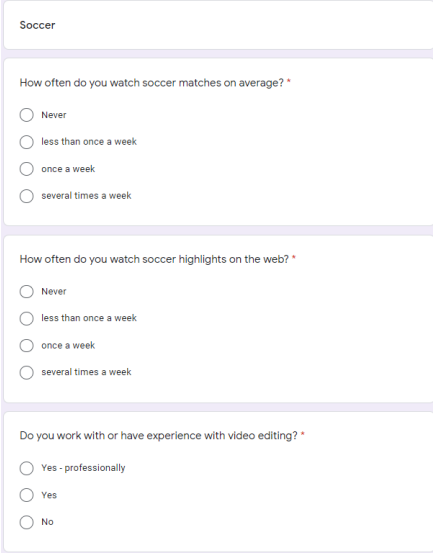

Figure 3.21: General questions about soccer and video editing presented to the participants.

3.7.2 Video event comparison

For the final section of the survey, we want to compare different clipping models against each other. In this part of the survey, we perform A/B testing[1], where each comparison shows the participant 2 different cuts (existing solution's cut, our short cut and our full-length cut) of the same event. The participant is then asked to give a score from 1 to 10 on each clip as seen in Figures 3.22 3.23 3.24. For each comparison we get insights to:

- What the quality of each video clip is
- What version is preferred
- How much do the clips differ in quality

The reason for having a 1 to 10 scale is that it gives more room for scoring the clips, which is useful in the cases where the difference is minimal, while still noticeable. For example, if the participant thinks a clip is slightly better they can give clip 1 score 8 and clip 2 score 9. If the scale were is 1 to 5, the participant would likely give both a score of 4, as there is no major difference. It can also be that they give the scores 4 and 5. This can make it harder to interpret exactly how good they think each clip are when looked at separately. We inform the participant that 1 on the scale refers to very poor, 5 is average, and 10 is broadcast ready. This is to reduce the room for subjective interpretation of the scale, though does not eliminate it. Our primary objective is to evaluate them with relation to each other.

We also provide the participants with an optional comment field for each comparison, where they can elaborate their ratings. This qualitative data can can provide us context behind given scores.

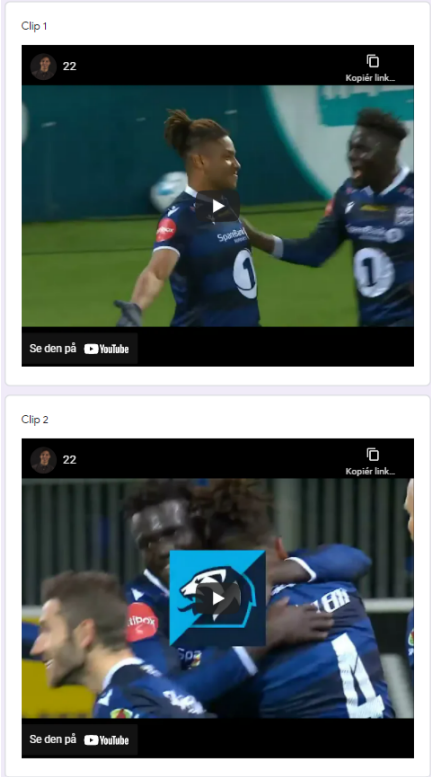

Figure 3.22: Figure of how the clips are presented.

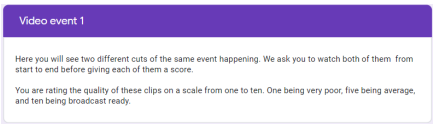

Figure 3.23: Description of the task presented to the participants.

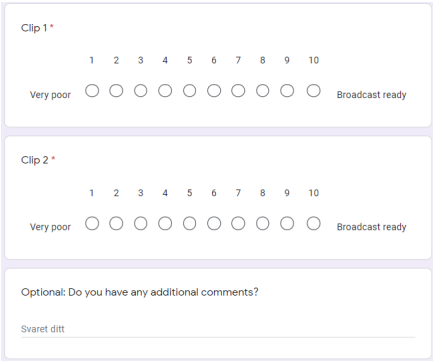

Figure 3.24: The scoring system and optional comment field provided for each comparison.

The structure of the A/B testing is set up as following:

- Clip 1 original VS Clip 2 our model - Short
- Clip 1 our model - Short VS Clip 2 our model with full crowd
- Clip 1 our model with full crowd VS Clip 2 Original
- Clip 1 Original VS Clip 2 our model with full crowd
- Clip 1 our model shorter crowd VS Clip 2 Original

For the highlight clips to be compared, we randomly choose the clips and inspect them to make sure they are not edge cases before inserting them in a random order for the survey. The order remains the same for every participant.

3.8 Summary

In this chapter, we introduce our proposed strategy for automatic highlight clipping of soccer goals. We continue to introduce the datasets of Eliteserien and SoccerNet [18] and how we use them. We further explain how we extract the logo recognition datasets with supplemented synthetic

data and the scene boundary detection dataset. Next, we describe how we preprocess the data to prepare for training and evaluation. Further, we describe which hardware is used for data preparation and experimentation, and what libraries we use for the implementation and development of the system. We then proceed to describe the Logo transition detection model of our system and the architecture of the different logo detection modules using a simple CNN, VGG inspired CNN, ResNet50V2 [16, 33] and SVM using VGG16 [62] and simple CNN as feature extractors. We discuss how the training on this module will be performed and how the metrics are used to evaluate the model's performance. We furthermore discuss the hyperparameters and hyperparameter tuning for use during training, testing, evaluation, and in a possible inference situation. We then proceed to discuss the Shot boundary detection module and how the training of this module is done using the architecture of TransNetV2 [64]. Further, we describe the architecture of the final system and the clipping protocol to be used for the final models. Finally, we present the method used for evaluating the final models' performance in the eyes of a consumer, by describing the structure of the survey and discussing the choices made for the survey.

Chapter 4

Experiments and Results

In chapter 3, we found a design for our system using a logo detection module, scene boundary detection module, and a video processing module using a production-based clipping protocol. We found a strategy to detect logo transitions using CNN and SVM logo frame classifiers and presented our candidate model architectures, and the shot boundary detection model TransNetV2 [64]. We explained our approach to test and evaluate the different models which we will carry out in this chapter.

We will start by presenting the experiments for the logo detection module on the Eliteserien dataset and the Premier League dataset, including training and preliminary evaluation providing insight and reasons for steps taken to further improve the models. Finally, we test them on the test set and analyzing the results. We proceed to present the experiments for the shot boundary detection module and analyzing the results. In the final section, we present the results for our final system using two different clipping protocols and compare them to the already existing model used in Eliteserien today. We evaluate based on our own technical analysis of the system's output, and analyze results gathered from the online survey. Finally, we discuss possible weaknesses and improvements of the experiments, biases, and other factors present in the survey data.

4.1 Logo detection

We will in this section present the experiments for the logo detection module. We will show the results on the validation set during training, and finally, report the results on the test sets. For Premier League 16/17, we include a test on the full soccer matches, to get a better picture of how it will perform with real data. We split the section into each league.

4.1.1 Model input

As mentioned in Section 3.4.3, we want to find a model that is fast and performs well. In images, much information lies in the colors used, but using grayscale can be much more computationally cheap. A higher resolution has more information, but also comes at the cost of speed. We

will therefore test with grayscale and RGB input. During preliminary experiments on Eliteserien, we found that even small resolutions perform well. This applied to grayscale as well. We also found that using a format of 1:1 performed well. The preliminary experiments were done on an earlier version of the frame logo dataset using the simple CNN. As reported in Subsection 4.1.2, small outputs yield good results on the validation set. For both datasets, we include 108×192, 54×96, 72×72 and 27×48 in order to compare the performance with a lower computational cost. Described in Subsection 4.1.3, we see a correlation between higher resolution and better performance. We decide to include 144×256 resolution for the SoccerNet dataset only, as the Eliteserien dataset was extracted with a smaller resolution of 108×192. We test with these resolutions on both datasets for comparability. ResNet is only trained on RGB images, as this is what the architecture is designed and trained on.

4.1.2 Eliteserien Experiments

We will in this section present the results from training and testing the models on Eliteserien as described in Section 3.4.3.

In Figure 3.2, we can see that the logos are very simple and take up much of the whole screen for most of the transition. There is also very little difference between the separate team logos, all of which are surrounded by a white box. To find the best model for this architecture, we tested on different input sizes, both in height and width dimensions, as well as the number of channels. We find that we can reach both good results and fast performance on this dataset.

Input	Precision	Recall	F1 Score
$108 \times 192 \times 3$	0.9793	0.9833	0.9813
$54 \times 96 \times 3$	0.9874	0.9792	0.9833
$72 \times 72 \times 3$	0.9917	**0.9958**	**0.9938**
$27 \times 48 \times 3$	**0.9958**	0.9733	0.9895
$108 \times 192 \times 1$	0.9790	0.9708	0.9749
$54 \times 96 \times 1$	0.9915	0.9750	0.9832
$72 \times 72 \times 1$	0.9833	0.9792	0.9812
$27 \times 48 \times 1$	0.9746	0.9583	0.9664

Table 4.1: Results for Simple CNN on the Eliteserien validation set.

The results for the Simple CNN on Eliteserien validation set is presented in Table 4.1. This shows an overall good performance and serves to prove that the transition is not very complex. It is surprising that the smaller input performs better on the RGB inputs, though there is not much of a difference. We can see that the recall for the logo class suffers when using grayscale. This suggests that this model learns important features from the colors to separate logos and backgrounds. We also notice that an input size of 72×72 has the best F1 and recall score. We also note that the

precision of the RGB 27×48 model is the best, meaning that it misclassifies fewer backgrounds as logos.

Input	Precision	Recall	F1 Score
$108 \times 192 \times 3$	0.9833	0.9792	0.9812
$54 \times 96 \times 3$	0.9833	0.9833	0.9833
$72 \times 72 \times 3$	0.9958	**0.9958**	**0.9958**
$27 \times 48 \times 3$	0.9514	0.9792	0.9651
$108 \times 192 \times 1$	**1.0000**	0.9500	0.9744
$54 \times 96 \times 1$	0.9957	0.9708	0.9831
$72 \times 72 \times 1$	0.9793	0.9875	0.9834
$27 \times 48 \times 1$	**1.0000**	0.9792	0.9895

Table 4.2: Results for VGG inspired CNN on the Eliteserien validation set.

For the VGG models, we note that the grayscale input gets very good precision. This is a deeper network that can see more complex patterns in the data, and may not have to rely as much on the colors to separate the classes, like in the Simple CNN. However, the recall suffers for the logo class. The RGB input performs better, and we see again that the 72×72 RGB input performs best, and it is slightly better than the Simple CNN. The results are reported in Table 4.2.

Training	Input (RGB)	Precision	Recall	F1 Score
Normal	108×192	0.9710	0.9750	0.9730
Normal	54×96	0.9835	**0.9917**	**0.9876**
Fine-tune	108×192	**0.9955**	0.9125	0.9522
Fine-tune	54×96	0.9671	0.8583	0.9095

Table 4.3: Results for ResNet50V2 on the Eliteserien validation set. All weights are initialized with the ImageNet weights.

In Table 4.3, we see that the freezing and fine-tuning ImageNet weights do not perform better for the recall. We suspect the learning rate of 0.0001 to be too low, making the model converge prematurely. However, we see that it scores very good on the precision for the bigger resolutions. This means there is a higher recall score for the backgrounds. This approach has shown good results for some tasks, as well as saving time on training. In our case, however, it seems the features are easy enough to learn from only initializing to the weights and train with a more fit learning rate of 0.001. But we also see a higher precision, which means there is a higher recall score for the backgrounds. In our actual use case, this can prove more important if the classifier predicts sufficient logo frames from each separate transition. We deem the performance of the more straightforward transfer learning of ResNet suffices, and we will use this going forward with the Eliteserien dataset.

Looking at the graphs in Figure 4.1 and 4.2, we see signs of overfitting. ResNet ran training 12 epochs and the Simple CNN ran for 14 epochs. For

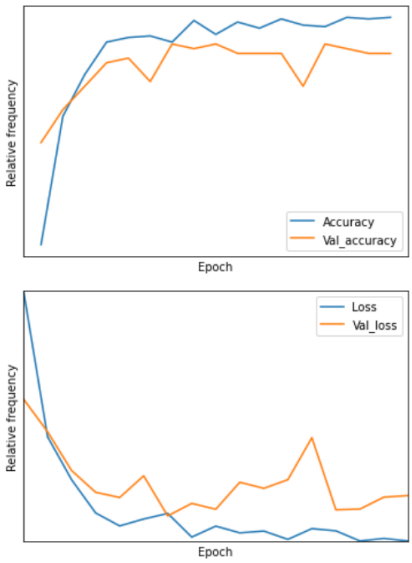

Figure 4.1: Comparing training and validation loss and accuracy for Simple CNN 72 × 72.

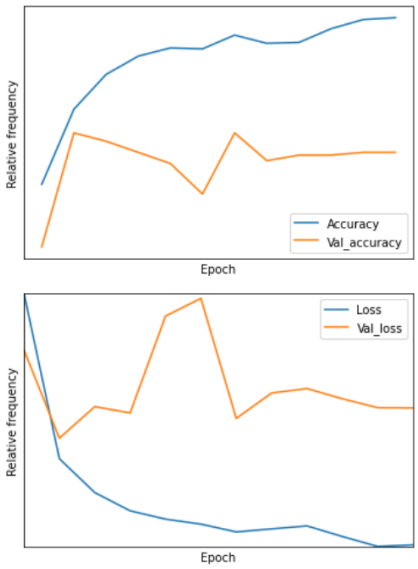

Figure 4.2: Comparing training and validation loss (low is better) and accuracy (high is better) for ResNet 108 × 192.

the ResNet model (right), the validation loss stopped improving already after 2 epochs, while the training loss kept improving. This suggests overfitting. The Simple CNN model had a steadier decrease of loss before it stopped improving after 7 epochs. This can be a result of the small dataset and low complexity. Though overfitting can be a significant problem for some tasks, there is a limit to how much harm it can do for this transition, due to most frames in the transition being very similar. We see that the models only misclassify the earliest frames, while all still hitting the rest. Alternatives to help solve overfitting are to stop earlier or choose the best weights.

Figure 4.3: The logos that ResNet 108 × 192 misclassifies.

When we look at the logos that are missed, most are within the first 5 frames of a team logo transition, and we notice that these team logos are not present in the training set. This can suggest overfitting due to the limited data. In Figure 4.3, we show the logo frames that are misclassified by ResNet 108 × 192. To tackle the insufficient data, we make a synthetic dataset in order to train the models to recognize all the team logos. We will only focus on a few models, and the results are presented in Table 4.4. For the Simple CNN, we choose 72 × 72 × 3, which have the best F1-score for this architecture. We also include 108 × 192 with RGB and grayscale, and

input shape $27 \times 48 \times 3$, to compare different size and color input. For our VGG model, we choose $72 \times 72 \times 3$, having the overall best F1-score, and 108×192 with both color inputs. For ResNet we go further with the models trained without the fine-tune training.

Model	Input	Precision	Recall	F1 Score
VGG inspired	$108 \times 192 \times 3$	**0.9917**	0.9958	**0.9938**
VGG inspired	$108 \times 192 \times 1$	**0.9915**	0.975	0.9832
VGG inspired	$72 \times 72 \times 3$	0.9836	**1.000**	**0.9917**
Simple CNN	$108 \times 192 \times 3$	0.8633	**1.000**	0.9266
Simple CNN	$72 \times 72 \times 3$	0.9189	0.9917	0.9539
Simple CNN	$27 \times 48 \times 3$	0.8852	0.9958	0.9373
Simple CNN	$108 \times 192 \times 1$	0.7539	0.9958	0.8582
ResNet (Normal)	$108 \times 192 \times 3$	0.9796	1.0	0.9897
ResNet (Normal)	$54 \times 96 \times 3$	0.9754	0.9917	0.9835

Table 4.4: Results (validation) from further training on the dataset supplemented with synthetic logo frames.

From the results in Table 4.4, we can see that training with the original training set together with our synthetic logo images led to an increase of recall, and worked in getting the models to recognize the team logos not encountered during training, like the images in 4.3. We also see a decrease in precision for all models. More backgrounds are now misclassified. Described in 4.1.4, we see that our dataset of SoccerNet logo frames contains too few backgrounds, and we assume this is the case for this dataset as well. By correcting this, the use of our synthetic set could have performed better. Because all of the frames later in the transition are correctly classified, we do not believe the trade-off is worth it in this case and decide to not go forward with these models.

Feature extractor	Input	Precision	Recall	F1 Score
Simple CNN	$108 \times 192 \times 3$	**1.0000**	**1.0000**	**1.0000**
Simple CNN	$72 \times 72 \times 3$	**1.0000**	**1.0000**	**1.0000**
Simple CNN	$27 \times 48 \times 3$	**1.0000**	0.9917	0.9958
Simple CNN	$108 \times 192 \times 1$	**1.0000**	**1.0000**	**1.0000**
Simple CNN	$72 \times 72 \times 1$	**1.0000**	**1.0000**	**1.0000**
Simple CNN	$27 \times 48 \times 3$	**1.0000**	**1.0000**	**1.0000**
VGG16	$108 \times 192 \times 3$	**1.0000**	0.9958	0.9979
VGG16	$72 \times 72 \times 3$	0.9367	0.8625	0.8980
VGG16 (RBF)	$108 \times 192 \times 3$	**1.0000**	0.9500	0.9744

Table 4.5: Validation results on the Eliteserien dataset for the SVM.

The results for the SVM on the Eliteserien dataset are presented in Table 4.5. To find the best model for the Eliteserien dataset, we tested on different input sizes and feature extractors. We start by looking at the results using the simple CNN as a feature extractor. As we can see from

the results, all the models perform extraordinarily well on the validation set. The interesting part about the results is that, with regards to the Simple CNN model, all the models have a perfect score except the RGB 27x48 model that scores 0.9917 on recall. The grayscale model with the same input size actually outperforms the model with RGB. Upon further inspection, we see that it classifies 2 very small logos from the first frames of the transitions, both with a close-up background with dark and blue colors. A theory is that the logo blends too much into the background and that the RGB version of this model relies more on the colors vs the grayscale being able to find the features.

We further inspect the VGG16 results as a feature extractor for Eliteserien as shown in Table 4.5. We see that the VGG16 performs better at higher resolution, and we see that the linear kernel actually outperforms the RBF kernel, so this indicates that the Eliteserien dataset is linearly separable and that an RBF kernel probably overfits the separation of the classes towards the training set. Upon further analysis, we see that the model using the RBF kernel struggles with small logos with no blue transition border around the logo, and we also notice that all the logos it miss classifies are closeup camera shots with either players, coaches, or the crowd covering up the background of the small logo, where it blends in with the background. The same pattern of miss classifications applies to the VGG16 72x72 model, but this model actually also miss classifies bigger logo appearances containing the blue transition fade of Eliteserien.

Model	Input	Precision	Recall	F1 Score
VGG inspired	$54 \times 96 \times 1$	**1.0000**	**1.0000**	**1.0000**
SVM (Simple CNN)	$108 \times 192 \times 1$	**1.0000**	0.9955	0.9978
VGG inspired	$72 \times 72 \times 3$	**1.0000**	0.991	0.9955
SVM (Simple CNN)	$27 \times 48 \times 3$	**1.0000**	0.9776	0.9887
SVM (Simple CNN)	$72 \times 72 \times 1$	0.9865	0.9865	0.9865
VGG inspired	$27 \times 48 \times 3$	**1.0000**	0.9731	0.9864
SVM (Simple CNN)	$72 \times 72 \times 3$	**1.0000**	0.9731	0.9864
Simple CNN	$72 \times 72 \times 3$	0.9954	0.9731	0.9841
ResNet	$54 \times 96 \times 3$	0.9954	0.9686	0.9818
VGG inspired	$108 \times 192 \times 3$	0.9954	0.9686	0.9818

Table 4.6: Best 10 results on the Eliteserien logo frame test set, based on F1-score.

The final results on the test set is presented in Table 4.6, and the full table is available in the appendix A.1. We see great performance on the test set, with the VGG inspired CNN model using grayscale $54 \times 96 \times 1$ input reaching 100% recall and precision. The F1-score increase by 1.7% from the validation set. We see that the SVM performs well on this. We notice many models with a significant decrease in performance, especially on the recall. This may be due to overfitting, as discussed earlier, and can be seen in the graphs of Figure 4.2 and 4.1. The models still hit more than 15 frames, meaning they will still perform well as part of the logo transition

module. The dataset is also very small and includes some team logos not present in the training set, and even one misclassified logo frame will have a significant impact on the recall.

When looking at how the models perform on the full clips from Eliteserien, we see that all models perform well enough to find all logos we tested on without any false positives. We tested on 50 clips. This proves that this simple strategy can work great. In the next section, we show how the models perform on a substantially bigger logo frame dataset before we test the module on full soccer matches.

4.1.3 SoccerNet Premier League 2016/2017

We now move on to the bigger dataset of SoccerNet PL16/17. This dataset will give us an indication of how our strategy translates to a more diverse logo composition, as well as much more diverse backgrounds. The volume will also give us more trustworthy results. It also provides us with the opportunity to run our final tests on full soccer matches.

All models are initialized and trained as discussed in Section 3.4.3, with glorot uniform weight initialization for the Simple CNN and VGG inspired model. With ResNet, we use transfer learning instead, meaning we initialized to the pre-trained ImageNet weights. We use a learning rate of 0.001 and reduce the learning rate on plateau with a patience of 7, and early stopping with a patience of 10. For ResNet, we also try fine-tune training, by first training the dense network only, then the whole network with a lower learning rate.

Input	Precision	Recall	F1 Score
$144 \times 256 \times 3$	0.9894	**0.9867**	**0.9881**
$108 \times 192 \times 3$	0.9911	0.9777	0.9843
$54 \times 96 \times 3$	0.9825	0.9408	0.9612
$72 \times 72 \times 3$	0.9668	0.9540	0.9604
$27 \times 48 \times 3$	0.9391	0.9186	0.9287
$144 \times 256 \times 1$	0.9937	0.9764	0.9850
$108 \times 192 \times 1$	**0.9957**	0.9641	0.9796
$54 \times 96 \times 1$	0.9759	0.9445	0.9599
$72 \times 72 \times 1$	0.9757	0.9530	0.9642
$27 \times 48 \times 1$	0.9414	0.9271	0.9342

Table 4.7: Simple CNN results for the Simple CNN on the SoccerNet validation set. There is a notable relation between the input size and results. The grayscale 108×192 has the best precision, but the recall of the logo class is lower.

This dataset is much more diverse and complex, with five different logotypes. We notice that the results differentiate much more than on the Eliteserien dataset across the different input sizes. For the Simple CNN models and the VGG-inspired models, we see that there is a direct relation

Input	Simple	Goal	PL1	PL2	Trophy
144 × 256 × 3	**0.9543**	0.9923	0.9921	**0.9997**	**0.9959**
108 × 192 × 3	0.9323	**1.0000**	0.9923	0.9991	0.9954
54 × 96 × 3	0.8124	0.9898	0.9892	0.9986	0.9921
72 × 72 × 3	0.8528	0.9949	0.9751	0.9908	0.9876
27 × 48 × 3	0.7386	0.9745	0.9569	0.9885	0.9739
144 × 256 × 1	0.9262	0.9950	0.9976	0.9994	0.9957
108 × 192 × 1	0.8963	0.9898	**0.9981**	0.9992	0.9954
54 × 96 × 1	0.8234	0.9949	0.9864	0.9983	0.9862
72 × 72 × 1	0.8572	0.9898	0.9868	0.9941	0.9852
27 × 48 × 1	0.7649	0.9821	0.9605	0.9832	0.9761

Table 4.8: Simple CNN recall on the SoccerNet PL16/17 logos in the validation set. The types are shown in Figure 3.4.

between the F1-score and the input resolution when we group by color mode (RGB and grayscale), which can be seen in Tables 4.7 and 4.9.

Figure 4.4: Some of the logo frames that is predicted wrong. There is very little contrast between the logo and the background, as well as it is very small at this stage of the transition.

It is especially one logotype that stands out, the Simple logo shown in Figure 4.4. In Table 4.12 we see the best recall achieved on this type is 0.9710 by RGB input with resolution 108 × 192 on the ResNet model, relatively low compared to the perfect recall on PL2 and Goal types and the almost perfect score for the PL1 and Trophy types. We also see in the same table that with input 54 × 96, it scores almost 99% for all other logotypes, but only 87.6% on the Simple. This extends to the other models as well, seen in Tables 4.8 and 4.10. Figure 4.4 shows some example logo frames of the Simple logo that the models predict wrong. One can see that it is hard to spot, even

Input	Precision	Recall	F1 Score
$144 \times 256 \times 3$	0.9954	**0.9916**	0.9935
$108 \times 192 \times 3$	0.9965	0.9821	0.9893
$54 \times 96 \times 3$	0.987	0.9869	0.987
$72 \times 72 \times 3$	0.9888	0.9818	0.9853
$27 \times 48 \times 3$	0.9687	0.9532	0.9608
$144 \times 256 \times 1$	**0.9971**	0.9911	**0.9941**
$108 \times 192 \times 1$	0.9969	0.9844	0.9906
$54 \times 96 \times 1$	0.979	0.9722	0.9756
$72 \times 72 \times 1$	0.9881	0.9836	0.9858
$27 \times 48 \times 1$	0.9629	0.9356	0.949

Table 4.9: VGG inspired model results on the validation dataset for SoccerNet validation set.

Input	Simple	Goal	PL1	PL2	Trophy
$144 \times 256 \times 3$	**0.9719**	0.9923	0.9957	0.9997	**0.9993**
$108 \times 192 \times 3$	0.9451	0.9974	0.9976	**1.0000**	0.9979
$54 \times 96 \times 3$	0.9455	**1.0000**	0.9943	**1.0000**	0.9943
$72 \times 72 \times 3$	0.9354	**1.0000**	0.9892	0.9997	0.9983
$27 \times 48 \times 3$	0.8392	0.9974	0.9694	0.9961	0.9948
$144 \times 256 \times 1$	**0.9719**	0.9949	0.9988	0.9997	0.9983
$108 \times 192 \times 1$	0.9525	0.9923	**0.9990**	0.9994	0.9979
$54 \times 96 \times 1$	0.9029	0.9974	0.9847	0.9975	0.9926
$72 \times 72 \times 1$	0.942	0.9974	0.9919	0.9969	0.9969
$27 \times 48 \times 1$	0.7926	0.9872	0.972	0.9955	0.985

Table 4.10: VGG inspired model recall on the SoccerNet PL16/17 logos in the validation set. The types are shown in Figure 3.4.

for us. This can explain why bigger resolutions perform better, as small resolutions might lose too much information around the small logo to be able to find the features needed to separate this logo from a background.

For ResNet, initializing the models with the pre-trained ImageNet weights and train with the same hyperparameters as we used with the other models described in Section 3.4.3, we get very good results on the SoccerNet dataset. The input size of 108×192 scores best on every category in Table 4.11. There seems to be no benefit of fine-tuning the pre-trained weights in the manner described in Subsection 3.4.3. We saw this for the Eliteserien dataset as well, in Section 4.1.2. We see from the results over each logotype in Table 4.12 that fine-tuning does not achieve good results on the Simple logo. Already after training the dense network only, with the pre-trained ResNet features, we see that it performs well on the other logotypes. One reason for this might be that the Simple logo is the least similar to the objects of ImageNet, and therefore the pre-trained features do not capture the information needed to perform well for this task. It

Training	Input (RGB)	Precision	Recall	F1 Score
Normal	$144 \times 256 \times 3$	0.9922	0.9909	0.9916
Normal	$108 \times 192 \times 3$	0.9958	0.9919	0.9939
Normal	$54 \times 96 \times 3$	0.9799	0.9632	0.9715
Fine-tune	$144 \times 256 \times 3$	0.9821	0.9709	0.9764
Fine-tune	$108 \times 192 \times 3$	0.9809	0.9338	0.9568
Fine-tune	$54 \times 96 \times 3$	0.9577	0.8647	0.9088
NN only	$108 \times 192 \times 3$	0.9505	0.8696	0.9082

Table 4.11: ResNet results on the validation dataset for SoccerNet validation set. We see that initializing to the pre-trained weights and train with a 0.001 learning rate performs better than using a low learning rate, as discussed in Section 3.4.3.

Training	Input	Simple	Goal	PL1	PL2	Trophy
Normal	$144 \times 256 \times 3$	0.9653	1.0	0.9945	1.0	0.9974
Normal	$108 \times 192 \times 3$	0.971	1.0	0.9971	1.0	0.9988
Normal	$54 \times 96 \times 3$	0.8761	1.0	0.9868	0.998	0.9917
Fine-tune	$144 \times 256 \times 3$	0.8989	0.9974	0.9914	0.9997	0.9902
Fine-tune	$108 \times 192 \times 3$	0.7777	1.0	0.9907	0.9994	0.9935
Fine-tune	$54 \times 96 \times 3$	0.5571	0.9923	0.9806	0.9952	0.9837
NN only	$108 \times 192 \times 3$	0.605	0.9668	0.9703	0.9896	0.9749

Table 4.12: ResNet recall on the SoccerNet PL16/17 logos in the validation set. The types are shown in Figure 3.4.

is also very small, and harder in general for all models. Going forward, we only use the model initialized to ImageNet weights and trained with a learning rate of 0.001 without freezing any layers and refer to it as ResNet.

The SVM results for the SoccerNet dataset are presented in Table 4.13. Here we also apply the same concept as for Eliteserien as discussed above. As we can see from figure res the SVM starts to perform poorly when it faces more complicated logotypes and a bigger dataset. We start to inspect what happens during training for the top three models with the best scores as shown in Table 4.13, to see if we can improve them. Upon further inspection we notice one thing both the linear kernel models have in common, it seems during the grid search the scores start to rise in parallel with the learning rate. This gives us information that the models probably have not been trained long enough, therefore we will try training these 3 models with a higher number of max iterations while keeping the best scoring parameters. The models start to show poor results when increasing the number of iteration, so when then assume that the reason behind this is that the model never converges so the good/bad results are more random than an actual good hyperplane to separate the classes. Given enough computing power and time the SVM could maybe find a good fit, or there could simply be that the classes cannot be separated using these feature

extractors for the SVM model. Based on the results on the SoccerNet dataset, computational cost, and execution time of the SVM compared to the CNN we choose to not pursue this model any further.

Feature extractor	Input	Precision	Recall	F1 Score
Simple CNN	$108 \times 192 \times 3$	**0.9635**	0.6807	**0.7978**
VGG16	$108 \times 192 \times 3$	0.5944	0.6376	0.6016
ResNet-RBF	$108 \times 192 \times 3$	0.4278	**0.9979**	0.5989
Simple CNN	$27 \times 48 \times 1$	0.4459	0.4430	0.4445
Simple CNN	$27 \times 48 \times 3$	0.2808	0.3477	0.3106

Table 4.13: Top 5 SVM scores on the SoccerNet PL16/17 logos in the validation set.

During training, the aim has been good precision and recall for the validation set. We have a lot of good results, but they do not necessarily generalize to new data. The problem with these results on the logo validation dataset is that we do not know how many consecutive frames are misclassified for both classes. To test some of the suited classifiers, we will run the logo detection module on whole matches to measure the performance on a real case scenario and with the whole logo detection module. This will also give us a better comparison to the total execution time performance of the different models because the values for the window size, stride, and frame rate parameters in the logo transition detection can make a significant difference.

4.1.4 Testing the logo detection module

The frame logo detection model is going to be used to classify frames in video clips in order to find logo transitions. To further analyze these models, we will run some of the best models on full matches from the validation set of the Premier League 16/17 SoccerNet dataset, as this contains annotations we can use to gather objective quantitative data and make an objective evaluation of the technical performance. For each model, we use FFmpeg 3.3.2 to extract the video in the target shape of each model. We will use a grid search for finding the best suitable window size and logo frame requirement for a logo transition to be predicted. From these, we can calculate the frame rate and stride to make the module more efficient. We will analyze the results to find the weaknesses in the context of the full module, and see if adjustments are needed. After this, we will run the module on the full matches in the test set with the final frame logo detection models with their respective configurations. The module is shown in Figure 3.8

We use the original frame rate of 25 fps on the low-quality videos in the SoccerNet Premier League 16/17 validation dataset, and a stride of 1. Each video is extracted with the target shape of the respective model's input shape using FFmpeg [73]. We first tested our VGG inspired model with input 108x192 as the logo frame classifier. Its best results are with a window

size of 13, a stride of 1, and the requirement of all of the frames being classified as a logo. The model finds 394 true positives, 22 false positives, and 0 false negatives. When looking closer at the false positives, we find that 4 of them are a logo. This means that this model scores a 95.67% precision on the validation set. We proceed to test a few more models and report the best results according to the F1-score in Table 4.14.

Classifier	Input	FN	FP	TP	Recall	Precision
ResNet	$144 \times 256 \times 3$	**0**	33	398	**1**	0,9234
ResNet	$108 \times 192 \times 3$	**0**	22	398	**1**	0,9476
Simple CNN	$108 \times 192 \times 3$	**0**	47	398	**1**	0,8944
Simple CNN	$108 \times 192 \times 1$	2	38	396	0.9950	0,9124
VGG inspired	$144 \times 256 \times 3$	**0**	8	398	**1**	**0,9803**
VGG inspired	$108 \times 192 \times 3$	**0**	18	398	**1**	0,9567
VGG inspired	$108 \times 192 \times 1$	**0**	18	398	**1**	0,9567

Table 4.14: Best results using the F1-score for our first logo transition detection test on the full validation set matches in SoccerNet PL16/17 for classifiers trained on the initial training set. We see very good recall, but there seems to be too many false hits on frame level, resulting in false logo transition predictions.

We further investigate the results and see that it is the Simple PL logo transition that is most commonly miss classified. This correlates with the results on the frame logo validation dataset reported earlier in this Section 4.1.3. By looking at the background images that are consecutively miss classified, we see some common features. Many of them are close-ups, where either the player might be similar to the trophy logo, or the background colors contain similarities with for example the white fade transition found in the bigger Premier League logos such as a shade from the stadium. We also see some of them containing the white goal net or something similar to a grid. From analyzing the results, we see that it is the Simple PL logo that has the most false-negative frames.

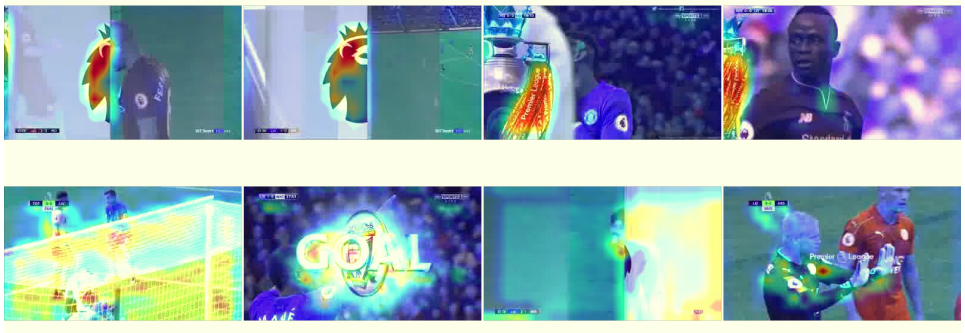

Figure 4.5: ResNet RGB 108×192 heatmap using Grad-CAM [59]. Warm colors signifies more activations.

In Figure 4.5, we have used Grad-CAM [59], implemented with Keras [16], to visualize what region of the frame gets activated when we classify using ResNet with RGB 108×192 input. These are all predicted correctly, but it is interesting to see that the two frames containing the Simple logotype (lower left and lower right image), do not focus on the actual logo. A possible explanation for this is that when the model encounters hard samples like in Figure 4.4, it adjusts to the surrounding features. This could be one of the reasons that we see backgrounds with similar features get wrongly classified, especially if the training set does not contain enough background frames that counter this learning.

This points to that the previous frame logo training set might not contain enough background images. In a real case, as this test tries to mimic, background frames are a much bigger portion of the frames (see Table 3.5), and our dataset should reflect that. To make the datasets more complete, we supplement with 7812 random background images to the premier league dataset. To make it even more robust, we add more edge case samples found by classifying all training video frames and sample over 6000 extra frames that were misclassified as a logo by either ResNet or VGG inspired with RGB input and 108×192 resolution. We also find 954 logo frames from transitions that have wrongly been annotated as abrupt or smooth and added them to the logos. The hypothesis is our models will improve by encounter more of these hard to classify backgrounds during training, and hope it leads to better predictions during inference. We only used videos from the training dataset for this. The set, Train Medium, now contains a total of 43260 background images and 23194 logo images. All frames are looked over manually. It is important to note that the extraction of hard samples can give a bias to the used models.

We choose to go forward with the best performing models and some models for the purpose of exploration and comparison. For the selected models, we load the already trained weights for each respective model and train with a learning rate of 0.001. We use early stopping and reduce learning rate on plateau as previously described in Section 3.4.3. The results on the same validation set are reported in Table 4.15. The results for the module on full matches on the validation set are in Table 4.16

We compare the recall and weighted F1 score in Table 4.15. For the most part, we see that the recall for the background increases, while the recall for the logo decreases. This applies to almost all the models, except for the VGG inspired $108 \times 192 \times 3$, which was used to extract the hard background samples. Even though the other model used for extraction had decreased performance (ResNet $108 \times 192 \times 3$), we believe this increase is due to the bias of extracting hard samples from this model. The weighted F1 score decreases, but it is important to remember that the logo class is over-represented in our dataset compared to what the ratio is in a full game or the expected video clip input of our system. While the increase in background recall is small, there are almost 75 times more background frames than logo frames for the system's input, making the impact of the improvement bigger than it may look like. Even though the recall for the logo decreases by a substantial amount, we hope it will still be adequate to

Classifier	Input	Before			After		
		l.rec.	b.rec	W-F1	l.rec.	b.rec.	W-F1
ResNet	$144 \times 256 \times 3$	0.991	0.994	0.993	0.985	0.998	0.993
ResNet	$108 \times 192 \times 3$	0.992	**0.997**	**0.995**	0.988	0.999	0.992
ResNet	$54 \times 96 \times 3$	0.963	0.985	0.976	0.935	0.999	0.972
Simple CNN	$144 \times 256 \times 3$	0.987	0.992	0.990	0.955	**1.000**	0.980
Simple CNN	$108 \times 192 \times 3$	0.978	0.993	0.987	0.949	**1.000**	0.978
Simple CNN	$72 \times 72 \times 3$	0.954	0.976	0.966	0.913	0.996	0.961
VGG inspired	$144 \times 256 \times 3$	0.992	**0.997**	**0.995**	0.987	**1.000**	**0.994**
VGG inspired	$108 \times 192 \times 3$	**0.997**	0.982	0.989	**1.000**	0.983	0.991
VGG inspired	$54 \times 96 \times 3$	0.987	0.990	0.989	0.979	0.995	0.988
VGG inspired	$72 \times 72 \times 3$	0.989	0.982	0.985	0.998	0.973	0.985
VGG inspired	$144 \times 256 \times 1$	0.991	0.999	0.995	0.988	0.999	**0.994**
VGG inspired	$72 \times 72 \times 1$	0.984	0.991	0.988	0.971	0.997	0.986

Table 4.15: Comparison of the results on the validation frame dataset before and after further training on the Train Medium dataset.

make our transition module separate the two classes accurately.

In Figure 4.6, we can see that the heatmaps produced using Grad-CAM [59] with RGB 108×192 input, before (above) and after (below) the extra training. These background frames was previously predicted wrong. We see that there are less activations, and may suggest that the model have learned better features to separate the classes.

Figure 4.6: Heatmap from three of the layers of the VGG inspired model with RGB 108×192 input, before and after the extra training. These background frames was previously predicted wrong.

The validation set results for the classifiers trained on the bigger dataset, reported in Table 4.16, suggests that we were right that the training set was insufficient, and that the models needed to encounter more backgrounds

during training. The models are still able to find all the logos, and the precision has improved a lot. ResNet with RGB 108×192 went from 22 wrongly predicted transitions to just 2. It is also surprising that the VGG inspired with RGB 54×96 input scores a perfect score when comparing to the results reported on the frame logo set in Table 4.9.

Classifier	Input	FN	FP	TP	Rec.	Prec.	l/ws
ResNet	$144 \times 256 \times 3$	**0**	8	398	**1**	0,980	13/13
ResNet	$108 \times 192 \times 3$	**0**	1	398	**1**	0,997	8/8
ResNet	$54 \times 96 \times 3$	**0**	**0**	398	**1**	**1**	14/14
Simple CNN	$144 \times 256 \times 3$	**0**	8	398	**1**	0,980	10/10
Simple CNN	$108 \times 192 \times 3$	**0**	4	398	**1**	0,990	12/13
Simple CNN	$72 \times 72 \times 3$	**0**	17	398	**1**	0,959	13/13
VGG inspired	$144 \times 256 \times 3$	**0**	**0**	398	**1**	**1**	10/11
VGG inspired	$108 \times 192 \times 3$	**0**	1	398	**1**	0,997	13/13
VGG inspired	$54 \times 96 \times 3$	**0**	8	398	**1**	0,980	13/13
VGG inspired	$144 \times 256 \times 1$	**0**	3	398	**1**	0,993	12/12
VGG inspired	$72 \times 72 \times 1$	**0**	2	398	**1**	0,995	12/12

Table 4.16: Best results for each logo transition detection after training the classifiers on the medium extended training set (Train Medium). l/ws - logo frames out of window size.

Because of this significant improvement, we will further expand the training dataset, report the results on the validation, and finally report the final results on the test set using the same window size and requirement as the best for the validation. Due to the time-consuming extraction and assembly of the extra frames, as well as training time, we only test this on some models to further analyze the effects of a bigger background training set.

The results after training on Train Max is reported in Table 4.17. For most of the models with the same input, we see a decrease in performance. ResNet with RGB 144×256 input gets quite a significant improvement and ResNet with RGB 108×192 and VGG inspired with grayscale 144×256 both achieves the same result as before. The rest has worse performance with the same settings as in Table 4.16, and compared to their respective best settings. This can suggest that the initial training set lacked hard background samples rather than quantity being the main problem.

For the final, test we will use the models trained on the medium extended set instead of the full set. We use the same window size and logo frame requirement as the best results on the validation test from Table 4.16, to see how it generalizes to a less biased set.

The final results for the module are presented in Table 4.18. The results are very good, with the best model, ResNet with RGB input and a resolution of 144×256, scoring an impressive F1 score being 0.9946, with zero missed logos and only 2 false positives. This model performed well on the validation set as well, The runner-ups got an F1 score of 0.9918 using the classifiers VGG ($144 \times 256 \times 3$) and ResNet ($144 \times 256 \times 3$). All models

Classifier	Input	Train Medium Rec.	Train Medium Prec.	Train Max Rec.	Train Max Prec
ResNet	144 × 256 × 3	**1.0000**	0,980	**1.000**	**1.000**
ResNet	108 × 192 × 3	**1.000**	**0,997**	**1.000**	**0,997**
ResNet	54 × 96 × 3	**1.000**	**1**	**1.000**	0.993
Simple CNN	144 × 256 × 3	**1.000**	**0,980**	**1.000**	**0,980**
Simple CNN	108 × 192 × 3	**1.000**	**0,990**	**1.000**	0.985
Simple CNN	72 × 72 × 3	**1.000**	0,959	**1**	**0.966**
VGG inspired	144 × 256 × 3	**1.000**	**1**	**1.000**	0.997
VGG inspired	108 × 192 × 3	**1.000**	**0.997**	**1.000**	0.988
VGG inspired	54 × 96 × 3	**1.000**	0,980	**1.000**	**0.983**
VGG inspired	144 × 256 × 1	**1.000**	**0,993**	**1.000**	**0,993**
VGG inspired	72 × 72 × 1	**1.000**	**0,995**	**1.000**	0.983

Table 4.17: Best results for logo detection module after training the classifiers on the medium extended training set (Train Medium) versus trained on the Train Max dataset. Bold text signifies best recall/precision within row.

performed very well on the validation set while we see a much higher variance here. This is expected, as we used prefixed parameters for the window size and frame requirement. The most surprising results are the bad performance using the VGG inspired model with 54 × 96 × 3, with 120 false negatives. This may be due to the module parameters not working on this set, or that the model overfit.

Classifier	Input	FN	FP	TP	Rec.	Prec.	F1	l/ws
ResNet	144 × 256 × 3	5	1	363	0.986	0.997	0.993	13/13
ResNet	108 × 192 × 3	**2**	**0**	366	**0.995**	**1.000**	**0.997**	8/8
ResNet	54 × 96 × 3	20	4	348	0.946	0.989	0.967	14/14
S. CNN	144 × 256 × 3	8	1	360	0.978	0.997	0.988	10/10
S. CNN	108 × 192 × 3	7	8	359	0.981	0.978	0.980	12/13
S. CNN	72 × 72 × 3	26	7	342	0.929	0.980	0.953	13/13
VGG	144 × 256 × 3	3	3	365	0.992	0.992	0.992	10/11
VGG	108 × 192 × 3	21	2	347	0.943	0.994	0.968	13/13
VGG	54 × 96 × 3	**2**	120	366	0.995	0.753	0.857	13/13
VGG	144 × 256 × 1	8	0	360	0.978	1.000	0.989	12/12
VGG	72 × 72 × 1	13	6	355	0.965	0.983	0.974	12/12

Table 4.18: Final test results classifiers trained on Train Medium dataset, and evaluated using the same window size and logo frame requirement as the best results on the validation test from Table 4.16.

The results are overall good, and we have models that are capable of finding most logos with few false negatives. If we consider that the modules' input when part of our full highlight clipping system will have significantly fewer background frames, the false negatives are a much

smaller problem. We see that even with more complex logos, our strategy works well.

The computational efficiency is presented in Table 4.19, and was measured as described in Section 3.4.3. We can see that even the slowest CNN models perform at a high fps rate. According to statistics provided by FIFA, there were on average 2.6 goals per match in the FIFA world cup 2018 [26]. In the naive solution of predicting all input frames (2 minutes adding up to 300 frames), the slowest CNN model, ResNet with RGB input of 144×256 resolution, would use a little over 2.5 second per game, while the Simple CNN with RGB input of 144×256 resolution would use 0.23 seconds, using the results from the module test in Table 4.18. Theoretically, the slowest CNN model could use 0.37 seconds each game by only predicting every 13th frame, and predict 12 frames on either side if it finds a frame, while still outputting the same result. $\frac{((3000/13)+24)\times 2.6}{1,798} = 0.37s$.

4.1.5 Computational cost

model	input	fps
ResNet	$144 \times 256 \times 3$	1,798
ResNet	$108 \times 192 \times 3$	3,117
ResNet	$54 \times 96 \times 3$	12,295
Simple CNN	$144 \times 256 \times 3$	340,936
Simple CNN	$108 \times 192 \times 3$	341,897
Simple CNN	$72 \times 72 \times 3$	339,099
VGG inspired	$144 \times 256 \times 3$	94,169
VGG inspired	$108 \times 192 \times 3$	65,281
VGG inspired	$54 \times 96 \times 3$	86,594
VGG inspired	$144 \times 256 \times 1$	96,428
VGG inspired	$72 \times 72 \times 1$	85,722
SVM (Simple CNN)	$108 \times 192 \times 3$	179
SVM (Simple CNN)	$27 \times 48 \times 3$	14023
SVM (Simple CNN)	$72 \times 72 \times 3$	1103
SVM (Simple CNN)	$72 \times 72 \times 1$	883
SVM (VGG16)	$108 \times 192 \times 3$	442
SVM (VGG16)	$72 \times 72 \times 3$	3172

Table 4.19: Execution times measured on the DGX2 server 3.3.1. All models was evaluated using Eliteserien dataset.

Even though the execution cost for the different CNN models is relatively high, this means that we, in practice, do not need to compromise in order to get good performance, as the computational cost is negligible for a high-end system such as DGX-2. It is still of interest for other use cases where processing power is more limited.

The SVMs perform slower, with a high increase of computational cost with higher resolutions. An important aspect of SVMs computational cost

is the number of parameters it takes in as input. Using the Simple CNN with an image resolution of $108 \times 192 \times 3$ outputs over 36,000 parameters, leading to a slow execution time for the SVM, and high memory usage. With resolution $72 \times 72 \times 3$, there are 8,192 features. The SVM is therefore not suitable with high resolution for the Simple CNN. Even though the SVM performs well on the Eliteserien dataset, its computational costs are too high, compared to the good performance of many of our tested CNNs on the same set, with much lower computational cost.

4.2 Shot boundary detection module

TransNetV2 is a state-of-the-art shot boundary detection model and has achieved very good performance on benchmark datasets, such as ClipShots [70], RAI [11], and BBC [10]. We will now proceed to test it on our shot boundary detection dataset, and SoccerNet [18], to see how it performs on soccer clips. We will evaluate based on the abrupt and gradual transitions, as these are the transitions our module is designed to process. We will evaluate the pre-trained weights trained on ClipShots and IACC.3 [64] [70] [8], and is further referenced to as TransNetV2 pre-trained. We will also train our model using our SoccerNet SBD dataset, and we reference this as TransNetV2 SoccerNet. TransNetV2 is introduced in Section 2.4.3 and described in Section 3.5.1.

4.2.1 Training TransNetV2

We train our weights from scratch to compare to the pre-trained weights provided with TransNetV2 [64]. We describe our dataset and method in Section 3.5.1. We start with the subset of the SoccerNet SBD dataset, SBD PL16/17, then the full SoccerNet SBD dataset, before we finally run an evaluation on the full matches of the full test set from SoccerNet. The datasets are described in Section 3.1.4.

For our preliminary experiment, we run 50 epochs of training on SBD PL16/17. The best results after 20 epochs. For our first evaluation, we compare results on the SBD PL16/17. We use our trained TransNetV2 and compare it to the pre-trained model. We use tolerance $\delta = 4$ and $\delta = 24$ (1 second), meaning the prediction must be at least within a distance of 2 and 12 frames, respectively, to be considered a true positive. We start with $\delta = 4$ for both models. This yields good results, meaning that we can assume that most annotations are accurately annotated. The results for the models are presented in Table 4.20.

Looking at the false positives manually, we notice that some are correct. This is especially apparent with logos. We notice that the logos are often detected at the end of a transition. This may be due to the more aggressive change in the fade-out as opposed to the fade-in. Some logos are also very small throughout the transition before it expands in the last 3 - 4 frames. We also notice that some gradual transitions are predicted at the start of the transition, and not in the middle like the labels are. Some annotations may

metric	Trained on SoccerNet $\delta = 4$	$\delta = 24$	Pre-trained $\delta = 4$	$\delta = 24$
Precision	93.80%	96.46%	97.85%	99.05%
Recall	84.09%	86.47%	97.85%	99.05%
F1 Score	88.68%	91.19%	89.33%	90.43%

Table 4.20: TransNetV2 SoccerNet results on SBD PL16/17 validation set.

be less accurate as well, by a few frames. This means that low tolerance could be an issue. It is also possible that some transitions are not annotated at all. We continue with a tolerance δ of 24 (1 second) to see the effect. This is the tolerance used for evaluation by Deliège et al. [18] on the models shown in Table 2.1.

We see an increase of over 1% for all metrics when using the higher tolerance. This suggests that the accuracy of some of the annotations is off by more than two frames. Some of the increases can also be previous actual false positives considered as true positives. Because the recall is at a high level, the increase is more likely due to actual transitions being detected, because if the true positive is found, the previous false positive for tolerance $\delta = 4$, will still be considered a false positive. There also are not very many false negatives, to begin with. We, therefore, consider a tolerance δ of 24 to reflect an accurate description of the performance.

We compare the scores for each transition type in Table 4.21 with a tolerance δ of 4. We can see that it has very bad performance on the logo transitions. The low precision can suggest that it is a result of inaccuracies between the predicted frame and the annotated point.

weights	metric	All	Abrupt	Gradual	Logo
SoccerNet	Precision	93.80%	96.93%	**91.62%**	21.11%
	Recall	84.09%	**98.26%**	**95.35%**	**4.61%**
	F1 Score	88.68%	97.59%	**93.45%**	7.57%
Pre-trained	Precision	**97.85%**	**98.88%**	91.33%	**56.67%**
	Recall	**82.17%**	97.16%	79.65%	4.13%
	F1 Score	**89.33%**	**98.01%**	85.09%	**7.69%**

Table 4.21: Both models performance for each transition type on the SBD PL16/17 Valid dataset. The tolerance used is 4 frames.

When we use a tolerance δ of 24, the precision increases to 88% and recall to 23% for the pre-trained model. When we look at them manually, we see that most of the predictions of the logos are at the end, treating it as an abrupt transition. This is probably due to the sudden change in colors happening at the end, as the fade-in is mostly gradual, and the fade-out is more aggressive. The pre-trained model achieves 99.0% precision and 96.5 recall when we consider abrupt and gradual transitions only. This is a good result. Our manual inspection also suggests that it is frame-accurate

for abrupt and smooth transitions. With our trained TransNetV2 model, we achieve 96.6% precision and 98.0% recall.

Due to this set being part of the training for our trained model, it is biased, and we only use it as an intermediate evaluation of both models. We can see that the pre-trained model has better precision, but our trained model achieves better recall. For smooth transitions, we see that the recall of the pre-trained model is 15% lower than that of the trained model. This can suggest that training on the soccer clips makes the model significantly better at detecting them in soccer videos.

We continue training for 30 epochs on the full dataset. We choose the best model and run a test on the test set of the small 100 frame clips and the full-length matches. The results are shown in Table 4.22 and 4.23.

4.2.2 Evaluation of TransNetV2

We start by testing on the SoccerNet SBD dataset. The results are presented in Table 4.22. We can see that the pre-trained model outperforms our model on all classes except for the abrupt class. On the abrupt class with the trained model, we see a higher recall than that of the pre-trained model. The same goes when we combine abrupt and gradual transitions in one metric. In the context of the function the module has in our system, we prioritize the precision over recall, as the recall is high for both. The consequences of a false positive are more severe than the consequences of a false negative. A false negative will lead to the system making a default cut, but false positives will potentially fool the system into including/excluding scenes that it is not supposed to. Based on this, the pre-trained model is preferred for our system. It may be that hyperparameter tuning and better annotations for gradual and logo transitions can achieve even better results. Souček and Lokoč [64] notes that synthetic data boosted the performance, and in our case, it could have provided gradual transitions with exact annotations without any manual work.

weights	metric	All	Abrupt	Grad	Logo	Abr.&Grad.
SoccerNet	Prec.	95.66%	98.37%	97.66%	76.80%	98.22%
Pre-trained		95.96%	98.73%	98.69%	77.26%	98.72%
SoccerNet	Rec.	80.35%	96.63%	87.51%	33.24%	94.56%
Pre-trained		80.67%	95.58%	89.19%	36.06%	94.13%
SoccerNet	F1-score.	87.34%	97.49%	92.31%	46.40%	96.36%
Pre-trained		87.65%	97.13%	93.70%	49.17%	96.37%

Table 4.22: Comparing both models' performance for each transition type on our SoccerNet SBD test set. Valid dataset. The tolerance δ is 24 frames.

We move on to the full SoccerNet test set, containing 100 full-length games. We consider a transition correct if it is within $+/-0.5$ seconds (tolerance δ of 24) from an annotated transition. The results are shown in Table 4.23. The recall performance is in line with the previous test, but we

see poor precision scores.

Weights	Precision	Recall	F1-score
SoccerNet	45.63%	78.08%	57.60%
Pre-trained	46.88%	78.85%	58.80%

Table 4.23: Result for TransNetV2 on the SoccerNet full-length test set with a tolerance δ of 24 frames.

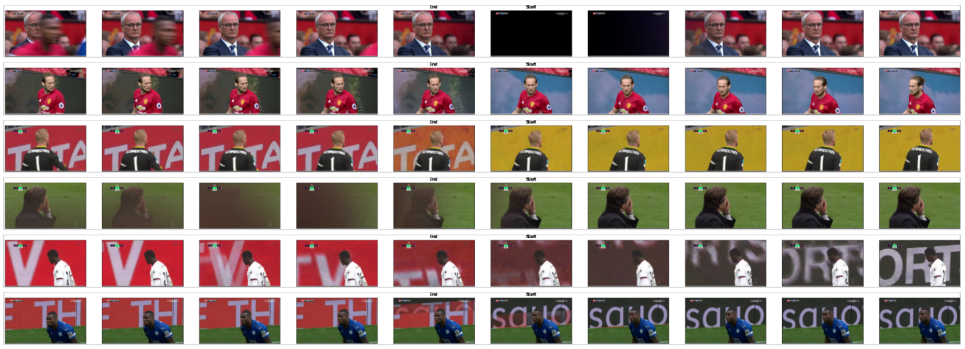

Figure 4.7: TransNetV2 model's false positives. We see close similarity to abrupt and fade transitions.

When we look closer at the results per game, we see that some achieve an almost perfect score above 90%, while others have hundreds of false positives. It may be the matches' properties, such as colors on the team jerseys, colorful advertising boards, lighting conditions, production style, and such, that makes the model detect false positives. It is also possible that the quality of some of the annotators has been poor, due to all false positives being clustered in the same matches. Due to all these false positives, we will manually look over some of the false positives and false negatives, using the pre-trained model, as this is the preferred model.

We start by looking specifically at the matches from the test set from Premier League season 2016 - 2017. Looking through a total of 861 false negatives, we only spotted 10 actual false positives. With the earlier results of 1,464 TP, 861 FP and 420 FN, we now have 2,325 TP and 10 FP. This means that the model achieved a precision of 99.57% and a recall of 84.70%. We continue to check 10 more random soccer periods from 10 different games. After looking at 1,378 more false positives, we find that only two of them are actual errors. Almost all cases of the false negatives we saw are abrupt or smooth transitions.

Looking at the false negatives, the transitions that the model misses, we see that it is mostly logos and smooth transitions. Examples can be seen in Figure 4.8. We also find abrupt transitions that seem simple.

Training TransNetV2 on the SoccerNet SBD dataset showed promising results. Our preliminary training on SBD PL16/17 dataset only showed potential for the model to perform better on smooth transitions when trained specifically on soccer clips. However, the differences disappeared

Figure 4.8: Some of the transitions the model misses. The screenshot is taken from our analyzing tool for shot boundary.

when running further training and tests on the full set. This may have been due to bias when we chose the best model, meaning it would not generalize over the whole test set. Continued training and test on the full SoccerNet SBD dataset show that both models perform well, but the pre-trained model is preferred, with better precision on abrupt and smooth transitions and a similar recall.

When evaluating the pre-trained TransNetV2 shot boundary detection system on the full SoccerNet-v2 test set matches, we found a lot of false positives. The model had a precision of less than 50%. From manual inspection on some of the transitions, we found that they were in fact transitions, and it is possible that most of the false positives are missing annotation for the other leagues as well. We only inspected a hand full of 1,300 transitions fully (a little over 5% of the false positives). The model shows good results if we assume that the dataset is missing a lot of annotated transitions. If we assume so, the model performs very well, with 99.7% precision, 89.2% recall, and 94.2% F1-score. However, we can not be sure, and we do not have enough objective data to reach a conclusion. What we can conclude, is that it achieves good recall. We also find that the models are frame-accurate, which is important for the technical standard when clipping the highlights. We conclude that the model performs at least adequate for our usage, as it is very accurate, and performs well on our SoccerNet SBD dataset seen in Table 4.22.

4.3 Final version of our system

For our full system, we will combine the logo detection and shot boundary detection module. The system takes the video frames and annotated goal as input. The modules then identify scene transitions and logo transitions. This is then fed to the clipping protocol to determine the interval for this specific goal. We describe the system in-depth in Section 3.6.

In Section 4.1, we show that the frame logo classification performs great

on the logo frame datasets, and they are capable of great performance spotting a logo transition. The best model for Eliteserien is VGG inspired with a resolution of $54 \times 96 \times 1$ (100% F1-score[1]) and ResNet with a resolution of $108 \times 192 \times 3$ (99.7% F1-score[2]) for SoccerNet. We use these models in our final system.

In Section 4.2.2, we find good results from training TransNetV2 [64] on SoccerNet [18], however, it did not outperform the pre-trained model trained on ClipShots [70] and IACC.3 [8]. We use the pre-trained model in our final system.

With the scenes and replay identified, we use a video processing module to make the highlight. The clipping protocol, in Algorithm 1, is used to decide where the highlight will start and end. We also have a version that trims the length by removing parts in between the goal and replay.

We have evaluated the objective technical standard from our experiments. To see how compelling the two different outputted highlights are, we will, in the next section, evaluate the performance of the system by analyzing the subjective data from the online survey.

4.4 Subjective evaluation of highlight clips

In this section, we evaluate the quality of our final highlight clips in relation to the existing model used today, using quantitative and qualitative subjective data gathered from our user survey. The survey is set up and distributed as described in Section 3.7. We start by going through the participants' background information to get an overview of the distribution. We then move on to the results for the highlight comparisons and discuss the results. We look at the overall results and results for different groupings of the participants.

4.4.1 General information about the participants

Upon inspection of the data, we see that 3/64 of the participants have given a score of 10 for all the clips they were asked to evaluate and no additional comments. Since these are the only participants giving an identical score for every clip, we make the assumption that these individuals have either just clicked their way through the survey without watching the clips or they have misunderstood what they were supposed to do. There is the low possibility that the individuals did not see any difference in the clips and think they all were perfect, but since their data only will up the average score of every model it does not provide any value for us and therefore we remove the data provided by these participants.

After removing some of the data as discussed above we start to process the remaining data to get a better overview of the participants taking part

[1] Results on the Eliteserien logo frame dataset

[2] Results on the full-length test set.

in our survey and be able to categorize them into different groups to see if there is any bias across the different groups.

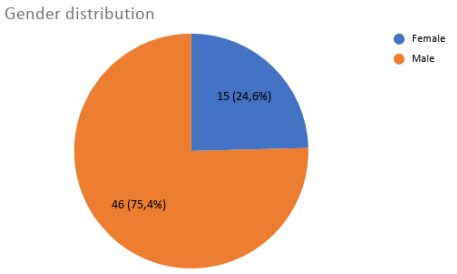

Figure 4.9: The distribution of gender.

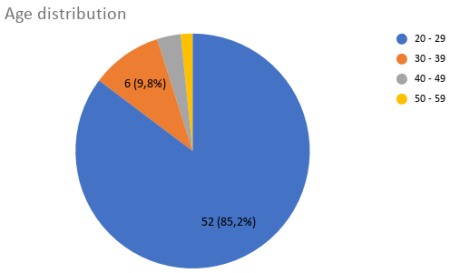

Figure 4.10: The distribution of age.

As we can see from figure 4.9 the distribution of females and males is more weighted towards males, which could lead to biased results because more females are not represented in the survey. Looking at the age distribution in 4.10 we were not able to get any participants in the age groups below 18 or over 59 which leads to the opinions of these ages groups not being represented in the results (assuming there is bias across different age groups). We also see that 82,2% participants in the survey fall under the age group 20 - 29 years old and the remaining 14,8% make up the participants between age 30-59, so the results of this survey mostly represent participants of age 20-29 and the male gender.

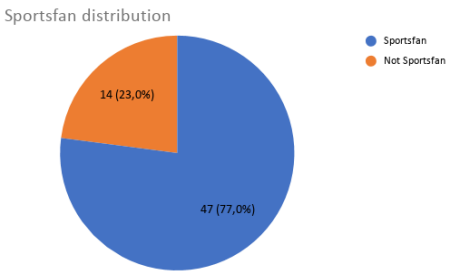

Figure 4.11: Distribution of people who consider themselves sports fans.

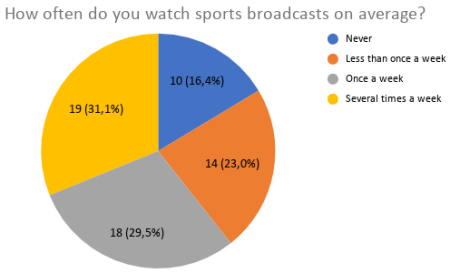

Figure 4.12: Distribution of how often the participants watch sports broadcasts on average.

As we can see from Figure 4.11 two thirds of the participants are sports fans which is a reasonable distribution given that people who are sports fans would be more interested in this type of survey and probably have a stronger foundation for evaluating the clips. In Figures 4.12, 4.13, 4.15, and 4.14, we see a fairly even distribution of how often participants watch matches and highlights which we will further in the chapter categorize into groups we think could have an impact on the results to see if there is any bias across the groups.

Most of the participants do not have any experience with video editing as shown in Figure 4.16, but 27,9% of the participant have experience

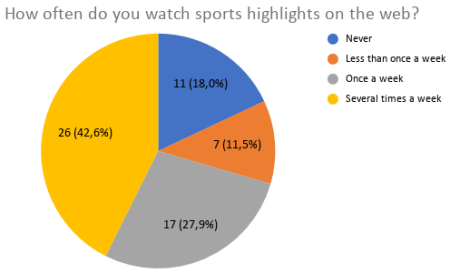

Figure 4.13: Distribution of how often the participants watch sports highlights (on web) on average.

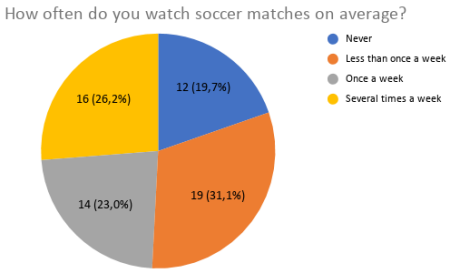

Figure 4.14: Distribution of how often the participants watch soccer matches on average.

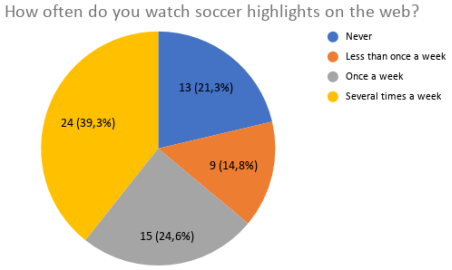

Figure 4.15: Distribution of how often the participants watch soccer highlights on average.

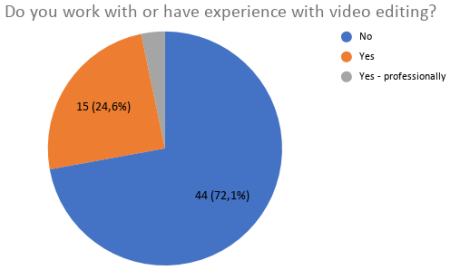

Figure 4.16: Distribution of the participants experience with video editing.

with video editing which could provide helpful data more focused on the cut themselves. Out of the 27,9% having experience with video editing, two of the participants answered they have professional experience with this which could be interesting to further inspect the opinion of these individuals, but we keep in mind that only two professionals is not enough data to represent the general opinion of this group.

4.4.2 Results

After sorting our data out, we start looking at the numbers themselves. The statistics we choose to inspect further are as follows

- Average score for a model across all the comparisons to tell us a general idea of the performance of the model.
- which model is preferred in every comparison to give us an overview of what model is preferred in each comparison.
- Standard deviation to see if there is a general agreement in the given average score or if there is strong disagreement from participant to participant.
- Median to tell us what the center of the data to be analyzed is.

The final reasoning behind the measurement choices above is that given an average is not enough due to the fact that one model could receive high scores from the participants preferring this model, while the other group preferring the other model gives a lower average score to their preferred model (therefore we also measure the preference). The standard deviation is also introduced due to the fact that even if most people prefer 1 model over the other they could be giving a very close score to the non-preferred model and be a small factor separating them. Further, we will be looking at these statics for different groups to see if there is any bias across the groups and present the data for the groups we assume to be most relevant for the evaluation of our model.

Model name	Average score	Standard deviation	Median
Our model - Short	7.40	1.98	8
Our model - Full	6.84	2.10	7
Original	5.89	2.12	6

Table 4.24: Average score for all the models across all the comparisons.

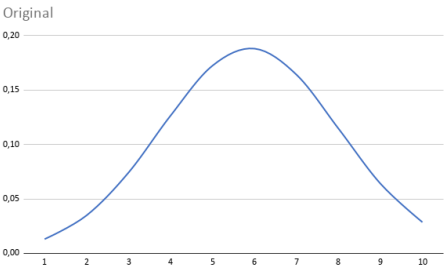

Figure 4.17: The standard deviation for the original model across all the comparisons.

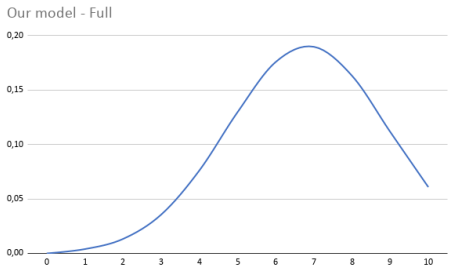

Figure 4.18: The standard deviation for Our model - Full across all the comparisons.

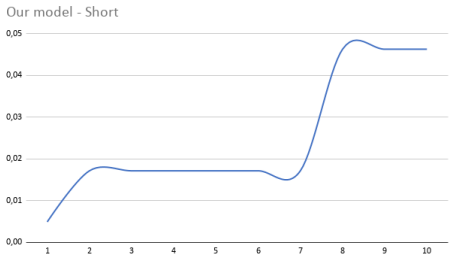

Figure 4.19: The standard deviation for Our model - Short across all the comparisons.

As we can see from Table 4.24, Our model - Short is a clear winner on average score, Our model - Full is the runner up and the original model achieves an average score of only 5.89. We further inspect the

results and plot a standard deviation graph for all the models as shown in Figures 4.174.194.18. We start by looking at the graph for the original model, here we see that the standard deviation is 2.12 so the scores will normally vary between 3.77 and 8.01. The median of 6 is fairly close to the average score which leads to an even distribution of the scores as shown in Figure 4.17. The graph for Our model - Full is fairly similar to the original except it is slightly more right-skewed which leads to a higher score on average. Now looking at the graph and table for Our model - Short we see that the median of 8 is relatively a lot higher than the average score of 7.40 which leads to a right-skewed standard deviation graph as shown in Figure 4.19. Due to the fact that our median is 8 we know that 50% of the scores are 8 or higher meaning that there are some relatively low scores bringing the average down to 7.40. Upon further inspection we see that in comparison 2 our model - Short scores 6.85 on average meaning this is the comparison that brings down the average of the model the most. Further, inspecting comparison 1 our model - Short receives an average score of 8.16 which is closer to the median meaning this is probably the comparison leading to a right-skewed graph 4.19.

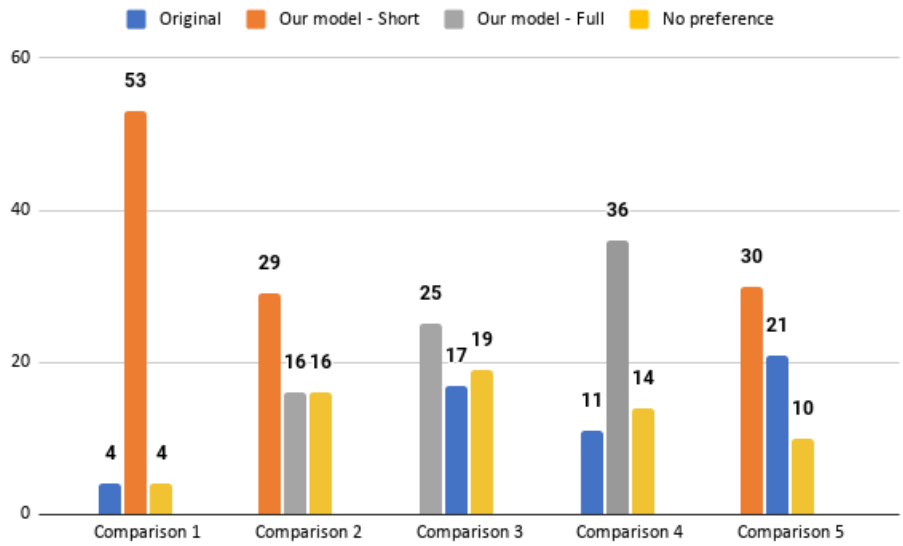

Figure 4.20: The preferred model with respect to the comparison.

Looking further into the preferences of the participants as shown in Figure 4.20 we are going to look at each individual comparison and analyze the comments to find a reasoning behind the preferences.

Comparison 1

In comparison 1, a corner goal is shown, the original model (clip 1) shows the corner being taken, the goal, full celebration, and cuts at the start of the replay. Our model - Short (clip 2) shows the corner being taken, the goal, short celebration, and the full replay. Looking at some of the comments

given by the participants:

- "The first clip seems to have more random points of start and stop. The second clip starts where the situation actually begins, and catches replays and celebration before it stops at what feels like a natural time."
- "I would say clip 1 is just the live footage, clip 2 is the highlight. Clip 2 is well cut, but could have shown a couple of seconds more of the celebration before the replay is played."
- "clip 2 was way more engaging"
- "Goal summary (replay) adds value."
- "first one feels longer and boring"
- "To much unnecessary content in clip 1."
- "I think the first one is too long. I and i miss a replay of the goal. Second one is great!"

There seems to be a consensus in the comments that the replay is almost a must to have in the clip, but there seems to be a disagreement regarding the celebration and celebration length. Some of the comments state that the original clip is boring due to the fact it is focusing too much on the celebration, while other comments stated that they wished to see more celebration in clip 2.

Comparison 2

In comparison 2, a counter-attack following a goal is shown. Our model - Full (clip 2) shows a short build-up, the goal, full celebration, and replay. Our model - Short (clip 1) shows a short build-up, the goal, short celebration, and replay. Looking at some of the comments given by the participants

- "for my feeling both are too long. one repetition of the goal is enough. also all the fans and celebrating is boring."
- "I think clip 2 is fairly good, but it shows the same clip of a celebration twice which is unnecessary."
- "Goal summary (replay) more closely after the goal seems better."
- "They seemed very similar."

The comments all seem to agree that the clips were too long or that they looked very similar. Some comments stated that they also disliked that the celebration after the goal also appeared at the end of the replay.

Comparison 3

In comparison 3, a cross ball following a goal is shown. Our model - Full (clip 1) shows two scenes before the goal, the goal, celebration, and the full replay. the Original (clip 2) model shows 3 scenes before the goal, the goal, full celebration and cuts a few second before the replay is finished. Looking at some of the comments given by the participants

- "In my opinion the second clip catches more of the situation leading up the the goal, which gives a better understanding of how the goal happened. The first seems to start a bit in the middle of a situation, which feels a bit abruptly."
- "too much additional things (trainer drinking water, etc.) Better two or three good representations of the goal and what lead to it."
- "Starting with the wide angle view (Clip 2) gives better context. Second angle for the goal summary (Clip 1) is good to have."
- "Clip 2 started a bit early"
- "Almost identical"

Some of the participants prefer clip 2 due to the fact it shows a bit more of the build-up towards the goal, while others think this is unnecessary. It also seems that some of the participants think it is too many unnecessary "celebration" scenes not actually showing the goal event and some comments state the clips look very similar.

Comparison 4

In comparison 4 a penalty kick is shown leading to a goal. The original model (clip 1) shows 1 scene of the tackle leading to the penalty, build-up, the goal, full celebration, and no replay. Our model - Full shows the build-up, the goal, full celebration, and full replay. Looking at some of the comments given by the participants

- "The second seems to be a cleaner cut for me, as it starts when the player is ready for the penalty. The first clip includes just a brief moment where the tackle happens, but if the situation leading to a penalty is decided to be included it should include much more of the situation. With just this brief clip it just contributes to making the clip more untidy."
- "again celebration and other things around feel too long. especially if you are from the opposite team. These things might be necessary but can be much shorter"
- "clip 1 missing replay of goal, clip 2 missing replay of penalty incident, would like both to be included"

We see in the comments that participants want to see a better clip of the tackle leading to the penalty and would rather have it not included as shown in clip 1 due to the fact it is so brief and seems untidy. The participants also seem to agree in the comments that the replay provides value for the highlight, but some participants expressed they would prefer it to be shorter.

Comparison 5

In comparison 5 a corner kick following a goal is shown. Our model - Short (clip 1) shows two scenes of the corner kick, the goal, shortened celebration, and full replay. the original model (clip 2) shows more of the build-up, the goal, full celebration, and no replay. Looking at some of the comments given by the participants

- "1 is better on 2 to much time is wasted for celebrating."
- "the first video here is exactly what I would expect."
- "Corner kick goal summary (replay) adds value."
- "Again, little bit weird when it goes straight to the replay. Feels a bit like fifa or a recap, so if this was during a game, it would be weird with nr 1. After the game, number 1 works"

The comments seem to agree that clip 1 containing the replay is a plus. A participant also commented that the cut before the replay in clip 1 is a little abrupt (unnatural transition). Looking at the last comment there is a possibility that some of the participants perhaps have misunderstood that these were actually supposed to be highlights posted after the game, and not during the game.

4.4.3 Grouping of participants

We will now group the participants into different groups to see if there is any bias across the groups that impacts the scores given or which model is preferred as discussed in Section 3.7.1.

Sports fans

As discussed in Section 3.7.1 we want to categorize participants who consider themselves sports fans and see how they compare to participants that do not consider themselves sports fans to see if there is any bias across the two different groups.

After grouping the participants into sports fans and non-sports fans we see that the ranking remains the same for both groups, as for the ranking given by all the participants in Figure 4.24. The interesting part for these two groups is that the group for sports fans seems to have given a significantly lower average score for all the models, than the participants in the non-sports fans group as seen in Tables 4.25 4.26. There is also a

Model name	Average score	Standard deviation	Median
Our model - Short	7.20	1.93	8
Our model - Full	6.57	1.89	7
Original	5.71	1.95	5

Table 4.25: Statistics for sports fans.

Model name	Average score	Standard deviation	Median
Our model - Short	8.07	2.03	9
Our model - Full	7.71	2.48	9
Original	6.50	2.53	6

Table 4.26: Statistics for non-sports fans.

significantly higher standard deviation for the group of non-sports fans which seems to not be the case for the group of sports fans. These results further strengthen our assumption discussed in Section 3.7.1 that sports fans have a stronger foundation of rating these clips due to the fact that they know what to look for and what they want. Also given the higher standard deviation and average score given by the non-sports fans groups could be an indication that the participants do not really see that much difference and think all the models are relatively good, and the given higher standard deviation for the non-sports fans suggests that the participants do not have a mutual agreement on what makes a good sports highlight.

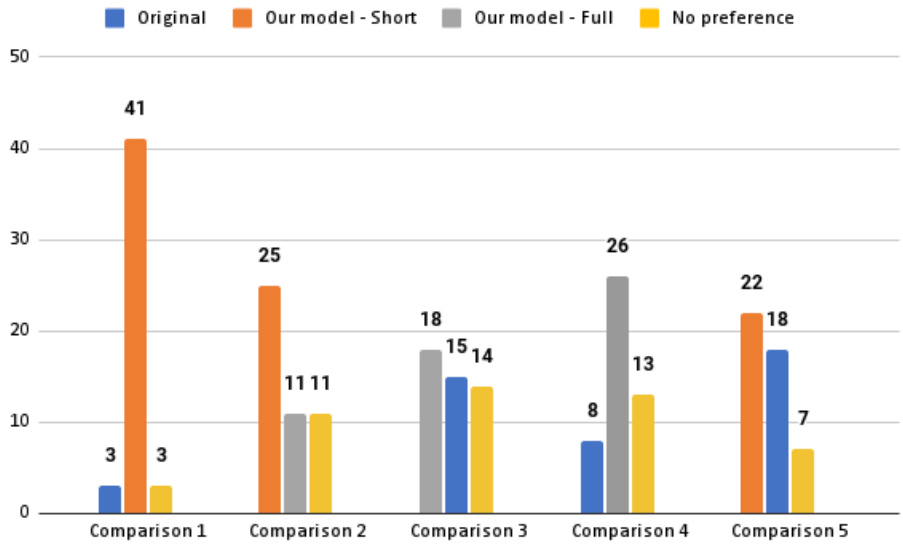

Figure 4.21: The preferred model for sports fans with respect to the comparison.

Looking in to the preferences for the two groups shown in Figures 4.22

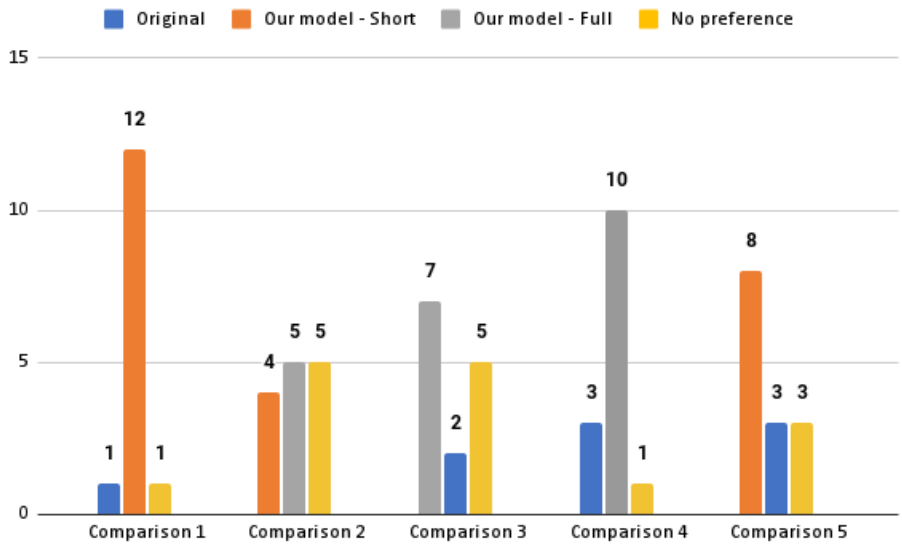

Figure 4.22: The preferred model for non-sports fans with respect to the comparison.

and 4.21. we see that for the sports fans in the majority of comparisons there is a clear preference in which model is preferred except in comparison 3 and comparison 5. For the non-sports fans, the interesting part is that in comparison 2 the number of participants having no preference is equal to the number of participants preferring Our model - Full and the number of participants preferring Our model - Short is almost equal. This gives an indication that sports fans prefer that the celebration scenes are shorter given the huge difference in preference towards Our model - Short, while the non-sports fans do not consider the length of celebration a huge factor for the highlight itself. This is further strengthened in comparison 3 where the sports fans preferences were pretty much equal and looking at the comments in Section 4.4.2 it is further confirmed that the length of the clip is a big factor for the sports fans. While for the non-sports fans the preference of Our model - full indicates that the clip length (even if a huge part of the clip is celebration scenes) is not that much of a factor impacting the score given. For comparison 5 it is interesting that there is a more mutual agreement in which model is preferred for the non-sports fans in contrast to the sports fans. Looking at some of the comments stated in Section 4.4.2 the majority of the sports fans prefer Our model - Short due to the fact that replay is added and is pretty much straight to the point, while another comment stated that the clip felt a little abrupt and would be weird including this clip during a game, but after a game it works fine. So this disagreement in the preferred model could come from that the clip felt abrupt or some of the participants misunderstanding that these clips actually are supposed to be highlights posted after a game.

Soccer fans

As discussed in Section 3.7.1 we want to categorize participants who watch soccer matches once or several times a week and see how they compare to participants that never watch soccer matches or less than once a week to see if there is any bias across the two different groups and filter out our target group who are most likely to be viewing these highlights in a realistic scenario.

Model name	Average score	Standard deviation	Median
Our model - Short	7.24	2.00	8
Our model - Full	6.72	1.88	7
Original	5.82	1.98	6

Table 4.27: Statistics for people watching soccer once a week or several times a week.

Model name	Average score	Standard deviation	Median
Our model - Short	7.55	1.96	8
Our model - Full	6.95	2.29	7
Original	5.96	2.24	6

Table 4.28: Statistics for people watching soccer less than once a week or never.

After grouping the participants into soccer fans and non-soccer fans we see that the ranking remains the same for both groups, as for the ranking given by all the participants in Figure 4.24. The interesting part for these two groups is that the group for soccer fans seems to have given a lower average score for all the models, compared to the participants in the non-soccer fans group as seen in Tables 4.27 and 4.28. The standard deviation for Our model- Short is fairly close, whereas the standard deviation for Our model - Full and Original is relatively higher for the non-soccer fans group. This suggests that there is a more mutual agreement for the soccer fans group for these models than the non-soccer fans group.

Looking further into the preferences of the two participant groups as shown in Figures 4.23 and 4.24 we see that for Comparison 1 and Comparison 4 the distribution of preferences remains pretty much the same regardless of being a soccer fan or not. The interesting thing about Comparison 2 is that for the non-soccer fans we see there is a majority that has a preference for either Our model - Full or Our model - Short while for the soccer fans the majority preferred Our model - Short while Our model - Full is significantly less preferred in comparison to participants with no preference. This could be an indication that the soccer fans do not really mind watching more of the celebration scenes in comparison to the non-soccer fans. In comparison 3 we see that for the soccer fans a high number of participants had no preference regarding Our model

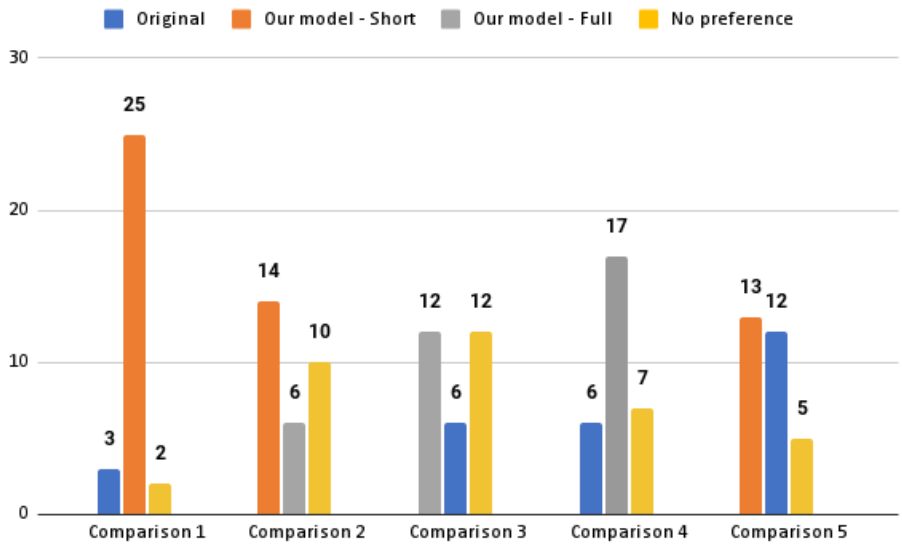

Figure 4.23: The preferred model for soccer fans with respect to the comparison.

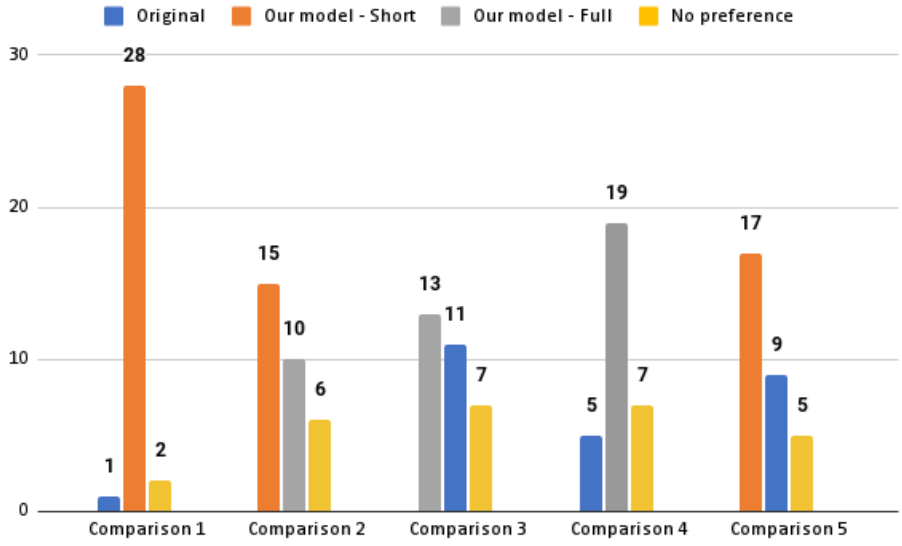

Figure 4.24: The preferred model for non-soccer fans with respect to the comparison.

- Full compared to the original model which is an indication that these participants did not care/notice that much about including more time for the replay. For the non-soccer fan participants, the preference of the model is almost evenly split. Further inspecting the comments as seen in Section 4.4.2 the comments suggest that the reason for these preferences are that some participants simply think they are similar, the replay adds

more value, and that some participants preferred the original model due to the fact that it showed additional angels of the build-up leading to the goal. Looking at comparison 5 we see a clear preference for Our model - Short for the non-soccer fans, while for the soccer fans we see a more split preference between the two models. Further inspection of the comments as seen in Section 4.4.2 that including the replay adds value, but that the cut is a little abrupt. This suggests that the reason for the split preferences could be that some of the soccer fans prefer the more clean transition than the replay containing more angles of the goal. Given the type of goal is also important to take into account, given that this goal is a cross ball following a header it is not an "amazing" goal, and perhaps given the quality of the goal the soccer fans do not feel they need to see as many replays of it (and therefore the cleaner cut is preferred) and the non-soccer fans feel like the replay is more needed for this goal and do not take the abrupt cut as much into account.

Gender

As discussed in Section 3.7.1 we want to group the participants by gender to see if there is any bias between the female and male gender.

Model name	Average score	Standard deviation	Median
Our model - Short	7.39	2.03	8
Our model - Full	6.72	2.15	7
Original	5.86	2.13	6

Table 4.29: Statistics for the Male gender.

Model name	Average score	Standard deviation	Median
Our model - Short	7.42	1.85	8
Our model - Full	7.18	1.90	7
Original	5.98	2.09	6

Table 4.30: Statistics for the Female gender.

Upon inspection of the statics shown in Table 4.29 and 4.30 we see that the ranking remains the same for both groups, as for the ranking given by all the participants in Figure 4.24. the interesting part is that the female participants gave a higher average score than the male participants for all the models and that the female participants have a lower standard deviation across all the models compared to the male participants.

Looking further into the preferences of the genders as shown in Figures 4.25 and 4.26 we see that for Comparison 1 and Comparison 5 the distribution of preferences remains pretty much the same regardless of the participant's gender. The interesting thing about Comparison 2 is that for the female gender we see there is a majority that has a preference for either Our model - Full or Our model - Short while for the male gender the

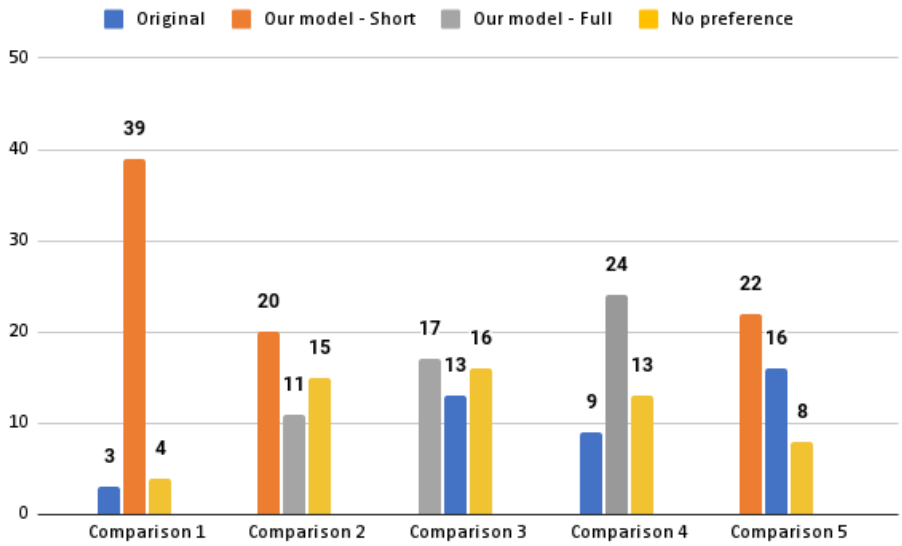

Figure 4.25: The preferred model for the male gender with respect to the comparison.

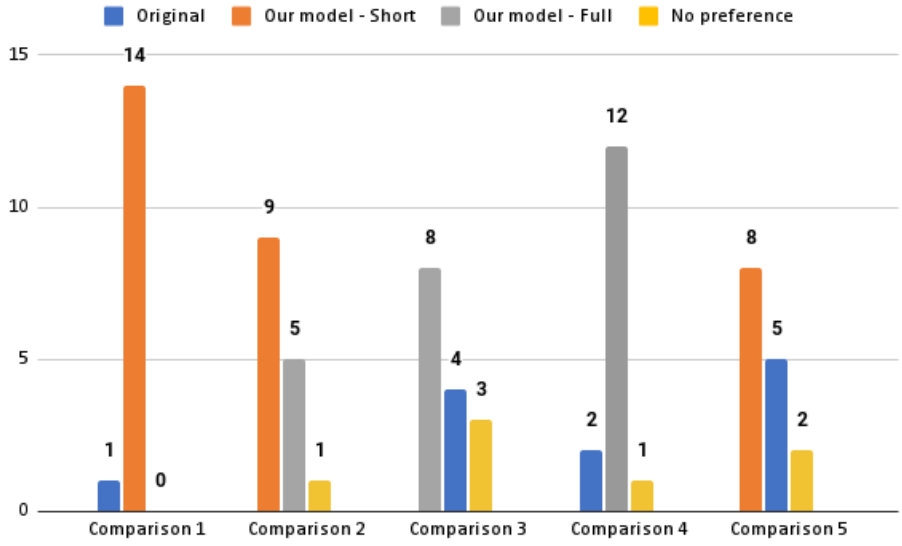

Figure 4.26: The preferred model for the female gender with respect to the comparison.

majority preferred Our model - Short while Our model - Full is significantly less preferred in comparison to participants with no preference. This could be an indication that the female gender does not really mind watching more of the celebration scenes in comparison to the male gender. In comparison 3 the preferences of the male gender are fairly evenly split between no preference, Our model - Full and no preference while for the female gender

Our model - Full is preferred by the majority which could be a product of some bias between the genders. As discussed for comparison 3 there seems to be an indication of some bias between the genders in comparison 4. The majority of the female gender preferred Our model - Full whereas the male gender 13 of the participants had no preference and 9 participants preferred the original model.

Age

As discussed in Section 3.7.1 we want to group participants by age and see how they compare, to see if there is any bias across the two different groups. The reason for diving the age groups into two groups is that as we can see from Figure 4.10 we were unfortunately only able to gather 9 participants over the age of 29, so the rest of the participants fall in under the age group $20 - 29$. Therefore we combine the age groups of everyone over 29 years to one age group to be able to have a decent amount to give us some pointers and indication of bias.

Model name	Average score	Standard deviation	Median
Our model - Short	7.31	2.01	8
Our model - Full	6.79	2.06	7
Original	5.86	2.18	6

Table 4.31: Statistics for the age 18 - 29.

Model name	Average score	Standard deviation	Median
Our model - Short	7.93	1.75	9
Our model - Full	7.11	2.31	8
Original	6.06	1.72	6

Table 4.32: Statistics for the Older participants group.

After grouping the participants into the different age groups we see that the ranking of the models given by the two age groups remains the same as in Table 4.24. The interesting part about the statistics (seen in Tables 4.31 4.32) for these two groups is that the group of older participants seems to give a higher average score for all the models compared to the group of younger participants. The older group has a significantly lower standard deviation for Our model - Short and Original compared to the younger participants, so there seems to be more of a mutual agreement for the scores given to these models. But, as we can see there is a higher standard deviation for Our model - Full for the older participants compared to the younger participants. These numbers for standard deviation indicate that there is a stronger mutual agreement in the older participants group for Our model - Short and Original compared to the younger participants group, but a stronger disagreement for Our model - Full in the older participants group compared to the younger participants group.

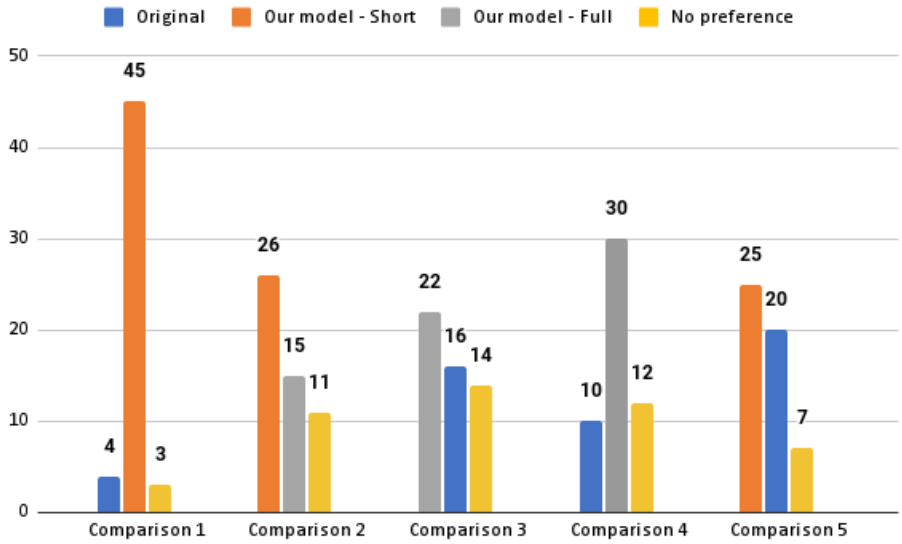

Figure 4.27: The preferred model for the younger participants with respect to the comparison.

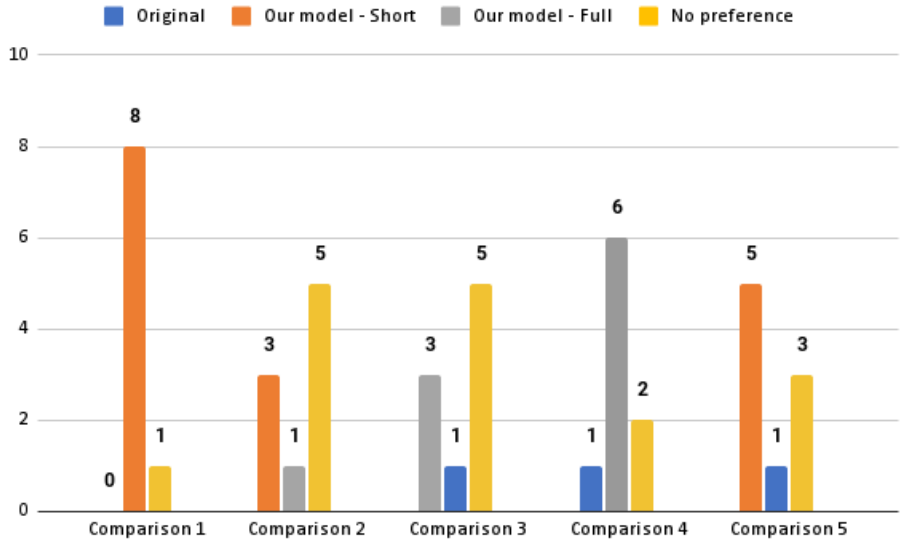

Figure 4.28: The preferred model for the older participants with respect to the comparison.

Looking at the preferences of the younger participants in Figure 4.27 we see that the distribution of preferences is fairly similar to the preferences for all the participants with no grouping of participants as shown in Figure 4.20. The reason for this is probably because the majority of the participants fall under this age group, therefore we will not comment on these preferences any further because it would end up being very similar

to the discussion in Section 4.4.2. For the older participants group, we see in Figure 4.28that for Comparison 1 and 4 the preferences do not vary that much from the preferences shown in Figure 4.20 (preferences with no grouping). In comparison 2 we see that the majority has no preference between Our model - Full and Our model - Short, which could be an indication that this age group does not mind the length of the celebration scenes that much. For comparison 3 wee see the same pattern as for comparison 2, the majority of participants in this age group did not really see a difference, and this is fair since in this comparison there really is not that much of a difference, except that Our model - Full has a cleaner cut (cuts on the logo, and not before). This could be an indication that this age group does not care that much about how clean the cut is, which is further confirmed in comparison 5 where Our model - Short has a more abrupt cut compared to the Original model.

Editing experience

As discussed in Section 3.7.1 we want to group filter out the participants that have experience with video editing from the rest of the participants, making the assumption that people with video editing experience will have a stronger foundation for evaluating the quality of the clip with more focus on the quality of the cut itself.

Model name	Average score	Standard deviation	Median
Our model - Short	7.59	2.33	8
Our model - Full	6.67	2.42	7
Original	5.47	2.42	5

Table 4.33: Statistics for the age participants with video editing experience.

After filtering out the participants that have experience with video editing, we see that the ranking of the models given by the participants with video editing experience shown in Table 4.33 remains the same as in the overall ranking shown in Table 4.24. The interesting part about these results is that the average score for Our model - Short is relatively higher compared to the majority of the scores given by the other groups, while the average score for Our model - Full and Original is relatively lower compared to the majority of the other groups. This could be an indication that people with video editing experience pay more attention to the shortening of the crowd, and are more pleased with this type of cut compared to the other two models. The interesting part here is that the standard deviation is relatively high for all the models compared to the standard deviation for the other groups, this indicates that there is more of a disagreement between the participants when it comes to the models.

Looking at the preferences of the younger participants in Figure 4.29 we see that the preferences for comparison 1 there is a mutual agreement that Our model - Short is preferred, except for 1 individual that had no preference. For comparison 2 there is pretty much a 50/50 split between

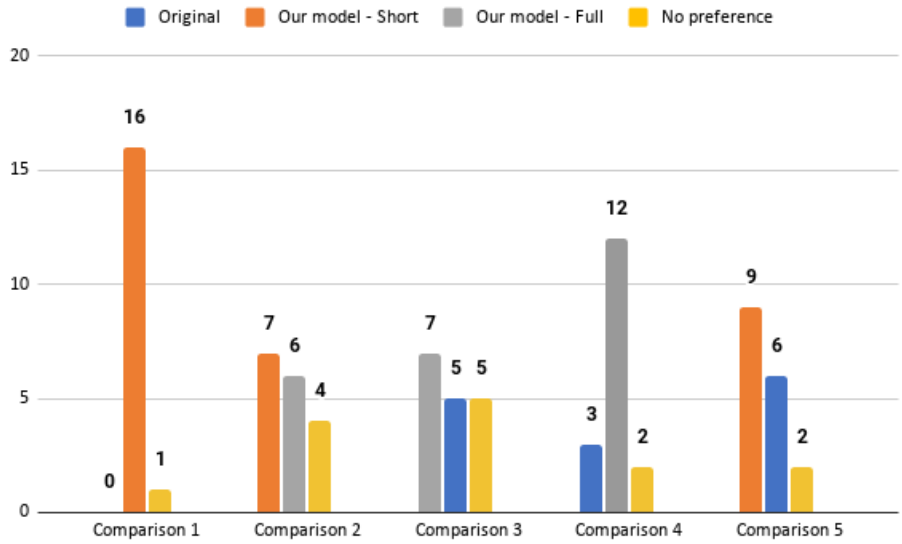

Figure 4.29: The preferred model for participants with video editing experience with respect to the comparison.

which model is preferred between Our model - Short and Our model - Full, which indicates that the participants with video editing experience were split between the preference of including the full celebration scene or not. For comparison 4 we see a clear preference for Our model - Full that indicates that the participants with video editing experience think that including the replay and having a clean cut is valuable for the highlight itself. The interesting part for comparison 5 is that Our model - Short is preferred by the majority, but a good amount of the participants preferred the original model, a reason for this could be that while the participants think replay is important to include, the participants that preferred the original model thought the cut provided by Our model - short was too abrupt.

We further inspect the scores given by the two participants that have worked with video editing in a professional setting as it is interesting to see their opinion on the different clips since these are the participants that will most likely know what to look for in a good cut on a professional level. Both of the participants with professional video editing experience answered they Never watch soccer matches, sports highlights on the web, and soccer highlights on the web. The only relationship towards sports they differentiate in is that participant 2 considers herself a sports fan and watches sports less than once a week, while participant 1 never watches sport and does not consider himself a sports fan. So it is important to keep in mind that these participants do not fall into the group of soccer fans and do not seem to have much relation to soccer, which could impact the scores given.

As we can see in Table 4.34 participant 1 mostly thinks all the models are a 3/10 even though he seemed pleased with Our model - Short in

Participant 1	Our model - Short	Our model - Full	Original
Comparison 1	7	-	2
Comparison 2	3	3	-
Comparison 3	-	3	3
Comparison 4	-	5	3
Comparison 5	3	-	3
Average score	4.33	3.67	2.75

Table 4.34: Scores and average given by professional editor 1.

Participant 2	Our model - Short	Our model - Full	Original
Comparison 1	9	-	3
Comparison 2	7	4	-
Comparison 3	-	4	8
Comparison 4	-	9	5
Comparison 5	9	-	5
Average score	8.33	5.67	5.25

Table 4.35: Scores and average given by professional editor 2.

comparison 1, and thought Our model - Full was decent in comparison 4. But, even with giving a relatively low score to all the models, by looking at the average score given by the participant we see that the ranking of the models remains the same as for the ranking of the models for all the other groups. The interesting part about participant 1 is that he left a comment on comparison 1, "clip 2 was way more engaging" (Clip 2 referring to Our model - Short) so this could indicate that the foundation of participant 1 for judging the clips is how engaging they are. If participant 1 was rating based on how engaging the clips are much of this responsibility falls on the actual production, we have no control over the camera angles available, type of goal, or the angles the production chooses to include in the replay. Looking at Table 4.35 we can see that this person was fairly optimistic for Our model - Short, but on average considered Our model - Full and Original almost equally good. Actually having a lower median for Our model - Full than the Original model. Looking at the scores given this could be an indication that this participant thinks that a highlight should be short, but does value the importance of replay (even if a consequence is a slightly longer clip) and think what makes good video editing is removing unnecessary scenes.

Looking at both Tables4.34 and 4.35 we see that the opinions of the participants with video experience are very split on the scores for the models. There could be several factors for this, such as variation of the professional setting they have worked in, personal bias, or experience with sports.

4.4.4 Final thoughts and bias

As discussed in Section 4.4.1 we were able to gather a decent number of 64 participants, but having to filter out 3 participants due to the score of 10 across all the models not providing any value for our analysis of the results. Due to our approach for participant selection and the number of participants we were able to gather we see that some of the groups are poorly represented. The groups that have the lowest representation are participants over the age of 29 and professional video editors, so the important thing to keep in mind is that the results provided for these groups will be stronger influenced by personal opinions and are too small to be able to generalize for these groups, but these results will still be good indications and bring value to our research. Upon further inspection of the results as shown in Section 4.4.2 we see a clear preference for Our model-Short with Our model - Full as runner up and the original model with the lowest preference when it comes to the scores and looking at the actual preferences with respect to the comparisons. The comments discussed in Section 4.4.2 are a huge factor for helping us understand what the participants value in a clip, and whatnot. The comments serve as great indicators for future work, to even improve our models further. By inspecting the comments we see that some of the comments given on certain comparisons disagree with each other, so clearly it is impossible to make everyone satisfied since this is such a subjective problem at hand. The most important takeaways we want to take with us further in the next iteration are that people thought some of the clips were too long, so to experiment with making the clips even shorter and experiment with different approaches to make the cuts where the crowd is removed cleaner (less abrupt). For comparison 4, a penalty clip was shown, and the original model included a short scene of what lead to the penalty which some of the participants seemed to like (but disliked that it looked so untidy), while Our model - short did not include what lead up to the penalty. So based on the comments given for comparison 4 we think it would be interesting to find an approach to handle these types of special events. Therefore it could be interesting to implement a model that if a special event such as a free-kick or penalty is present to include what lead up to it in the highlight.

We further have grouped our participants into different groups in Section 4.4.3. We split the participants into more general groups such as gender and age to see if there is any bias depending on this, and we discover that the female gender had a lower standard deviation and gave higher scores for all the models compared to the Male gender. Also looking at the age grouping we discovered that the older participants gave higher scores compared to the younger participants, and had the lowest standard deviation for Our model - Short and the original model. Then looking at the groups that are more sports orientated we discover that sports fans and soccer fans gave lower scores than the groups of non-soccer fans and non-sports fans, this could be an indication that the group for sports fans and soccer fans are more aware and have higher expectations for the highlights they want to see. Perhaps the most interesting part is that for all the groups

of participants the ranking of the models remained the same with Our model - Short in first place, Our model - Short second, and the original model last place. This is a good indication that we are on the right path, and Our model - Short is the best model with the most potential based on this evaluation.

The final things to take into consideration for the results are that a group of 61 participants is a relatively low number to be able to say that this generalizes for everyone. Therefore, these results should only be viewed as indications for further work, and it would be ideal to conduct this type of survey/evaluation on a larger scale with a higher number of participants. By conducting this evaluation on a larger scale we would be able to further confirm if there is bias across the groups and strengthen some of the theories proposed and see if they still generalize to a larger number of participants. As we also have seen from the preferences of the participants the preferences for comparison 1 had a stronger mutual agreement for the preferred model compared to some of the comparisons, so the actual event shown is something that will impact the scores and preferences. If we were to evaluate the models on a different set of events the scores would probably look different, and also not having a true random order for showing the events due to the limitations of google forms could impact the results. Optimally we would like the participants to watch a much higher number of clips, but it is important to limit the time of the survey due to the fact that the longer the survey goes the chance of participants dropping out, just doing enough to complete the survey without rather than providing thoughtful answers and not answering optional questions such as comments in our case [12]. Taking this into consideration some of the participants could have been bored towards the end and not provided thoughtful answers, this could also be an explanation for the decrease in comments provided towards the end of the survey. The participant's mood could also be a final factor for the scores given, so an interesting experiment could be to follow up the same participants on a later point, to see if their preferences will such as on some days they do not mind watching longer clips and on some days they are in the mood for shorter clips [47].

4.5 Discussion

In the previous sections, we have presented the process and result from our experiments, resulting in a final system able to make two different versions of a highlight. We compared these to the existing solution used today. In this section, we will further discuss the results of some choices taken during the process.

4.5.1 Clipping in practice

Our presented models for clipping rely heavily on high-quality production patterns which are mostly found in the top soccer leagues. In a realistic scenario, our models would fail to generalize to lower quality leagues

where there is no guarantee for scene changes or logo appearances and end up making a default cut for the highlights. There is also no guarantee that every top soccer league follows the same production pattern and that our models would be able to generalize to these, so there is a good possibility if this system were to be used by different leagues there would be a need to study the production pattern and make adjustments to fit these particular leagues.

Furthermore, the quality of a highlight is a task that is difficult to evaluate and satisfy everyone due to the subjective nature of the task at hand and all the real-life factors that could impact the results. Such as which teams the participants support, as one participant commented in the evaluation "Always nice to see Lillstrøm concede a goal, thank you" which most likely impacted the score given for these clips. Other factors such as which type of goal is shown could also be impacting the scores given, as one participant commented "Clip 2 with a length of 57 seconds is too much for a goal that is not nominated for the puskas" so if this was a more exciting goal perhaps the participant would be fine with the length of the clip. We also discovered through further conversations with the participants and by looking at the comments the scores will vary depending on the goal shown. While we have the capability to make the clips more engaging and better, some things are out of our control that will impact the scores given. This includes the camera angles available, what the production chooses to include in the highlight, and the quality of a goal. The final thing that could impact the results is depending on the goal shown and participant watching the highlight, there will most likely be a preference depending on if they want to relive the atmosphere provided by the live goal and celebrations or they just want to see a quick summary of the goal.

4.5.2 Retrospect of process

The fact that our SVM models failed to generalize for a larger dataset as the PL (16/17) compared to the CNN, was a consequence that when the SVM models was implemented our scope was for a smaller dataset such as Eliteserien containing simpler logotypes, but due to the release of SoccerNet-v2 our scope expanded and the SVM models showed disappointing results. In retrospect due to the available resources and results of the SVM models we probably would either have dropped the SVM leaving more time for improving the CNN or doing further research on other aspects of this thesis. It would also be interesting to see if the SVM models would have been able to generalize to more complex datasets if implemented differently or using other feature extractors. Some ideas that we think would improve the SVM models even further is using end to end training where you train the feature extractor based on the SVM as the output layer [20]. We could also have done some kind of clustering for the features or adding a max pooling layer for the CNN to reduce the number of features given to the SVM and increase the number of iterations for the SVM with a lower learning rate (or introduce a decreasing learning rate).

One of the weaknesses with our Eliteserien logo dataset is the size

of both classes. To counter the lack of team logos present in the train dataset, we made a synthetic dataset to supplement the already existing one. We saw a decline in performance for the background class when introducing this dataset, which may be due to the lack of backgrounds. Adding backgrounds from our SoccerNet set to Eliteserien could have made the final evaluation on Eliteserien more reliable. This was not done as we had already started experimentation on Eliteserien before we included SoccerNet-v2.

When training and evaluating the logo frame classifiers, we overestimated the importance of logo recall while underestimating the precision. It was not before we ran tests on the full logo transition module on SoccerNet that we found the problem of false positives. In hindsight, we should have analyzed the false positive more thorough in the frame set earlier, to change our weighting of the metrics when evaluating them.

When we added hard samples to the Train Medium dataset of our SoccerNet logo frame dataset in Section 4.1.4, we used only two of the models, VGG inspired and ResNet with input $108 \times 192 \times 3$. We saw that the VGG model increased overall performance, while the others had a decrease in weighted F1-score and precision. This suggest a bias, and in retrospect, we should have extracted them by using an unrelated classifier in order to compare the results without any bias. The results still suggests that hard samples are beneficial, and that the hard samples are relevant to all the models.

With our transfer fine-tune learning strategy for ResNet50V2 from Keras [16, 33], we hypothesized that with a small learning rate, features learned from the ImageNet [19] would be preserved. We mainly proposed this strategy with the Premier League dataset in mind, as this is more complex, and could utilize the advantage. This is a fair assumption, but from the results, we see that the performance is bad for both leagues. In retrospect, we should have tested with a higher learning rate, as it looks like the low learning rate do not enable the model to learn important features that is not present in the pre-trained ImageNet weights. We saw good performance with the other, more straight forward transfer learning, which gives sufficient results. It may still be the case that a solution in between would have performed even better.

4.6 Summary

In this chapter, we started by defining what we expect from the logo detection module and scene boundary detection module to be able to produce highlight clips of a good technical standard. We looked at what resolutions to experiment with.

We found that all the models performed well on the Eliteserien dataset and that the transitions are very simple. We found that almost all of the SVM models we tested, outperformed the CNNs on the validation with perfect scores, but our lightweight VGG model with a grayscale input of 54×96 pixels performed best on the test set, with a 100% F1-score. The

best Simple CNN scored an F1-score of 98.4%, while the more complex and deep network ResNet50V2 [33] model scored 98.2%.

We then moved on to the SoccerNet PL16/17 logo dataset is much more diverse in both logos and backgrounds. We saw that higher input resolutions performed better, and we included the resolution 144×256, which performed best for the VGG inspired CNN with grayscale input, achieving an F1-score of 99.4%. When looking at the recall for each type of logo, we found that it was the Simple PL logo that was harder for the small resolutions to correctly classify, as this is very small, and in many cases hard to see when it overlaps a bright background.

We experimented with fine-tuning ResNet50V2 with pre-trained weights trained on ImageNet [19], by first training the dense network alone, before training the full model with a low learning rate. The model was not able to learn the Simple PL logo. We concluded that the model had not learned the necessary features to separate this class from backgrounds when the background colors were too similar. This was most likely due to a too low learning rate.

We found that the SVM was hard to train on this set, and struggles with converging. Due to the poor results achieved, computational cost, and execution time, we decided to discard the SVM for this dataset.

We ran experiments on the full-length test set of SoccerNet-v2 for the Premier League season 2016 - 2017 and found that the models tested found all logo transitions. However, the number of false positives suggested that they had not encountered enough backgrounds in training. We added almost $8,000$ more backgrounds extracted evenly from the training set videos, as well as extracting over $6,000$ hard samples. The results showed great improvement. We experimented with more than 40.000 extra frames, but saw a decrease in performance. This suggests that hard samples are an effective way to better learn to separate the outlying samples.

On the full logo module experiment, the Simple and VGG inspired CNN performed well for their best models, but was outperformed by ResNet, suggesting that a deeper network is necessary for a league with a much more diverse broadcast production than Eliteserien. With an RGB input of 108×192, it reached a precision of 100% and a recall of 95.5%.

We trained and evaluated TransNet-V2 [64] on the SoccerNet shot boundary dataset, and compared the performance to the pre-trained version. The preliminary test on the Premier League subset showed promising results for the model we trained, with great performance on the gradual transitions. However, further training and testing on the full test set, showed that it was outperformed by the pre-trained model. It still performed well, and with more complete labeling of gradual and logo transitions may boost the performance. Further testing on the full-length dataset of SoccerNet-v2 showed a very poor precision, but after analyzing all the transitions from the Premier League matches, we found that almost all were transitions failed to be annotated.

Moving on, we described how the subjective evaluation of highlight clips was conducted, the distribution of participants gathered, and how some groups are poorly represented. We discussed the metrics used for

analyzing the results and presented the results for the subjective evaluation. We then inspected the comments, different groups of participants, and individual answers to filter out valuable information. We further discovered based on the results of the evaluation that the highest-scoring model is Our model - Short, Our model - Full as runner up and the Original model in the last place. We then discussed how difficult of a task it is to evaluate and how preferences will vary from participant to participant and real-life factors that could impact the results. In the final Section 4.5 we discussed possible flaws of our models and how the preferences of what a good highlight is could vary in practice, and just how difficult of a task it is to satisfy the majority. we then finally discussed how we could have done things differently and perhaps should have done differently.

Chapter 5

Conclusion

Today, highlights in soccer matches are manually annotated and clipped by human operators. This is a time-consuming, tedious, and expensive task. The clips are often a preset time interval instead of a tailored interval that fits the specific event. The editors might not even have time to clip it as it can often be important to distribute it as close to the live event itself. It could be edited later, but in many cases, this is too expensive.

In this thesis, we experimented with automating the process of highlight generation using Scene boundary detection, logo detection, and a production-based algorithm.

Through experimentation, we concluded that the VGG inspired CNN using grayscale input of 54×96 achieving a 100% F1-score was the best fit for our logo detection module on Eliteserien. For the more complex Premier League logo dataset, we concluded that the ResNet CNN using RGB input of 108×192 achieving an 0.997 F1-score was the best fit for our logo detection module. We trained and evaluated TransNet-V2 [64] on the SoccerNet shot boundary dataset, and compared the performance to the pre-trained version, and concluded that the pre-trained version was sufficient for the Scene boundary detection model of our system.

Further, we combined these modules and implemented two different configurations of our system, one including full celebration scenes, and the other removing certain celebration scenes. We compared these to the already existing model in Eliteserien.

Based on the qualitative and quantitative evaluation through a user study, we showed that Our model - Short and Our model - Full consistently produces more compelling highlight clips compared to the original model used in Eliteserien today. Upon inspection of the preferences of the participants we discovered that due to the random nature of the original model (using a set time interval for highlight extraction), it achieves low scores when it "misses", while in the cases where it "hits", the preference of model is more even.

The results showed that this is a complicated task and there is a variety of which model is preferred impacted by several different factors such as background, real-world factors, mood, etc.

5.1 Main contributions

Based on the problem statement described in Section 1.2, we wanted to make a machine learning model that provides a soccer highlight of a high standard, and this involves objective evaluation of key modules and a subjective evaluation of the final system. We will here restate the objectives set in Section 1.2, and our main contributions in association with each of them.

Objective 1 Research and design a system to automatically extract highlight clips from soccer videos. Identify and prepare the necessary data needed for development and final evaluation.

To meet this objective, we researched machine learning approaches for video summarization, Scene boundary detection, and logo detection. Based on soccer broadcast production, we proposed a highlight clipping system based on logo recognition tailored for a specific league and season and a shot boundary detection.

We designed our logo detection as a binary image classification task. We analyzed state-of-the-art approaches in the field of image recognition. We settled on VGG [62] and ResNet [32, 33] architectures, both reaching impressive performance on the ImagNet ILSVLC dataset [19, 58]. Our candidate logo recognition models are ResNet50V2 [33], a lightweight CNN based on the VGG architecture [62], a simple CNN architecture, and an SVM using VGG16 [62] as a feature extractor.

We created a frame logo recognition datasets for two different leagues, Eliteserien season 2018 containing $1,025$ logo and $7,025$ background frames, and Premier League season 2016 - 2017 extracted from SoccerNet-v2 [18] containing $23,194$ logo and $43,260$ background frames. Both with high quality with respect to the sampling and labeling quality, but differ in size and complexity of logos. To compensate for insufficient data from Eliteserien, we supplemented with synthetic data using a script adding extra logo frames.

Shot boundary detection is a popular field of research and has shown great performance results in the recent years [39, 63, 64, 70]. For our shot boundary detection task, we used TransNet-V2 [64], a state-of-the-art model with great performance on the shot boundary benchmark datasets ClipShots [70], RAI [11], and BBC [10]. We tested TransNet-V2 with its complimentary pre-trained weights, trained on ClipShots [70] and generated transitions using clips from TRECVid IACC.3 [8], as well as performing our own training on soccer clips only.

To train and evaluate, we extracted over $150,000$ clips of 100 frames containing transitions from the full SoccerNet-v2 dataset with labels suitable for TransNetV2 [64]. Finally, we prepared a subjective evaluation for our system and the current system used in Eliteserien, on the Eliteserien dataset.

Objective 2 Implement a system for clipping highlights and perform an objective evaluation of the different modules used, i.e., logo detection and scene boundary detection.

To meet this objective, we implemented the candidate models for logo detection, using SVM and CNN. We experimented on the Eliteserien dataset and Premier League dataset and assessed the performance using several metrics. We showed that for the Eliteserien dataset both the SVM and CNN achieved satisfactory results for the task at hand and the VGG model with a grayscale input of 54×96 pixels achieved the best result with a 100% F1-score. We also showed that with a larger and more complex dataset such as the Premier League dataset, the CNN still performed well, while the SVM models failed to reach satisfactory results. We further improved the CNN models by adding more backgrounds, including hard samples extracted by our classifiers, which proved to be effective. We find that the ResNet model with an RGB input of 108×192 reaches the best scores with a precision of 100% and a recall of 95.5% for logo transition detection on five full-length matches.

We evaluated the state-of-the-art shot boundary detection model TransNetV2 [64] on the SoccerNet-v2 [18] dataset. We showed that a pre-trained version trained on regular video clips performed well on soccer videos for gradual and abrupt transitions. We experimented with training the model specifically on soccer clips, which showed potential but did not reach the levels of the pre-trained model. We find the model to be frame-accurate and therefore a sufficient model for our scene boundary detection module.

We combined logo detection and shot boundary detection in order to form a full system that outputs highlight clips, with high technical performance. We implemented two different clipping protocols. The first configuration of the system includes all the celebration scenes between the event and the replay, and the other configuration of the system excludes several celebration scenes.

Objective 3 Perform a qualitative and quantitative evaluation of the system through a user study that evaluates the subjective nature of high-quality soccer highlight clips.

For this objective, we performed a qualitative and quantitative evaluation through a user study for Our model - Short, Our model - Full, and the Original model used today in Eliteserien.

64 participants rated highlights of five goals generated by our system and the existing solution and compared them with each other. The rating goes from 1 (worst) to 10 (best). Based on the results from the survey, we found the following ranking of the models:

1 Our model - Short achieved an average score of 7.40

2 Our model - Full achieved an average score of 6.84

3 Original model used in Eliteserien today achieved an average score of 5.89.

We found that due to the random nature of the Original model using a fixed interval for highlight extraction it achieves low scores when it "misses", while in the cases where it "hits", the original model achieves decent results compared to the other models.

Further, we grouped the participants by soccer fans, sports fans, gender, age, and editing experience, and found that the ranking of the models remains the same for all the groups, but the preferences, scores, standard deviation, and median varied.

Finally, we identified possible biases for the different groups of participants and discuss possible biases and real-world factors that could impact the results.

Our contributions are interesting in the context of the problem statement, and the presented results are valuable as for how much impact a good highlight clip has on consumer satisfaction. We showed that the machine was able to provide highlight clips of reliable technical standards based on the technical results and empirical evaluation. From the gathered quantitative results from the online survey, we showed that the technical performance in conjunction with our two different clipping protocols leads to better results than the solution of the fixed interval used today. We also identified that what is considered a compelling highlight is subjective, and there are differences in what production strategy the potential users prefer. Our work gives a strong foundation for further work with using machine learning to generate automatic highlight clips in soccer.

5.2 Future work

There are several aspects of the solution that have the potential for future works. We show that a simple image classifier can work as a logo transition detector when tailored for one league, given that the transitions are consistent throughout the season. It could be interesting to see if a more generalized model can be made, by for example utilizing temporal features, such as color histograms. This could help to mitigate the problem of false negatives. One example could be to modify an existing model for shot boundary detection, such as TransNetV2 [64], to be able to classify transitions as abrupt, gradual, and logos.

We focus on goals only, and it could be of interest to widen the scope to other events, such as goal attempts, cards, fouls, substitutions, or events in other sports. It would also be interesting to see it expand into a full summarization system. As discussed in Section 4.4, capturing the event leading up to a goal, card and such, could be very interesting. Linking associated events, and combining them as a highlight clip can enrich the experience by providing the context of an event. In the context of a full summary system, this can also help make the summary more complete.

Audio is an important aspect of sports broadcasts. Both commentaries and crowd cheering. It could be interesting to see how this impacts the quality of a highlight. Examples could be not to cut in the middle of a word from the commentators, or using natural language processing in order to capture enough commentaries to retain the initial meaning. Also capturing the commentator's reaction can increase the quality.

High-quality, high-volume datasets are one of the most important parts of enabling research, both for training and benchmarking. In order to achieve the aforementioned research, more comprehensive datasets and labels are needed, such as full labeling of logo and gradual transitions, and more action event annotations. This could be done manually, or by designing more complex gathering tools that can collect relevant data from for example web sources.

We would also like to see a more thorough study of how the design of highlights in soccer or sports in general impact the consumer, in order to gain information and insights into how the automatic clipping can best capture the event, and either strengthen/disprove the different biases discussed in Section 4.4. One aspect that was not mentioned in the survey was sound, and it could be interesting to see how it affects a clip. This can be done by gathering qualitative data, subjective or objective, on a bigger scale. Collecting qualitative data from more comprehensive interviews or tests.

Finally, it would be interesting to see how our system compares to highlight clips extracted by a professional editor.

Appendix A

Appendix

Algorithm 1: Clipping protocol.

```
1  Clipping protocol (sceneschanges, logotransitions, event);
   Input  : temporal anchors (sceneschanges, logotransitions, event)
   Output: temporal anchors (start, end, cuts)
2  start = defaultStart;
3  end = defaultEnd;
4  cuts = None;
5  for each scene change before event do
6      if Scene change is between thresholds then
7          start = scene change;
8          break;
9      end
10 end
11 if logo trainsitions is found then
12     end = endLogo ;
13     if cutCrowd is true then
14         cuts = celebration scenes outside of thresholds;
15     end
16 else
17     for each scene change after event do
18         if scene change is between end thresholds then
19             end = scene change;
20             break;
21         end
22     end
23 end
24 return start,end,cuts.
```

Model	Input	Precision	Recall	F1 Score
VGG inspired	$54 \times 96 \times 1$	**1.0000**	**1.0000**	**1.0000**
SVM (Simple CNN)	$108 \times 192 \times 1$	**1.0000**	0.9955	0.9978
VGG inspired	$72 \times 72 \times 3$	**1.0000**	0.991	0.9955
SVM (Simple CNN)	$27 \times 48 \times 3$	**1.0000**	0.9776	0.9887
SVM (Simple CNN)	$72 \times 72 \times 1$	0.9865	0.9865	0.9865
VGG inspired	$27 \times 48 \times 3$	**1.0000**	0.9731	0.9864
SVM (Simple CNN)	$72 \times 72 \times 3$	**1.0000**	0.9731	0.9864
Simple CNN	$72 \times 72 \times 3$	0.9954	0.9731	0.9841
VGG inspired	$108 \times 192 \times 3$	0.9954	0.9686	0.9818
ResNet	$54 \times 96 \times 3$	0.9954	0.9686	0.9818
Simple CNN	$108 \times 192 \times 3$	0.9954	0.9686	0.9818
Simple CNN	$27 \times 48 \times 3$	**1.0000**	0.9641	0.9817
VGG inspired	$72 \times 72 \times 1$	0.9954	0.9641	0.9795
Simple CNN SVM	$108 \times 192 \times 3$	**1.0000**	0.9596	0.9794
SVM (Simple CNN)	$27 \times 48 \times 1$	**1.0000**	0.9596	0.9794
Simple CNN	$72 \times 72 \times 1$	0.9733	0.9821	0.9777
Simple CNN	$54 \times 96 \times 1$	0.9819	0.9731	0.9775
Simple CNN	$108 \times 192 \times 1$	0.9907	0.9596	0.9749
Simple CNN	$54 \times 96 \times 3$	0.9861	0.9552	0.9704
VGG inspired	$108 \times 192 \times 1$	**1.0000**	0.9417	0.97
VGG inspired	$54 \times 96 \times 3$	0.9729	0.9641	0.9685
VGG inspired	$27 \times 48 \times 1$	0.986	0.9462	0.9657
ResNet	$108 \times 192 \times 3$	0.9680	0.9507	0.9593
Simple CNN	$27 \times 48 \times 1$	0.9811	0.9327	0.9563
SVM (VGG16)	$108 \times 192 \times 3$	0.9528	0.9058	0.9287
SVM (VGG16)	$72 \times 72 \times 3$	0.7714	0.8475	0.8077

Table A.1: The final results on the Eliteserien logo frame test set.

Bibliography

[1] *A Refresher on A/B Testing*. June 2017. URL: https://hbr.org/2017/06/a-refresher-on-ab-testing# (visited on 23/04/2021).

[2] *About FFmpeg*. https://www.ffmpeg.org/about.html. (Accessed on 06/01/2021).

[3] 'ACTIVITY REPORT 2018'. In: (2019). URL: https://resources.fifa.com/image/upload/yjibhdqzfwwz5onqszo0.pdf.

[4] Chen-Yu Chen et al. 'Motion Entropy Feature and Its Applications to Event-Based Segmentation of Sports Video'. In: (2008). URL: https://doi.org/10.1155/2008/460913.

[5] Hossam M. Zawbaa et al. 'Machine Learning-Based Soccer Video Summarization System'. In: (2011). URL: https://www.researchgate.net/publication/216880199_Machine_Learning-Based_Soccer_Video_Summarization_System.

[6] Peng Xu et al. 'Algorithms and Systems for Segmentation and Structure Analysis in Soccer Video'. In: (2001). URL: https://www.researchgate.net/publication/228907792_Algorithms_And_System_For_Segmentation_And_Structure_Analysis_In_Soccer_Video.

[7] Tjondronegoro D et al. 'Sports video summarization using highlights and play-breaks'. In: (2003). URL: https://www.researchgate.net/publication/27463639_Sports_Video_Summarization_using_Highlights_and_Play-Breaks.

[8] George Awad et al. 'TRECVID 2020: comprehensive campaign for evaluating video retrieval tasks across multiple application domains'. In: *Proceedings of TRECVID 2020*. NIST, USA. 2020.

[9] Muhammad Awais et al. 'Can pre-trained convolutional neural networks be directly used as a feature extractor for video-based neonatal sleep and wake classification?' In: (2020). URL: https://doi.org/10.1186/s13104-020-05343-4.

[10] Lorenzo Baraldi, Costantino Grana and Rita Cucchiara. 'A Deep Siamese Network for Scene Detection in Broadcast Videos'. In: *Proceedings of the 23rd ACM International Conference on Multimedia*. MM '15. Brisbane, Australia: Association for Computing Machinery, 2015, pp. 1199–1202. ISBN: 9781450334594. DOI: 10.1145/2733373.2806316. URL: https://doi.org/10.1145/2733373.2806316.

[11] Lorenzo Baraldi, Costantino Grana and Rita Cucchiara. 'Shot and Scene Detection via Hierarchical Clustering for Re-using Broadcast Video'. In: *Computer Analysis of Images and Patterns*. Ed. by George Azzopardi and Nicolai Petkov. Cham: Springer International Publishing, 2015, pp. 801–811. ISBN: 978-3-319-23192-1.

[12] Pete Cape. *Questionnaire length, Fatigue effects and response quality: Revisited*. https://www.warc.com/content/paywall/article/questionnaire_length,_fatigue_effects_and_response_quality_revisited/97469. (Accessed on 26/05/2021). 2010.

[13] J. Carreira and A. Zisserman. 'Quo Vadis, Action Recognition? A New Model and the Kinetics Dataset'. In: (Feb. 2018). URL: https://arxiv.org/abs/1705.07750 (visited on 13/06/2020).

[14] Brandon Castellano. *PySceneDetect: Python and OpenCV-based scene cut/transition detection program & library.* https://github.com/Breakthrough/PySceneDetect. (Accessed on 05/13/2021).

[15] François Chollet. *Xception: Deep Learning with Depthwise Separable Convolutions*. 2017. arXiv: 1610.02357 [cs.CV].

[16] François Chollet et al. *Keras*. https://keras.io. 2015.

[17] Anthony Cioppa et al. *A Context-Aware Loss Function for Action Spotting in Soccer Videos*. 2020. arXiv: 1912.01326 [cs.CV].

[18] Adrien Deliège et al. *SoccerNet-v2 : A Dataset and Benchmarks for Holistic Understanding of Broadcast Soccer Videos*. 2020. arXiv: 2011.13367 [cs.CV].

[19] Jia Deng et al. 'ImageNet: A Large-Scale Hierarchical Image Database'. In: (2009). URL: https://www.researchgate.net/publication/221361415_ImageNet_a_Large-Scale_Hierarchical_Image_Database.

[20] Erkan Deniz et al. 'Transfer learning based histopathologic image classification for breast cancer detection'. In: (2018). URL: https://www.researchgate.net/publication/327942696_Transfer_learning_based_histopathologic_image_classification_for_breast_cancer_detection.

[21] Scikit-Video Developers. *scikit-video: Video Processing in Python*. https://github.com/scikit-video/scikit-video.

[22] Brandon Doyle. 'TikTok Statistics – Updated August 2020'. In: (2020). URL: https://wallaroomedia.com/blog/social-media/tiktok-statistics/.

[23] 'Entertaining audiences'. In: (2019). URL: https://www.premierleague.com/this-is-pl/the-fans/686489?articleId=686489.

[24] C. Feichtenhofer, A. Pinz and A. Zisserman. 'Convolutional Two-Stream Network Fusion for Video Action Recognition'. In: (2016). URL: https://ieeexplore.ieee.org/document/7780582/ (visited on 13/06/2020).

[25] 'FIFA Survey: approximately 250 million footballers worldwide'. In: (2001). URL: https://www.fifa.com/who-we-are/news/fifa-survey-approximately-250-million-footballers-worldwide-88048.

[26] *FIFA World Cup: Goals scored per game 1930-2018 - Statista*. https://www.statista.com/statistics/269031/goals-scored-per-game-at-the-fifa-world-cup-since-1930/. (Accessed on 05/20/2021). Aug. 2018.

[27] Silvio Giancola et al. 'SoccerNet: A Scalable Dataset for Action Spotting in Soccer Videos'. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)* (June 2018). DOI: 10.1109/cvprw.2018.00223. URL: http://dx.doi.org/10.1109/CVPRW.2018.00223.

[28] Xavier Glorot and Y. Bengio. 'Understanding the difficulty of training deep feedforward neural networks'. In: *Journal of Machine Learning Research - Proceedings Track* 9 (Jan. 2010), pp. 249–256.

[29] Cyril Goutte and Eric Gaussier. 'A Probabilistic Interpretation of Precision, Recall and F-Score, with Implication for Evaluation'. In: vol. 3408. Apr. 2005, pp. 345–359. ISBN: 978-3-540-25295-5. DOI: 10.1007/978-3-540-31865-1_25.

[30] Daumé III Hal. *A Course in Machine Learning*. http://ciml.info/dl/v0_99/ciml-v0_99-ch08.pdf. (Accessed on 05/31/2021). Jan. 2017.

[31] Wang Haohan and Raj Bhiksha. *1702.07800.pdf*. https://arxiv.org/pdf/1702.07800.pdf. (Accessed on 05/31/2021). Mar. 2017.

[32] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV].

[33] Kaiming He et al. *Identity Mappings in Deep Residual Networks*. 2016. arXiv: 1603.05027 [cs.CV].

[34] Jie Hu et al. *Squeeze-and-Excitation Networks*. 2019. arXiv: 1709.01507 [cs.CV].

[35] Nathalie Jeans. *How I Classified Images With Recurrent Neural Networks*. https://medium.com/@nathaliejeans/how-i-classified-images-with-recurrent-neural-networks-28eb4b57fc79. (Accessed on 05/31/2021). Jan. 2019.

[36] Dag Johansen et al. 'Search-based composition, streaming and playback of video archive content'. In: *Multimedia Tools and Applications* 61.2 (Oct. 2012), pp. 419–445. ISSN: 1573-7721. DOI: 10.1007/s11042-011-0847-5. URL: https://doi.org/10.1007/s11042-011-0847-5.

[37] Will Kay et al. 'The Kinetics Human Action Video Dataset'. In: (May 2017). URL: https://arxiv.org/abs/1705.06950 (visited on 13/06/2020).

[38] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG].

[39] Harilaos Koumaras et al. 'Shot boundary detection without threshold parameters'. In: *J. Electronic Imaging* 15 (Apr. 2006), p. 020503. DOI: 10.1117/1.2199878.

[40] Alex Krizhevsky, Ilya Sutskever and Geoffrey Hinton. 'ImageNet Classification with Deep Convolutional Neural Networks'. In: *Neural Information Processing Systems* 25 (Jan. 2012). DOI: 10.1145/3065386.

[41] Tomei Matteo et al. *RMS-Net: Regression and Masking for Soccer Event Spotting*. https://arxiv.org/pdf/2102.07624v1.pdf. (Accessed on 05/29/2021). Feb. 2021.

[42] James McCaffrey. *How to Do Neural Network Glorot Initialization Using Python - Visual Studio Magazine*. https://visualstudiomagazine.com/articles/2019/09/05/neural-network-glorot.aspx. (Accessed on 05/07/2021). May 2019.

[43] Microsoft. 'Attention spans'. In: (2015). URL: https://dl.motamem.org/microsoft-attention-spans-research-report.pdf.

[44] Rabia Minhas et al. 'Shot Classification of Field Sports Videos Using AlexNet Convolutional Neural Network'. In: *Applied Sciences* 9 (Jan. 2019), p. 483. DOI: 10.3390/app9030483.

[45] Maryam Mohsin. '10 TikTok Statistics That You Need to Know in 2020 [Infographic]'. In: (2020). URL: https://www.oberlo.com/blog/tiktok-statistics.

[46] 'More than half the world watched record-breaking 2018 World Cup'. In: (2018). URL: https://www.fifa.com/worldcup/news/more-than-half-the-world-watched-record-breaking-2018-world-cup.

[47] Kjell Gronhaug Morten Heide. *Respondents' Moods As a Biasing Factor in Surveys: an Experimental Study*. https://www.acrwebsite.org/volumes/7218/volumes/v18/NA-18. (Accessed on 26/05/2021). 1991.

[48] 'Neural Style Transfer'. In: *ReNom* (). URL: http://www.renom.jp/notebooks/tutorial/image_processing/neural-style-transfer/notebook.html (visited on 15/04/2021).

[49] Olav A. Norgård Rongved et al. 'Real-Time Detection of Events in Soccer Videos using 3D Convolutional Neural Networks'. In: *2020 IEEE International Symposium on Multimedia (ISM)*. 2020, pp. 135–144. DOI: 10.1109/ISM.2020.00030.

[50] *NVIDIA DGX-2 Datasheet*. https://www.nvidia.com/content/dam/en-zz/Solutions/Data-Center/dgx-1/dgx-2-datasheet-us-nvidia-955420-r2-web-new.pdf. (Accessed on 06/01/2021).

[51] Pier Paolo. 'SVM: Feature Selection and Kernels'. In: (2019). URL: https://towardsdatascience.com/svm-feature-selection-and-kernels-840781cc1a6c (visited on 17/02/2021).

[52] DENNING PETER J. et al. 'COMPUTING AS A DISCIPLINE'. In: (1989). URL: https://dl.acm.org/doi/pdf/10.1145/63238.63239.

[53] 'Premier League global audience on the rise'. In: *Premier League* (4th July 2019). URL: https://webcache.googleusercontent.com/search?q=cache:u46S5PBsEKgJ:https://www.premierleague.com/news/1280062+%5C&cd=3%5C&hl=no%5C&ct=clnk%5C&gl=no (visited on 14/04/2021).

[54] Muhammad Rafiq et al. 'Scene Classification for Sports Video Summarization Using Transfer Learning'. In: *Sensors* 20 (Mar. 2020), p. 1702. DOI: 10.3390/s20061702.

[55] Arnau Raventos et al. *Automatic Summarization of Soccer Highlights Using Audio-visual Descriptors*. 2014. arXiv: 1411.6496 [cs.IR].

[56] R. Ren and J Jose. 'Football Video Segmentation Based on Video Production Strategy'. In: (2005). URL: https://www.researchgate.net/publication/221397642_Football_Video_Segmentation_Based_on_Video_Production_Strategy.

[57] Olav Rongved. 'Automatic event detection in soccer videos'. In: (2020). URL: http://home.ifi.uio.no/paalh/students/OlavRongved.pdf (visited on 31/08/2020).

[58] Olga Russakovsky et al. 'ImageNet Large Scale Visual Recognition Challenge'. In: (2015). URL: https://arxiv.org/pdf/1409.0575.pdf.

[59] Ramprasaath R. Selvaraju et al. 'Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization'. In: *International Journal of Computer Vision* 128.2 (Oct. 2019), pp. 336–359. ISSN: 1573-1405. DOI: 10.1007/s11263-019-01228-7. URL: http://dx.doi.org/10.1007/s11263-019-01228-7.

[60] Gunnar A. Sigurdsson, Olga Russakovsky and Abhinav Gupta. 'What Actions are Needed for Understanding Human Actions in Videos?' In: (Aug. 2017). URL: https://arxiv.org/abs/1708.02696 (visited on 31/08/2020).

[61] K. Simonyan and A. Zisserma. 'Two-Stream Convolutional Networks for Action Recognition in Videos'. In: (Nov. 2014). URL: https://arxiv.org/abs/1406.2199 (visited on 13/06/2020).

[62] Karen Simonyan and Andrew Zisserman. *Very Deep Convolutional Networks for Large-Scale Image Recognition*. 2015. arXiv: 1409.1556 [cs.CV].

[63] Tomáš Souček. 'Deep Learning-Based Approaches for Shot Transition Detection and Known-Item Search in Video'. In: (2019).

[64] Tomáš Souček and Jakub Lokoč. *TransNet V2: An effective deep network architecture for fast shot transition detection*. 2020. arXiv: 2008.04838 [cs.CV].

[65] *Sports fans take fandom to new levels with online video - Think with Google*. https://www.thinkwithgoogle.com/consumer-insights/sports-fans-video-insights/. (Accessed on 05/26/2020). Jan. 2018.

[66] Christian Szegedy et al. 'Going deeper with convolutions'. In: (2014). URL: https://arxiv.org/abs/1409.4842.

[67] Christian Szegedy et al. *Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning*. 2016. arXiv: 1602.07261 [cs.CV].

[68] Christian Szegedy et al. *Rethinking the Inception Architecture for Computer Vision*. 2015. arXiv: 1512.00567 [cs.CV].

[69] Y. Tabii and R. O. H. Thami. 'A New Method for Soccer Video Summarizing Based on Shot Detection, Classification and Finite State Machine'. In: (Mar. 2009).

[70] Shitao Tang et al. *Fast Video Shot Transition Localization with Deep Structured Models*. 2018. arXiv: 1808.04234 [cs.CV].

[71] *tf.keras.Model, TensorFlow Core v2.5.0*. https://www.tensorflow.org/api_docs/python/tf/keras/Model. (Accessed on 05/20/2021).

[72] *The Difference Between AI, Machine Learning, and Deep Learning? | NVIDIA Blog*. https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/. (Accessed on 09/01/2020). 2016.

[73] Suramya Tomar. 'Converting video formats with FFmpeg'. In: *Linux Journal* 2006.146 (2006), p. 10.

[74] Du Tran et al. 'Learning Spatiotemporal Features with 3D Convolutional Networks'. In: (Oct. 2015). URL: https://arxiv.org/abs/1412.0767 (visited on 13/06/2020).

[75] *TRECVID 2019 Guidelines*. https://www-nlpir.nist.gov/projects/tv2019/data.html. (Accessed on 05/06/2021). 2019.

[76] Rikiya Yamashita et al. *Convolutional neural networks: an overview and application in radiology | Insights into Imaging | Full Text*. https://insightsimaging.springeropen.com/articles/10.1007/s13244-018-0639-9. (Accessed on 05/31/2021). June 2018.

[77] *YouTube for Press*. https://www.youtube.com/about/press/. (Accessed on 05/03/2021). 2020.

[78] Xu Yun, Zomer Simeone and Brereton Richard G. 'Support Vector Machines: A Recent Method for Classification in Chemometrics'. In: *Critical Reviews in Analytical Chemistry* 36.3-4 (2006), pp. 177–188. DOI: 10.1080/10408340600969486. eprint: https://doi.org/10.1080/10408340600969486. URL: https://doi.org/10.1080/10408340600969486.

[79] Hossam Zawbaa et al. 'Event Detection Based Approach for Soccer Video Summarization Using Machine learning'. In: *International Journal of Multimedia and Ubiquitous Engineering (IJMUE)* 7 (Jan. 2012). URL: https://www.researchgate.net/publication/232252059_Event_Detection_Based_Approach_for_Soccer_Video_Summarization_Using_Machine_learning.

[80] Hossam Zawbaa et al. 'Machine Learning-Based Soccer Video Summarization System'. In: vol. 263. Jan. 2011. ISBN: 978-3-642-27185-4. DOI: 10.1007/978-3-642-27186-1_3.