Live Commenting of Football Games Using Machine Learning and Natural Language Generation

Marcus Grönvall

Master's thesis
2021:E5

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

Live Commenting of Football Games Using Machine Learning and Natural Language Generation

Marcus Grönvall

March, 2021

Abstract

In this degree project, automatic live comments of the events in a football game are generated in text format. The events are detected using machine learning, where CNNs are fit using audio recordings and player positions of games. Suitable features are extracted, where several models are fit to detect different types of events. The results indicate that the detections of sound powers and referee whistles are sensitive to the arena, where difficult to determine the event of the referee whistle. However, ongoing attacks are detected accurately. The detected events are commented using natural language generation, where the comments are generated using data-to-text generation. The results indicate that the complexity of the comments is sensitive to the information able to be extracted of the events.

Acknowledgements

This degree project would not have been accomplished without the involvement of several people. I would like to thank my supervisors Klas Josephson and Christian Balkenius, for their advice and support throughout the work. I would also like to thank Carl Larsson, who gave me the opportunity of completing the degree project at Spiideo. Thank you.

Contents

1 Introduction **6**

1.1 Background 6

1.2 Motivation 6

1.3 Objective 7

1.4 Outline 7

1.5 Previous Work 8

2 Machine Learning **9**

2.1 Convolutional Neural Network 10

2.1.1 Architecture 10

2.1.2 Activation Function 12

2.2 Learning Procedure 13

2.2.1 Loss Function 14

2.2.2 Optimizer 15

2.2.3 Class weights 16

2.2.4 Metric 16

2.3 Evaluation 17

2.4 Feature Extraction 20

2.5 Normalization 21

2.6 Hard Negative Mining 22

2.7 Constraints 22

3 Natural Language Generation **23**

3.1 Data-to-Text Generation 23

3.1.1 Text Planner 24

3.1.2 Sentence Plan 25

3.1.3 Realizer 26

3.2 Constraints 28

4 Data **29**
4.1 Tags . . . 29
4.2 Audio Recordings . . . 29
4.3 Player Positions . . . 30
4.3.1 Foreground Ratio . . . 30
4.3.2 Player Tracks . . . 30

5 Feature Extractions **32**
5.1 Spectrogram . . . 32
5.2 Mel-Frequency Cepstral Coefficients . . . 32
5.3 Position Feature . . . 33
5.3.1 Motion fields . . . 33
5.3.2 Concatenating feature extractions . . . 33
5.4 Parameter settings . . . 34

6 Sampling **38**
6.1 Sample . . . 38
6.2 Sample Generator . . . 39

7 Datasets **41**
7.1 Attack Datasets . . . 41
7.2 Whistle Dataset . . . 43
7.3 Hard Negative Mining of Whistle Dataset . . . 44
7.4 Whistle Event Dataset . . . 46

8 Detector **48**
8.1 Sound Detection . . . 48
8.2 Attack Detection . . . 48
8.3 Whistle Detection . . . 49
8.4 Whistle Event Determination . . . 49

9 Commentator **52**
9.1 Commentator State . . . 53
9.1.1 Content . . . 53
9.1.2 Status . . . 54
9.2 Game Information . . . 58
9.2.1 Team Location Determination . . . 58
9.2.2 Game Statistics . . . 58
9.3 NLG System . . . 59
9.3.1 Text Plan . . . 59
9.3.2 Sentence Plan . . . 60

9.3.3 Realization 60

10 Experiment Setup **61**
10.1 Machine Learning 61
10.1.1 Model Architectures 61
10.1.2 Learning Procedure 63
10.1.3 Model Evaluation 63
10.2 Natural Language Generation 64
10.2.1 NGL System 64
10.2.2 System Evaluation 64

11 Results **65**
11.1 Machine Learning Models 65
11.1.1 Attack Models 65
11.1.2 Whistle Models 69
11.1.3 Whistle Event Model 72
11.2 NLG System 75

12 Discussion **85**
12.1 Detector 85
12.2 Commentator 88
12.3 Computational Speed 89
12.4 Future Work 90

13 Conclusion **92**

A Appendix A **93**

B Appendix B **102**

C Appendix C **109**

1 Introduction

1.1 Background

Around the world, human voiced live commenting of sports has been around since radio broadcasting in the early 1920's. The first broadcast took place on April 11, 1921, from Motor Square Garden in Pittsburgh, where the 10-round no-decision boxing match between Johnny Ray and Johnny Dundee was commented. Since then, the commentators have gone from being broadcast on radio and television to streaming services over the internet. [1]

During the past resent years, the applications for machine learning and natural language generation have grown rapidly with the increase of big data, computational power and effective algorithms. Applications such as event detection using deep learning and text generation using data-to-text generation, have become feasible in practice. For example, this includes the events in a football game, which calls for automatic live commenting. [2]

1.2 Motivation

At Spiideo, the Swedish sports video analysis company based in Malmö, a streaming service is provided for their costumers in different sports. The company has identified a growing market for commenting games not normally commented by human voiced commentators, such as games in lower divisions and youth games. Therefore, Spiideo is set to develop automatic live commenting of games, where their ambition motivates this degree project.

The degree project is intended to be used as a basis of how automatic live commenting may be implemented, paving way for more advanced implementations by Spiideo in the future.

1.3 Objective

The main objective of the degree project is to generate automatic live comments of the events in a sports game, where the method is to use machine learning for event detection and natural language generation for commenting in text format. The project is constrained to football games in Sweden, thus football related events are at focus. Due to broadcasting requirements of the streaming service, the aim is to generate comments at relatively low cost. Therefore, provided data such as audio recordings, player positions and tags of event information are used, excluding direct analysis on video data.

The research issues of the degree project are given in accordance with the following list.

- Is it possible to automatically detect the events in a football game using machine learning?
- Is it possible to generate descriptive comments of the events in a football game using natural language generation?

1.4 Outline

In the first part of the project, an event detector is constructed. The provided data are used to fit machine learning models, where suitable features are extracted of the audio recordings and player positions. The tags are used for classification, where events not included in the provided tags are tagged manually if necessary. Several models are fit to detect different types of events, where initially if an event is occurring and further what event if possible. Moreover, additional properties of the provided data are extracted, where the audio recordings are used to detect changes in sound power and player positions changes in movement.

In the second part of the project, a commentator is constructed. The detected events are commented using natural language generation, where the comments are generated using data-to-text generation. To increase the complexity of the comments, information of the provided data is extracted. The tags are used to determine the team locations in the field and the player positions the general movement of the players and player sprints. In addition, statistics of the detected events and tags are gather during a game, such as the number of goals, shots, corners, etc.

1.5 Previous Work

For event detection using machine learning, most previous works concerning the events in a football game are based on broadcasting material. Replays have been extracted from video data, which were recognized by the camera changing view or focus, thereafter the events of the replays were classified [3]. Video data have also been used to detect goals, which were detected by the blurriness of the video as the camera moves and zooms in connection with a goal [4]. Audio keywords have been extracted from audio data, which were recognized by the referee whistle and excitement of the commentator and audience, thereafter the events of the keywords were classified using a rule-based system [5]. However, some works are not based on broadcasting material. Player positions estimated from video data have been used to detect different events, which decreased the computational cost compared to direct video analysis [6].

For text generation using natural language generation, previous works generating both summaries and event comments of a football game have been published. Syntactic templates have been used to generate summaries in speech, which used tabular information as input [7]. Interactive multi-agent systems have been used to generate comments of different events, where the events were simulated from a server [8].

2 Machine Learning

In the following narrative on machine learning, mostly a descriptive outline of the subject is provided. Most content is cited from *Machine learning: A Probabilistic Perspective* by Murphy [9], *Deep learning* by Goodfellow et al. [10], *Pattern Recognition and Machine Learning* by Bishop [11] and *Deep Learning with Python* by Chollet [12], which offer more details from a mathematical point of view and how to implement machine learning algorithms.

Machine learning (ML) is defined as methods that automatically recognize patterns in data, which can be used to predict on future data. The goal of ML is to learn a mapping from input to output, which is performed by observing a labeled set of inputs and targets.

Deep learning is a subfield to ML, which emphasises on learning successive layers of meaningful representations of the data, where the term deep refers to multiple intermediate layers. The structure is modeled by artificial neural networks (ANNs), where the layers are represented by a series of stacked functional transformations expressed in terms of each other. The functional transformation of a layer is modeled by a set of units, where each unit is represented by a linear combination of the layer input and trainable weights applied by a non-linear activation function. This allows a complex model built on simpler concepts, where each layer provides a new representation of the data. An ANN can be visualized as a graph of connected units, where the weights determine the mapping from input to output. A graph of a simple ANN, consisting of one intermediate layer, is given by Figure 2.1.

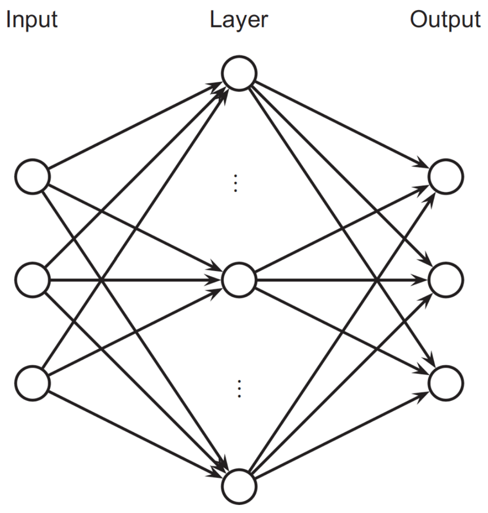


Figure 2.1: Graph of a simple ANN, consisting of one intermediate layer. [9]

2.1 Convolutional Neural Network

Convolutional neural networks (CNNs) is a branch of ANNs, where the input to some layers consists of mathematical convolutions. Each unit of a convolutional layer is represented by a linear combination of the convolutions and the weights, as opposed to the full input as the layer of a standard ANN. This reduces the number of weights significantly, but does not affect the ability to learn generalized patterns. The convolutions are applicable to inputs of arbitrary dimension, where examples are sequences in one, images in two and videos in three dimensions. The convolutions are also translation invariant, thus local patterns can be learned independently of their spatial locations. CNNs identify hierarchies of patterns as the spatial complexity increases for each convolutional layer, which allows a network to learn more abstract patterns.

2.1.1 Architecture

The architecture of a CNN consists of a series of alternately stacked convolutional and maxpooling layers, which is followed by a flatten layer. The architecture is ended by a series of stacked fully-connected layers, where the output layer achieves classification.

Convolutional Layer

The convolutions of a convolutional layer operate over tensors, which are referred to as feature maps. The convolution operation swipes a window over the input feature map and extracts patches, where a patch can be extracted every step or given step size using strides. Each patch is transformed by a tensor dot product with a kernel, where the assembly results in the output feature map. Considering the initial convolutional layer of a CNN, the axes of the input feature map correspond to the spatial dimensions and channel depth of the input. The depth axis of the output feature map corresponds to the number of filters, where the filters represent the learned patterns. A visualization of how the convolutions of a convolutional layer work, for an input image of two channels, is given by Figure 2.2.

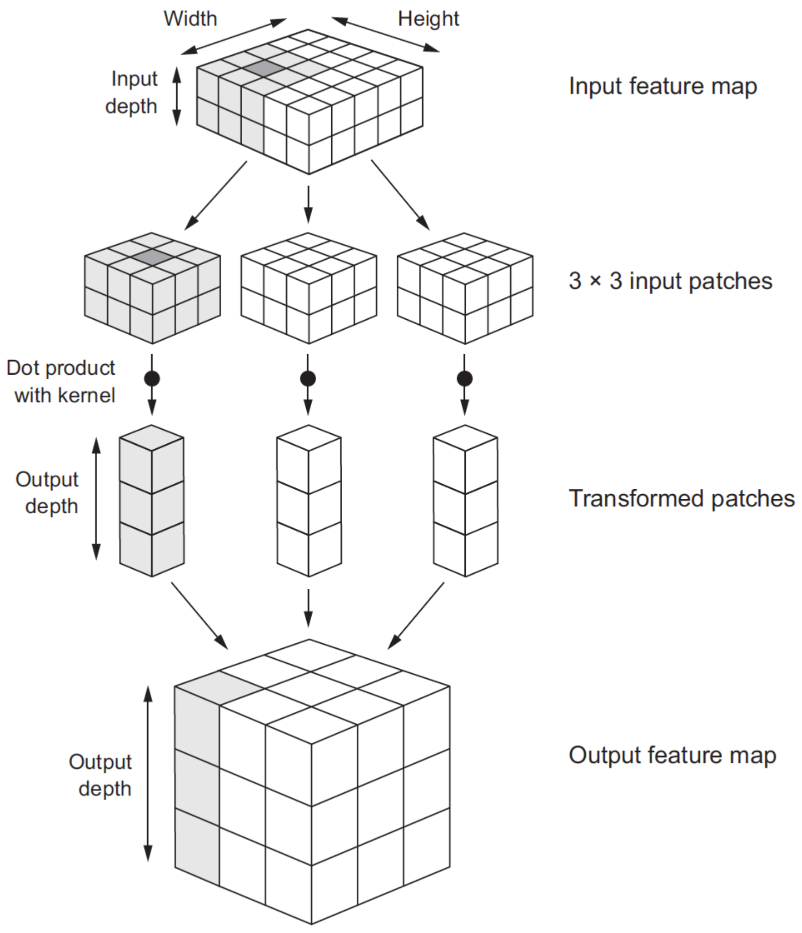


Figure 2.2: Visualization of how the convolutions of a convolutional layer work, for an input images of two channels. Notice, 3 × 3 input patches are extracted every step. [12]

Maxpooling Layer

As convolutional layers are stacked in a CNN, the sizes of the feature maps increase as the number of filters grows. To keep the feature maps at reasonable sizes and to allow subsequent convolutional layers to experience great spatial extent, a convolutional layer may be followed by a maxpooling layer. The maxpooling operation swipes a window over the feature map and extracts patches, where a patch can be extracted every step or given step size using strides. The local maximums of the depth axis for each patch are calculated, where the assembly results in the downsampled feature map. A maxpooling layer speeds up computation as the size of the feature map decreases, but information is lost due to the downsampling.

Flatten Layer

The resulting feature map of successive convolutional and maxpooling layers is filtered through a flatten layer. The flatten operation transforms the feature map tensor into a one dimensional vector, which is the valid input shape of a fully-connected layer.

Fully-Connected Layer

The fully-connected layer is essentially the layer of an ANN, where all units between successive layers are connected as visualized in Figure 2.1. Depending on if a layer is intermediate or output, the activation function is selected in accordance with Section 2.1.2. A fully-connected layer identifies global patterns and is not translation invariant, thus requires to learn a pattern anew if appearing at another spatial location.

2.1.2 Activation Function

Each unit of a layer is applied by a non-linear activation function, which transforms the linear combination of the layer input and trainable weights into a non-linear function. This to extend the hypothesis space of mappings from input to output and prevent a network from collapsing into a linear regression model. However, applying non-linear activation functions, the problem of finding the optimal weights becomes non-convex, but the benefits outweigh the harms. There are different types of non-linear activation functions, where the choice depends on the purpose of the layer. The most essential demand is to maintain the function of a unit differentiable.

ReLU

For intermediate layers, consisting of a convolutional or fully-connected layer, the activation function of the units is usually selected as the Rectified Linear Unit (ReLU) given by

$$\sigma(x) = \max(0, x), \tag{2.1}$$

where x is the linear combination of the input and weights of the unit, which sets the output to zero if negative.

Sigmoid

For binary classification problems, the activation function of the output unit is selected as the sigmoid function given by

$$\sigma(x) = \frac{1}{1 + e^{-x}}, \tag{2.2}$$

where x is the linear combination of the input and weights of the unit, which converts the output into a probability.

Softmax

For multi-class classification problems, the activation function of the output units is selected as the softmax function given by

$$\sigma_k(\boldsymbol{x}) = \frac{e^{x_k}}{\sum_{i=1}^{n} e^{x_i}}, \tag{2.3}$$

where x_i is the linear combination of the input and weights of the unit i and n the number of units, which converts the output of the unit k into a probability.

2.2 Learning Procedure

Before initiating the learning procedure of a neural network, the labeled set of inputs and targets is split into training, validation and test sets. The training set includes the samples a network is observing when learning a mapping from input to output, the validation set the samples a network is validated against during the learning procedure and the test set the samples a fit model is tested against when the learning procedure is accomplished. This is necessary to find the model of optimal generalization, reduce over-fitting and test the performance of a fit model against unobserved samples.

During the learning procedure, the output prediction of a network given an input is calculated using the forward-propagation algorithm. Forward-propagation feeds the input forward through the layers, where the output is calculated given the current weights of the network. The performance of the network is measured using a loss function, thereafter the weights are updated by an optimizer using the back-propagation algorithm. Back-propagation feeds the loss score backward through the layers, where the loss is slightly reduced using an optimization method. In general, the weight update is performed for a batch of samples, where the procedure is repeated for a sufficient number of epochs. This until the loss is reduced enough and the network has learned a mapping from input to output. A visualization of the learning procedure of a neural network is given by Figure 2.3.

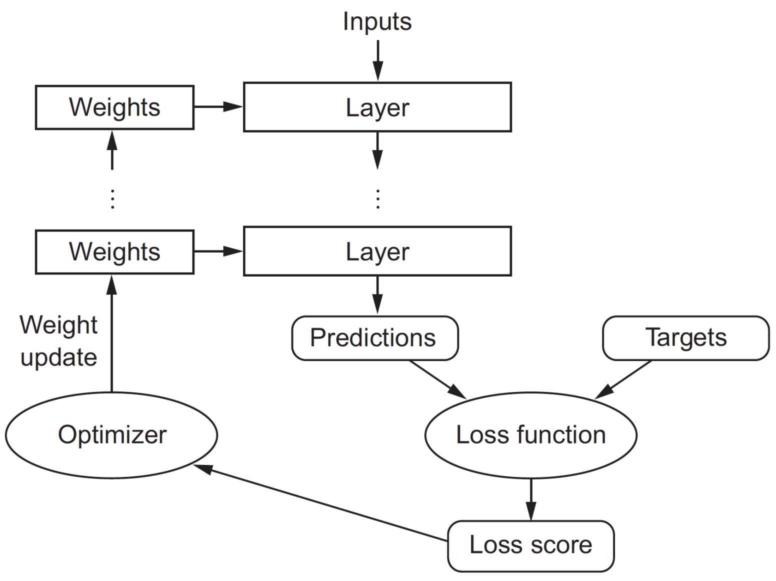


Figure 2.3: Visualization of the learning procedure of a neural network. [12]

2.2.1 Loss Function

The performance of a network during the learning procedure is measured using a loss function, which calculates the distance between the labeled targets and corresponding predictions. There are different types of loss functions, where the choice depends on the classification problem.

Binary Cross-Entropy

For binary classification problems, the loss function is selected as binary cross-entropy given by

$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i + (1 - y_i)\log(1 - \hat{y}_i), \tag{2.4}$$

where $\boldsymbol{y}$ are the targets, $\hat{\boldsymbol{y}}$ the predictions and n the number of samples in the batch.

Categorical Cross-Entropy

For multi-class classification problems, the loss function is selected as categorical cross-entropy given by

$$L(\boldsymbol{y}, \hat{\boldsymbol{y}}) = -\frac{1}{n}\sum_{i=1}^{n} y_i \log \hat{y}_i, \tag{2.5}$$

where $\boldsymbol{y}$ are the targets, $\hat{\boldsymbol{y}}$ the predictions and n the number of samples in the batch.

2.2.2 Optimizer

The weight update during the learning procedure is performed by an optimizer, which slightly reduces the loss by adjusting the weights to fit the mapping from input to output. Since the problem of finding the optimal weights is non-convex, local minimums can be found analytically. The objective is to find a global minimum among the local minimums, while difficult to determine if a minimum is global or local and easy to get stuck in a local minimum.

The optimization problem is solved by gradient-based optimization methods, which calculate the gradient with respect to the weights and reduce the loss by moving in the opposite direction of the gradient. The objective of finding a global minimum is not to determine its exact location, but rather settle close enough to speed up computation. Therefore, the step size is important. Large step sizes may overstep the location and cause a randomized search. Small step sizes increase the number of iterations and endanger getting stuck in a local minimum. This is prevented by applying adaptive learning rates to the gradient-based optimization methods. There are different types of adaptive optimization methods, where there is no consensus on which is most suitable for a general neural network [10].

Adaptive Moment Estimation

The Adaptive Moment Estimation (ADAM) is an adaptive optimization method, which uses adaptive learning rates for each of the weights. ADAM estimates the exponentially decaying averages of the first and second moments of the past normalized gradients, which introduce the adaptive learning rates. On the other hand, the moment estimates are biased towards zero if calculated directly, which is avoided by estimating the bias-corrected moments instead. As a metaphor, the first moment can be visualized as a ball rolling down a slope, where the second moment illustrates friction. ADAM tends to prefer flat minimums in the error surface, but modifies the step size suitably in favor of finding a global minimum.

2.2.3 Class weights

If the distribution of labeled targets is imbalanced among the classes, the neural network endangers to be biased towards the dominant class during the learning procedure. This is a problem if the targets are heavily imbalanced, which affects the recognition results negatively. In case of weak imbalance, the problem may be prevented by applying class weights to each of the targets in accordance with

$$w_i = \frac{n_{targets}}{n_{classes} \cdot n_i}, \tag{2.6}$$

where $n_{targets}$ is the number of targets, $n_{classes}$ the number of classes and n_i the number of targets for class i.

2.2.4 Metric

In addition to the loss, a metric is used to monitor during the learning procedure and evaluate the test set when the learning procedure is accomplished. The metric measures the prediction ability of the model. During the learning procedure, the model of optimal generalization corresponds to when the loss and metric of the training and validation sets cease to improve and the gaps between the training and validation sets are minimal. At this point, it is appropriate to terminate the learning procedure. Furthermore, a neural network requires a sufficient number of weights to reduce under-fitting and not letting the learning procedure proceed further to reduce over-fitting. There are different types of metrics, where the choice depends on the distribution of labeled targets among the classes in the sets of samples.

Accuracy

For balanced classification problems, the metric is usually selected as the accuracy given by

$$A = \frac{n_{tp}}{n}, \tag{2.7}$$

where n_{tp} is the number of true positives and n the number of samples in the set. The accuracy measures the fraction of inputs that are correctly classified by the model.

F1 Score

For imbalanced classification problems, the metric may be selected as the F1 score given by

$$F_1 = 2\frac{P \cdot R}{P + R}, \tag{2.8}$$

where P is the precision and R the recall given by

$$P = \frac{n_{tp}}{n_{tp} + n_{fp}}, \tag{2.9}$$

$$R = \frac{n_{tp}}{n_{tp} + n_{fn}}, \tag{2.10}$$

where n_{tp} is the number of true positives, n_{fp} the number of false positives and n_{fn} the number of false negatives. The F1 score measures the harmonic mean between precision and recall by the model. If the distribution of targets is imbalanced, the accuracy provides a skew measure of the prediction ability of the model. This especially if the bias is large, which is avoided by using the F1 score instead.

2.3 Evaluation

When finding the optimal machine learning model of a mapping from input to output, usually several models are fit using different structures and learning parameters. There are different types of methods for evaluating and comparing fit models, where the choice depends on the distribution of labeled targets among the classes in the sets of samples.

Confusion Matrix

The distribution of predictions for a given threshold, can be visualized by a confusion matrix in accordance with Figure 2.4. The notations are given by

$$\text{TPR} = \frac{n_{tp}}{n_{tp} + n_{fn}}, \tag{2.11}$$

$$\text{TNR} = \frac{n_{tn}}{n_{fp} + n_{tn}}, \tag{2.12}$$

$$\text{FPR} = \frac{n_{fp}}{n_{fp} + n_{tn}}, \tag{2.13}$$

$$\text{FNR} = \frac{n_{fn}}{n_{tp} + n_{fn}}, \tag{2.14}$$

where TPR is the true positive rate, TNR the true negative rate, FPR the false positive rate, FNR the true negative rate, n_{tp} the number of true positives, n_{tn} the number of true negatives, n_{fp} the number of false positives and n_{fn} the number of false negatives. A model separating the classes perfectly has zero FPR and FNR, while a contrary model has zero TPR and TNR. The concept of confusion matrices can be extended to multi-class classification problems, which is performed by using the number of predictions for each class in a similar manner.

Receiver Operating Characteristic Curve

The TPR and FPR in Equation 2.11 and 2.13 respectively, can be visualized for a number of thresholds by a Receiver Operating Characteristics (ROC) curve in accordance with Figure 2.5a. A model separating the classes perfectly has a threshold in the top left corner, while a contrary model has a threshold on or below the diagonal. The area under the curve (AUC) summarizes the ROC curve, where the optimal value is 1. The optimal threshold balancing the TPR and FPR, can be obtained by calculating the geometric mean for each threshold in accordance with

$$\text{G-mean} = \sqrt{\text{TPR}(1 - \text{FPR})}, \tag{2.15}$$

where the optimal threshold corresponds to the threshold of maximum geometric mean. ROC curves are informative for balanced classification problems, where the distribution of targets is approximately even among the classes. The concept of ROC curves can be extended to multi-class classification problems, which can be performed by using the one label versus the rest approach.

Precision-Recall Curve

The precision and recall in Equation 2.9 and 2.10 respectively, can be visualized for a number of thresholds by a Precision-Recall curve in accordance with Figure 2.5b. A model separating the classes perfectly has a curve hugging the top, while a contrary model has a curve hugging the bottom. The area under the curve summarizes the Precision-Recall curve, which is estimated as the mean of precisions. The optimal threshold maximizing the F1 score, can be obtained by calculating the F1 score for each threshold in accordance with Equation 2.8, where the optimal threshold corresponds to the threshold of maximum F1 score. Precision-Recall curves are informative for imbalanced classification problems, where the distribution of targets is approximately uneven among the classes. The concept of Precision-Recall curves can be extended to multi-class classification problems, which can be performed by using the one label versus the rest approach.

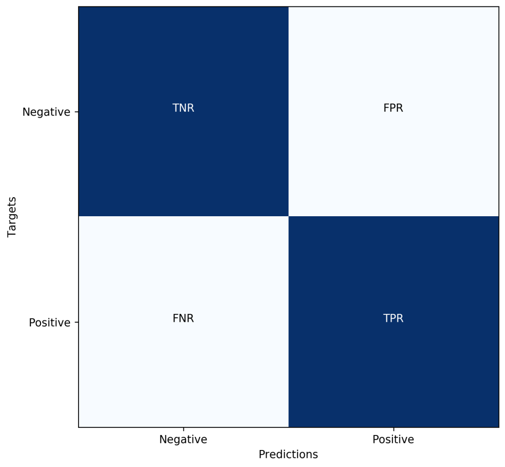
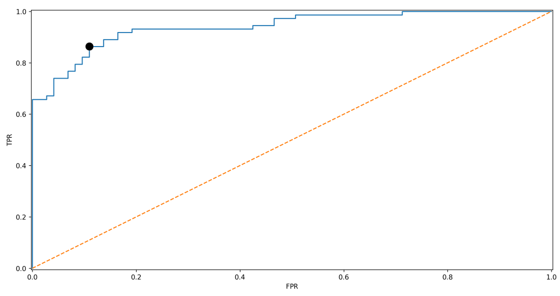
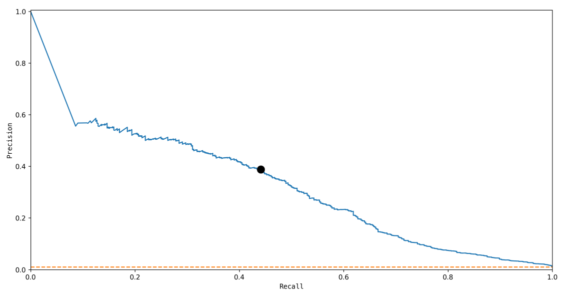


Figure 2.4: Visualization of the confusion matrix.



(a) ROC curve.



(b) Precision-Recall curve.

Figure 2.5: Visualization of the ROC and Precision-Recall curves. The black dots indicate the corresponding optimal thresholds.

2.4 Feature Extraction

Before fitting a machine learning model, some feature extraction of the inputs may be required. There are numerous different types of feature extractions, where the choice depends on the representation of the original inputs and what the model is intended to learn.

Spectrogram

Since some of the provided data consist of audio recordings, the spectrogram may be sufficient as feature extraction. The spectrogram of an audio segment is a visual representation of the power spectral density as the segment changes over time, where the pixels in the image correspond to the intensities in the time-frequency domain. [13]

The spectrogram of an audio segment is extracted by performing the Short-Time Fourier Transform (STFT), thereafter calculating the intensities. The segment is divided into shorter time frames, typically in range $23 - 93$ milliseconds, depending on the type of audio recording. Each frame is applied by a window function, where the Hanning window is adequate for most applications in audio signal processing. The purpose of the window function is to reduce spectral leakage, but sets both ends of a frame close to zero. This causes information loss at the edges of adjacent frames, which is prevented by framing the segment with some overlap, typically as a quarter of a frame. Each windowed frame is applied by the Discrete Fourier Transform (DFT) in accordance with

$$X_k = \sum_{n=0}^{N-1} x_n e^{-\frac{i2\pi}{N}kn}, \quad k = 0, 1, ..., N-1, \tag{2.16}$$

where x_n is an amplitude and N the number of amplitudes in the sequence of a windowed frame. The assembly after performing the DFT corresponds to the complex valued STFT. The complexity is handled by calculating the intensities using the absolute value, which results in the extracted spectrogram. [13]

Mel-Frequency Ceptral Coefficients

As an extension of the spectrogram feature, the Mel-Frequency Cepstral Coefficients (MFCC) may be sufficient as feature extraction. The MFCC of an audio segment is a visual representation of the Mel-Frequency Cepstrums (MFCs) as the segment changes over time, where the pixels in the image correspond to the coefficient values in the time-cepstrum domain. [14]

The MFCC of an audio segment is extracted by calculating the MFCs, which collectively build up the MFCC. The MFCs are calculated by filtering the spectrogram through the Mel-filterbank, which divides the spectrogram into frequency bands and calculates the summarized intensities for each filter. The frequency bands are equally spaced on the Mel-scale, which is a scale of pitches judges by listeners to be equal in distance. Due to the human auditory, the Mel-scale is non-linear, where the distance between pitches increases exponentially with frequency. Since the frames are set to overlap performing the STFT, adjacent frames are highly correlated. This is prevented by applying the Discrete Cosine Transformation (DCT) in accordance with

$$X_k = \sum_{n=0}^{N-1} x_n \cos\left(\frac{\pi}{N}\left(n + \frac{1}{2}k\right)\right), \quad k = 0, 1, ..., N-1, \tag{2.17}$$

where x_n is an intensity and N the number of intensities in the sequence summarized filter intensities. The assembly after applying the DCT corresponds to the MFCs, where the MFCC is chosen equal or as a selection of the extracted MFCs. [14]

2.5 Normalization

In addition, before fitting a machine learning model, the inputs may require normalization if the values are large. This to prevent the model to learn large weights during the learning procedure, which may cause a slow and unstable learning process and in extension affect the recognition results. There are different types of normalization methods, where the choice depends on the representation of the inputs.

Minmax

Applying minmax normalization, the values of the inputs are normalized to a specified range, given a minimum and maximum estimated over the training set. Normalization to range $[0, 1]$ is then given by

$$y = \frac{x - \min(x)}{\max(x) - \min(x)} \tag{2.18}$$

and to range $[-1, 1]$ by

$$y = 2\frac{x - \min(x)}{\max(x) - \min(x)} - 1 \tag{2.19}$$

where x is the input value.

Cepstral Mean-Variance

For automatic speech recognition using the MFCC feature, the MFCCs are recommended to be normalized using Cepstral Mean Variance Normalization (CMVN). CMVN is shown to improve the recognition results in various environments and reduce noise. Applying CMVN, the coefficient values of the MFCCs are standardized for each MFC separately, given the cepstral means and standard deviations estimated over the training set. Normalization is then given by

$$y_i = \frac{x_i - \mu_i(x)}{\sigma_i(x)}, \quad i = 1, 2, ..., n, \tag{2.20}$$

where x_i is the cepstral input value, μ_i the cepstral mean, σ_i the cepstral standard deviation and n is the number of cepstrals. [15]

2.6 Hard Negative Mining

In accordance with *Cost-Optimized Event Detection in Football Video* by Arpe and Ericsson [6], hard negative mining is examined but with some modifications. Given a model fit on initial data, the method is to use the model to predict on new data. The samples the model classifies as false negatives are extracted and added to the the initial data, thereafter a new model is fit to the updated data. The intention is to improve the model to become more robust and accurate of separating the classes, hopefully generating less false positives as well. The exact approach is given in Section 7.3.

2.7 Constraints

If having access to a set of samples consisting of multiple inputs for the same targets, there is a possibility to fit parallel neural networks. This is the case considering both the audio recordings and player positions of the provided data. On the other hand, such networks may be of high computational cost and slow in a live commentator setup. Therefore, to constrain the degree project, parallel neural networks are not examined.

3 Natural Language Generation

In the following narrative on natural language generation, mostly a descriptive outline of the subject is provided. Most content is cited from *Survey of the State of the Art in Natural Language Generation: Core Tasks, Applications and Evaluation* by Gatt and Krahmer [16] and *Building Applied Natural Language Generation Systems* by Reiter and Dale [17], which offer a survey on natural language generation and how to build systems for text generation.

Natural language generation (NLG) is defined as computer systems that automatically generate understandable text from some underlying information. The representation of information input into an NLG system may vary considerably, possibly consisting of semantic representations or non-linguistic data. Therefore, the methods for generating text within a system may vary considerably as well, where the choice depends on the information input and application.

3.1 Data-to-Text Generation

Data-to-text generation is a branch of NLG, where the representation of information input consists of non-linguistic data. As mentioned in the introduction in Section 1.3, the objective of the degree project is to live comment detected events in a football game. Such events may be summarized as non-linguistic data, thus data-to-text generation is implemented and the following outline adapted to the branch and application.

The NLG problem of converting information input into output text can be divided into a number of subproblems, where there are different methods of how to solve each subproblem and in what order. The bound between subproblems may be diffuse, where tasks may be shared and solved together depending on application. The classical approach is to use a modular pipeline architecture, which solves the subproblems in sequential order. An NLG system using a modular pipeline architecture is given by Figure 3.1.

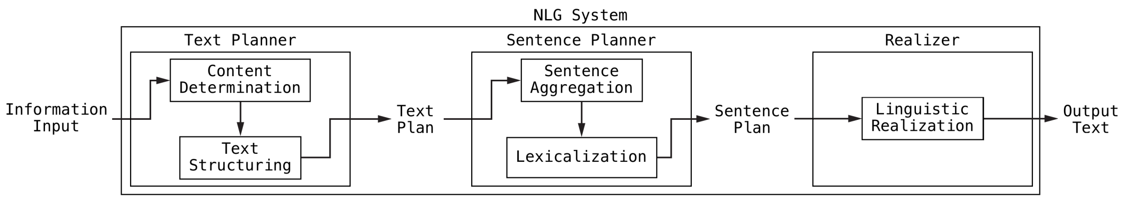


Figure 3.1: NLG system using a modular pipeline architecture.

3.1.1 Text Planner

The text planner is considered as the early strategy process, which includes choices concerning the information input into the NLG system. The text plan is crucial for the development of NLG systems and often intimately connected to the application. The text planner includes solving the subproblems of content determination and text structuring, which constructs a text plan of the information input.

Content Determination

The content determination subproblem consists of deciding which information to include in the text under construction, where typically more information is contained in the input than desired to be expressed. For the NLG problem of live commenting the events in a football game, provided data consisting of audio recordings and player positions are available for whole games. However, only data in connection with the detected events are of interest, thus the event detector is the main part of content determination.

Given the decided information to be included in the text, the information requires to be abstracted into some preverbal messages. One approach is to use attribute-valued matrices, which are able to store the information of the detected events suitably. Assuming the event of a goal is detected, the preverbal message represented as an attribute-valued matrix may be given in accordance with Figure 3.2. Similar representations may be constructed for other events.

Text Structuring

The text structuring subproblem consists of determining in which order the information is presented in the text under construction, where the order is strongly connected to the application. For the NLG problem of live commenting the events in a football game, the order is determined by the temporal order of detected events during the game. Therefore, the event detector is the main part of text structuring.

$$\begin{bmatrix} \text{category:} & \texttt{detection} \\ \text{event:} & \texttt{goal} \\ \text{type:} & \texttt{shot} \\ \text{minute:} & \texttt{38} \\ \text{player:} & \begin{bmatrix} \text{name:} & \texttt{Markus Rosenberg} \\ \text{team:} & \texttt{Malmö FF} \\ \text{score:} & \texttt{1} \end{bmatrix} \\ \text{team:} & \begin{bmatrix} \text{home:} & \begin{bmatrix} \text{name:} & \texttt{Malmö FF} \\ \text{score:} & \texttt{3} \end{bmatrix} \\ \text{away:} & \begin{bmatrix} \text{name:} & \texttt{Helsingborgs IF} \\ \text{score:} & \texttt{0} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure 3.2: Preverbal message represented as an attribute-valued matrix of a goal event.

3.1.2 Sentence Plan

The sentence planner is considered as the late tactics process, which includes choices concerning the linguistics of the text under construction. The sentence planner is often independent of application, thus the subproblems within may be shared between applications. The sentence planner includes solving the subproblems of sentence aggregation and lexicalization, which constructs a sentence plan of the text plan.

Sentence Aggregation

The sentence aggregation subproblem consists of deciding which information to present in individual sentences, where typically all preverbal messages are not required to be expressed in separate sentences or a preverbal message in a single sentence. This in order to generate text more fluid and readable. For the NLG problem of live commenting the events in a football game, one comment is intended to be generated for each detected event separately. Therefore, the sentence aggregation subproblem concerns if to split the information into multiple sentences or not and what information to include in each sentence if separated. Considering the preverbal message of the goal event in Figure 3.2, the information may be separated into multiple sentences of the player scoring and the updated standings of the game. Similar decisions may be considered for other events.

Lexicalization

The lexicalization subproblem consists of determining which words and phrases to express the information of the individual sentences, where the NLG system initiates the conversion to natural language. For the NLG problem of live commenting the events in a football game, the objective is to generate descriptive comments of the detected events. One approach is to decide a word or phrase randomly among a set of alternatives, thus the complexity of the comments depends on the alternatives the system can provide. Assuming a goal is scored, the incident may be phrased as 'to score a goal', 'to have a goal', 'to put the ball in the net', etc. Further, 'to score a goal' is an unfortunate way of expressing an own goal, thus the NLG system may regard for stylistic constraints. Similar sets of alternatives can be constructed for other incidents.

3.1.3 Realizer

The realizer is considered as the final process, which generates the natural language text. The realizer includes solving the subproblem of linguistic realization, which constructs the output text of the sentence plan.

Linguistic Realization

The linguistic realization subproblem consists of combining all the relevant words and phrases into well-formed sentences. In other words, linguistic realization involves ordering the constituents of the individual sentences and generating correct grammatical forms, such as selecting verbs, prepositions and punctuation marks. One approach is to use human handcrafted templates, which fit the mentioned lexicalization method suitably. Templates are well sufficient if the application domain is small, as for the events in a football game. The advantage is that the implementer has full control of the linguistics, thus minimizing grammatical errors. The disadvantage is that templates are labour demanding and constrained to the imagination of the implementer. Considering the preverbal message of the goal event in Figure 3.2, the linguistic realization using sentence aggregation and lexicalization may be given in accordance with Figure 3.3. Similar human handcrafted templates may be constructed for other events.

The output text is generated by deciding a word or phrase randomly among the sets of alternatives of the human handcrafted templates. Considering the linguistic realization of the goal event in Figure 3.3, the comment may be generated in accordance with Figure 3.4. Similar comments may be generated for other events.

$$[(\{\text{minute}\}')] \begin{bmatrix}\texttt{Goal}\\ \texttt{Score}\end{bmatrix} \begin{bmatrix}\texttt{.}\\ \texttt{!}\end{bmatrix}$$

(a) First sentence. Initiates by proclaiming a goal is scored and the minute of occurrence.

$$[\{\text{player.name}\}] \begin{bmatrix}\texttt{scores}\\ \texttt{scores a goal}\\ \texttt{has a goal}\\ \texttt{kicks a goal}\\ \texttt{makes a goal}\\ \texttt{records a goal}\\ \texttt{puts the ball in the net}\end{bmatrix} \begin{bmatrix}\texttt{and does}\\ \texttt{and makes}\\ \texttt{and it is}\\ \texttt{, which is}\end{bmatrix} [\texttt{his } \{\text{player.score}\} \texttt{ goal}] \begin{bmatrix}\texttt{.}\\ \texttt{in the game.}\\ \texttt{for today.}\end{bmatrix}$$

(b) Second sentence. States the scoring player and the corresponding number of goals scored.

$$\begin{bmatrix}\texttt{The } \{\text{team}\} \texttt{ team}\\ \{\text{team.name}\}\end{bmatrix} \begin{bmatrix}\texttt{extends}\\ \texttt{expands}\\ \texttt{increases}\end{bmatrix} \begin{bmatrix}\texttt{the lead}\\ \texttt{the distance}\end{bmatrix} \begin{bmatrix} \\ \texttt{again}\\ \texttt{further}\end{bmatrix} \begin{bmatrix}\texttt{to}\\ \texttt{to the standings}\end{bmatrix} [\{\text{home.score}\} - \{\text{away.score}\}.]$$

(c) Third sentence. States the goal increases the lead and the updated standings of the game. Notice, a word or phrase can be skipped, as shown in the fourth set of alternatives.

Figure 3.3: Linguistic realization using sentence aggregation and lexicalization of the preverbal message of the goal event in Figure 3.2. The curly brackets correspond to the attribute values of the preverbal message.

$$\boxed{\texttt{(38')}}\ \boxed{\texttt{Goal}}\ \boxed{\texttt{!}}\ \boxed{\texttt{Markus Rosenberg}}\ \boxed{\texttt{puts the ball in the net}}\ \boxed{\texttt{, which is}}\ \boxed{\texttt{his first goal}}$$
$$\boxed{\texttt{in the game.}}\ \boxed{\texttt{The home team}}\ \boxed{\texttt{expands}}\ \boxed{\texttt{the lead}}\ \boxed{\texttt{again}}\ \boxed{\texttt{to the standings}}\ \boxed{\texttt{3-0.}}$$

Figure 3.4: Comment generated by randomly deciding a word or phrase among the sets of alternatives of the linguistic realization of the goal event in Figure 3.3.

3.2 Constraints

Automatized commentators do not require a large audience, where high profile games are already commented by human commentators. For example, there are games important to the teams involved but not commented by human voiced commentators, such as games in lower divisions and youth games. Therefore, to constrain the degree project, the comments are adjusted to the later. Tags are not commented at all, due to typically not being provided live for such games.

There is a possibility of generating subjective comments adjusted to the fans of each team, where an event can be positive or negative depending on the team. To constrain the degree project, subjective comments are not generated. The aim is to generate objective comments for the general audience.

4 Data

The provided data used to fit machine learning models, consist of 33 football games from the Swedish leagues Allsvenskan and Superettan during season 2019. The games were recorded by the Spiideo devices from Stadion in Malmö, Tele2 Arena in Stockholm and Gavlevallen in Gävle. The data are collected by request from the Spiideo API server, which consist of audio recordings, player positions and tags for each game. Specifically, a tag contains information of an event and its occurrence, which is an object implemented by Spiideo.

The games and their number of tags in the provided data are given by Table A.1 in Appendix A. Notice, the games are recorded during the summer and autumn, where the number of tags varies between $20 - 85$.

4.1 Tags

The tags of a game are received as a list of objects when requested from the server, which are provided by Svensk Elitfotboll. The tags are kickoff, end of half, goal, shot, corner, penalty, free kick, yellow card, red card and substitution, where corner and free kick are provided from October 2019. Depending on the main event, each tag includes additional information of the event, such as team, players, etc. if available.

4.2 Audio Recordings

The audio recordings of a game are received as a list of items when requested from the server, where each item corresponds to a segment in the game. The segments are received in AAC-format, but converted to 32-floating-point time series using the LibROSA library [18].

A selection of the audio segments in the game Malmö FF - Helsingborgs IF, 26 November 2019, is given by Figure 4.1.

4.3 Player Positions

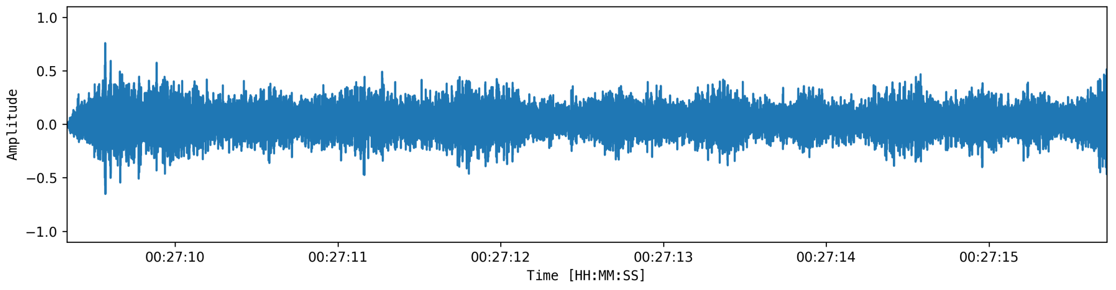
The player positions of a game are received as a list of items when requested from the server, where each item corresponds to a segment in the game. The segments contain information of the foreground ratio and player tracks, which are initially extracted from video data [19].

4.3.1 Foreground Ratio

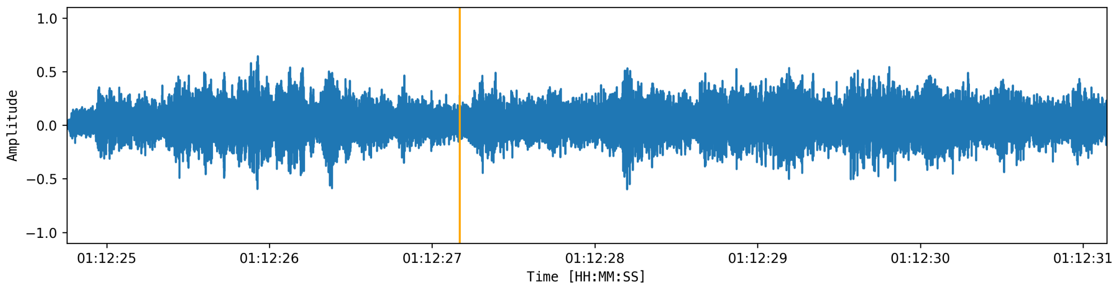
The pixels of a video image are modeled as $P : \Omega \mapsto \{0, 1\}$, indicating the likelihood of a pixel belonging to a moving non-stationary object or the foreground. Assuming the foreground consists mainly of players, the distribution of foreground pixels indicates the positions of the players in the field. The football field is represented by a 17×28 grid, where the modeled pixels are projected into each subregion and estimated as a ratio. In other words, the foreground ratio is represented by an image of one channel, where the ratios in each subregion correspond to the pixels. Each segment consists of a sequence of foreground ratios in chronological time order, where the time difference between the timestamps is 80 milliseconds. [19]

4.3.2 Player Tracks

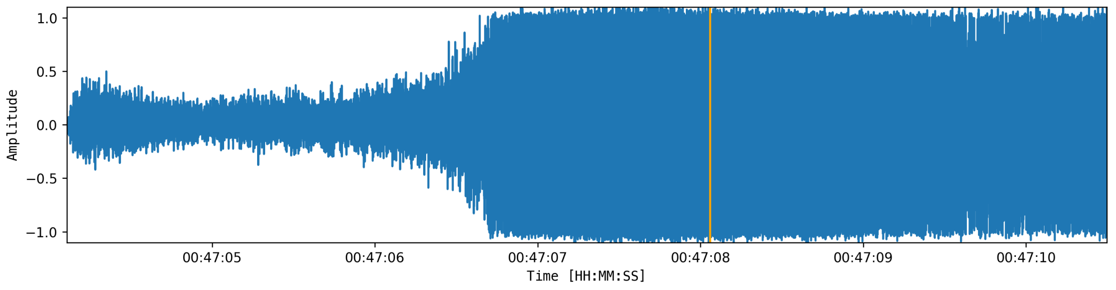
For the player tracks of a segment, each track consists of a sequence of the positions of a player at different timestamps. The player tracks are extracted from the foreground pixels of the video images using a tracking algorithm, where the positions are projected into the field given the field dimensions of the arena. Notice, the foreground pixels belong to moving non-stationary objects, which include the players, referee, ball or anything moving in the video. Therefore, the assumption that a moving object is a player, defines a track as a player track. The track positions are given in chronological time order, where each position is represented by its x and y coordinate in the field and a timestamp. [19]



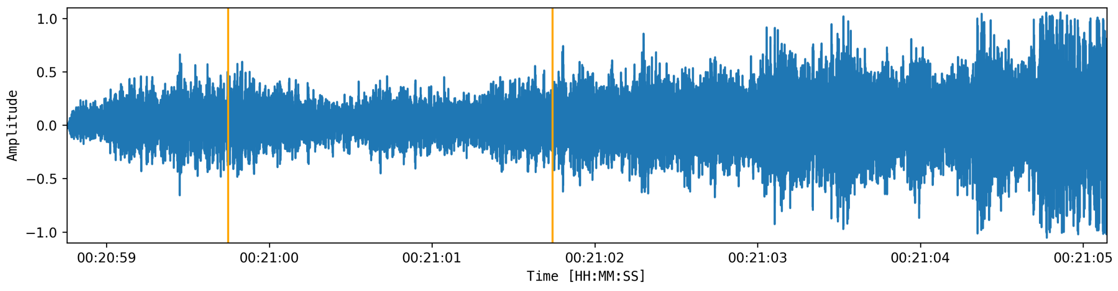
(a) No tag.



(b) Kickoff.



(c) Goal. Notice, the event occurs before the tag.



(d) Shot, shot.

Figure 4.1: Malmö FF - Helsingborgs IF, 26 November 2019. Selection of the audio segments. An orange vertical line indicates the occurrence of a tag.

5 Feature Extractions

To fit machine learning models, some feature extractions of the provided data are required. This concerns both the audio recordings and player positions, where the corresponding feature extraction depends on what the model is intended to learn.

5.1 Spectrogram

Screening through the video recordings of the games in Table A.1, one notices that some events may be detected if able to identify the referee whistle. Examining the spectrogram of the audio segments including tags indicated by a whistle, the conclusion is that referee whistles are distinguishable visually. In particular, a whistle is characterized by two horizontal lines in frequency range $[3000, 5000]$ Hz, where the lengths of the lines indicate its duration. Therefore, the spectrogram is examined as feature extraction of the audio segments for detecting the referee whistles in a game. A selection of the spectrogram feature extraction of the audio segments in the game Malmö FF - Helsingborgs IF, 26 November 2019, is given by Figure 5.1.

5.2 Mel-Frequency Cepstral Coefficients

For automatic speech recognition and modeling the subjective pitch of audio segments, the Mel-Frequency Cepstral Coefficients (MFCC) is proven effective in various environments [5]. Therefore, the MFCC is examined as feature extraction of the audio segments for detecting the events in a game. A selection of the MFCC feature extraction of the audio segments in the game Malmö FF - Helsingborgs IF, 26 November 2019, is given by Figure 5.2.

5.3 Position Feature

To feed the player tracks of the position segments into a machine learning model, some additional feature extraction is required. For detecting the events in a game, both the foreground ratio and motion fields of the position segments are examined as feature extraction. This as a concatenated position feature.

5.3.1 Motion fields

The motion fields of a segment is extracted from the player tracks, which is a feature produced by Spiideo. The momentary velocities of a moving non-stationary object are estimated from the positions and corresponding timestamps of the track, where a velocity is represented by its speed in the x and y direction and a timestamp. The football field is represented by a 17×28 grid, where the speeds in each subregion are approximated given the positions and velocities. In other words, the motion fields is represented by an image of two channels, where speeds in the x and y direction in each subregion correspond to the pixels in the first and second channel respectively. Each segment consists of a sequence of motion fields in chronological time order, where the time difference between the timestamps is 80 milliseconds. [19]

Examining the extracted motion fields of the position segments, one notices that motion fields are not extracted for all timestamps. The absence of motion fields occurs when no tracks including the concerned timestamp are extracted from the foreground pixels. Screening through the video recordings of the games in Table A.1, the conclusion is that the situation may arise when there is close to no movement in the field.

5.3.2 Concatenating feature extractions

The position feature of a segment is extracted by concatenating foreground ratios and motion fields of similar timestamps together, where two timestamps are considered similar if within the distance of 40 milliseconds. As mentioned, motion fields may not be extracted when there is close to no movement in the field, thus the motion fields are padded with zeros for the concerned timestamps. In other words, the position feature is represented by an image of three channels, where the foreground ratio and motion fields in the x and y directions correspond to the channels respectively. Each segment consists of a sequence of position features in chronological time order, where the time difference between the timestamps is 80 milliseconds defined as the timestamps of the foreground ratios. A selection of the position feature extraction of the position segments in the game Malmö FF - Helsingborgs IF, 26 November 2019, is given by Figure 5.3.

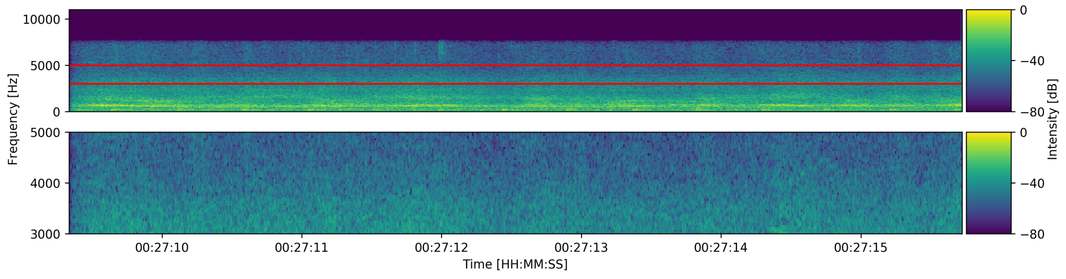
5.4 Parameter settings

When extracting the spectrogram and MFCC features of the audio segments, various parameter settings are available. To constrain the degree project, only one set of parameters is examined. The feature extractions are performed in accordance with Section 2.4 using the LibROSA library, where the set of parameter settings used is given by Table 5.1. [18]

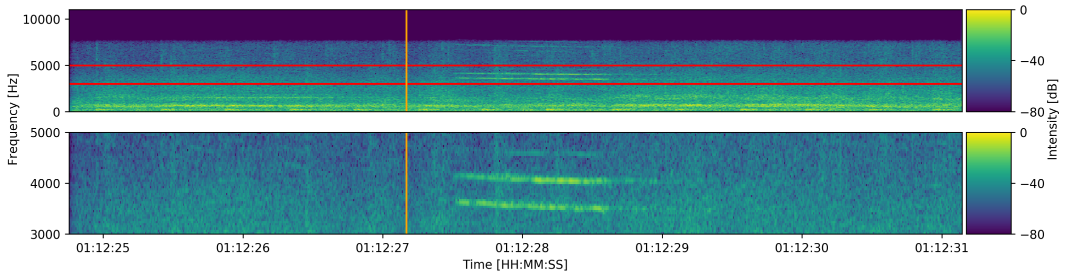
For the spectrogram feature, the default parameters of the LibROSA library are used. The audio segments are received at native sample rate 16000 Hz, but resampled to 22050 Hz when received. The time frames are set to 512 samples corresponding to 23.2 milliseconds, where the overlaps are set to 128 samples corresponding to 5.80 milliseconds. The Hanning window is used as window function, which is recommended for speech processing. For the MFCC feature, the number of filters extracted is set to 26. There are different types of DCTs, where DCT-II is used as the default type of the LibROSA library. Only the first 13 filters are kept as cepstrums. [18][20]

Parameter settings		
Feature extraction	Spectrogram	MFCC
Sample rate	22050 Hz	
Time frames	512 samples 23.2 ms	
Overlaps	128 samples 5.80 ms	
Window function	Hanning	
Filters		26
DCT-type		II
Cepstrums		1-13

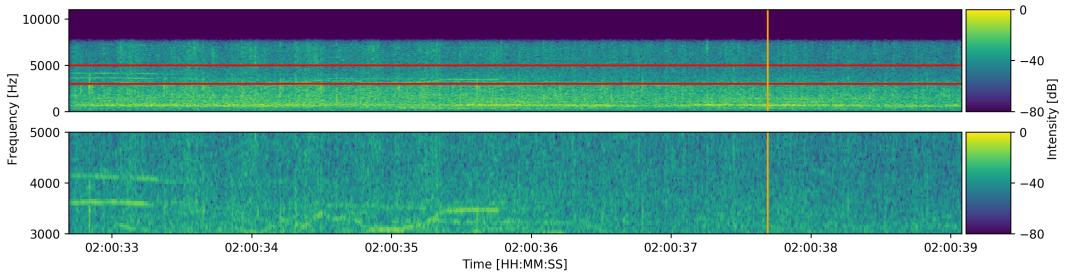
Table 5.1: The set of parameter settings used for the spectrogram and MFCC feature extractions of the audio segments.



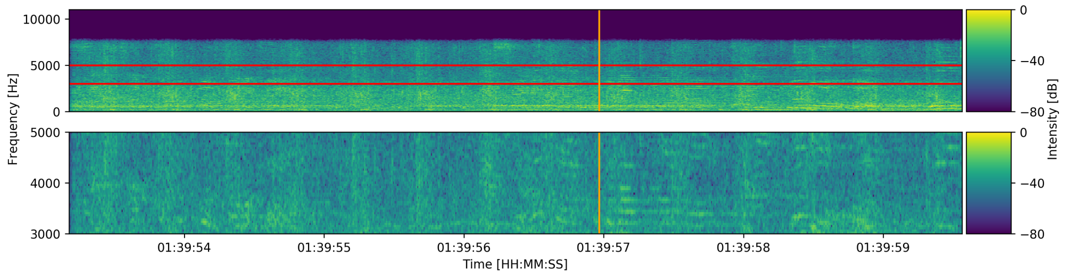
(a) No tag.



(b) Kickoff. Notice, the whistle occurs after the tag.

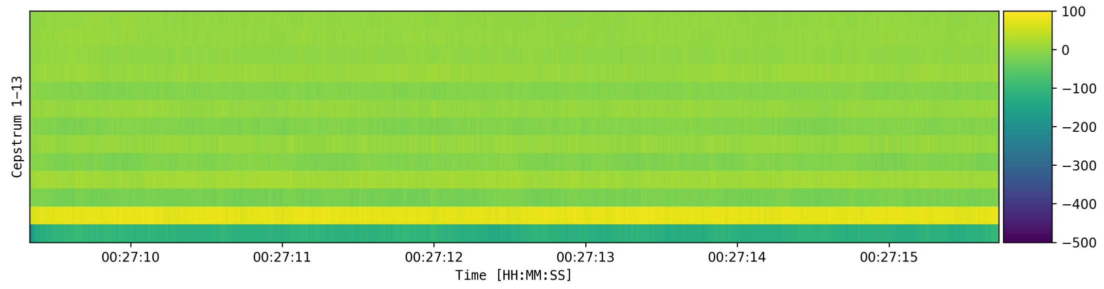


(c) End of half. Notice, the whistle occurs before the tag.

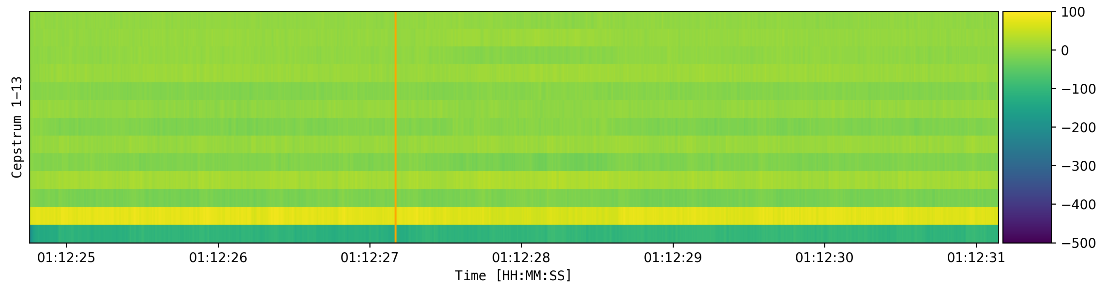


(d) Substitution. Notice, the whistle does not occur in the segment.

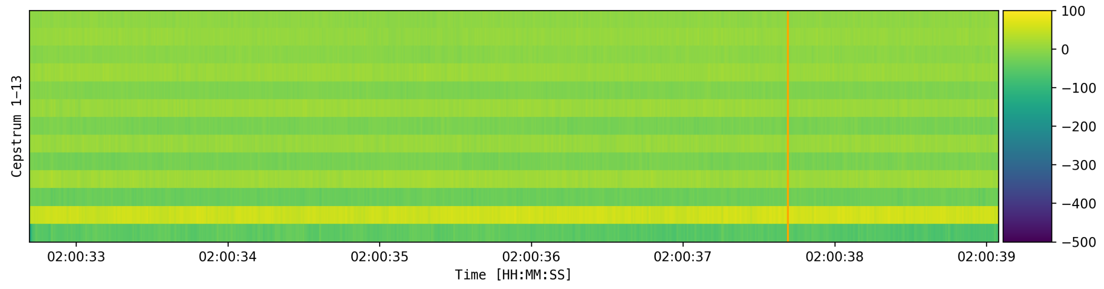
Figure 5.1: Malmö FF - Helsingborgs IF, 26 November 2019. Selection of the spectrogram feature extraction of the audio segments. The upper images visualize the full spectrogram, while the lower are divided into frequency range $[3000, 5000]$ Hz. An orange vertical line indicates the occurrence of a tag.



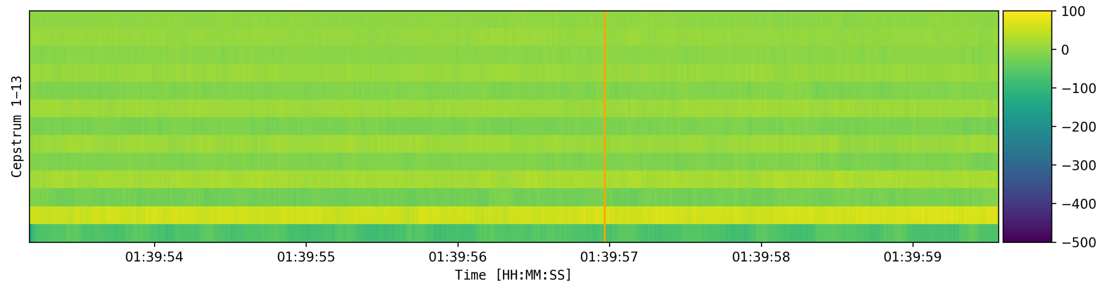
(a) No tag.



(b) Kickoff.

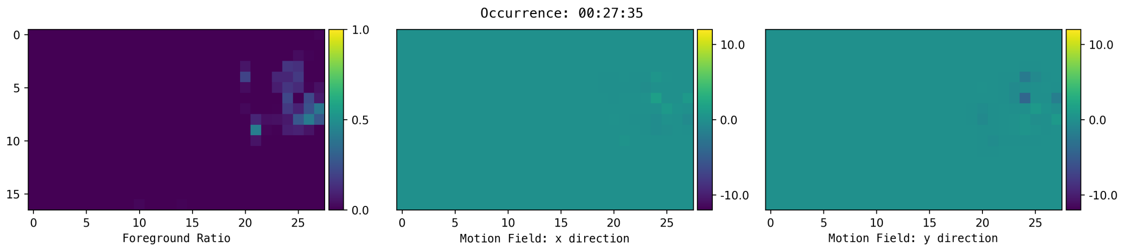


(c) End of half.

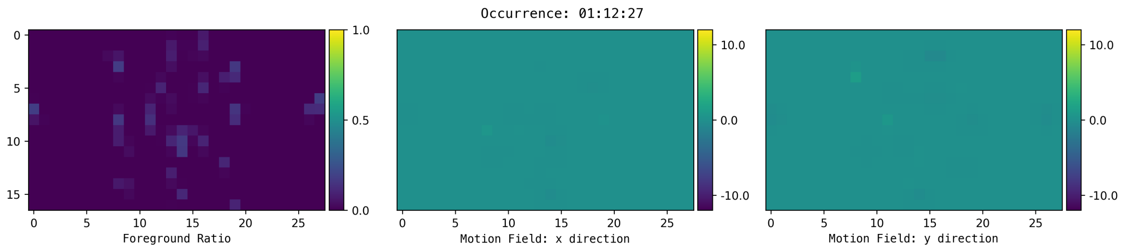


(d) Substitution.

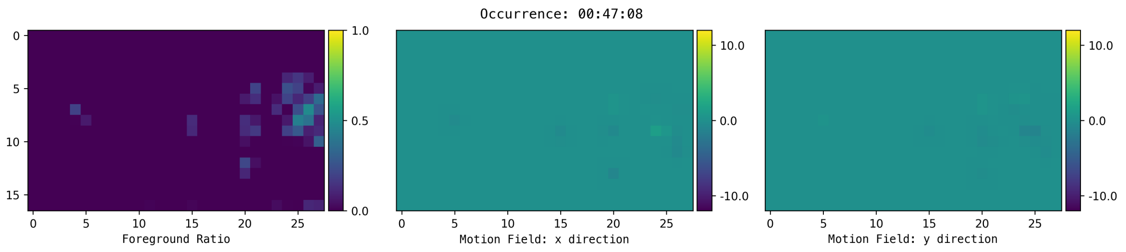
Figure 5.2: Malmö FF - Helsingborgs IF, 26 November 2019. Selection of the MFCC feature extraction of the audio segments. An orange vertical line indicates the occurrence of a tag. Notice, even if the MFCCs may be hard to separate visually, there is a difference between the feature extractions.



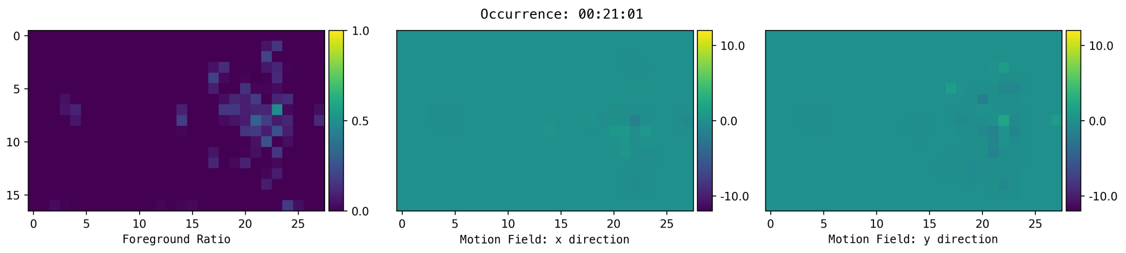
(a) No tag.



(b) Kickoff.



(c) Goal.



(d) Shot.

Figure 5.3: Malmö FF - Helsingborgs IF, 26 November 2019. Selection of the position feature extraction of the position segments. If a segment includes a tag, the frame of closest distance is visualized else the midpoint frame. Notice, even if the motion fields may be hard to separate visually, there is a difference between the feature extractions.

6 Sampling

To feed the data of audio recordings and player positions into a live commentator setup in a convenient way, the data are sampled as if the items are pushed from the Spiideo API server during a live recording of a game. This is performed using a sample generator, which is implemented to construct samples online. Each sample consists of the spectrogram, MFCC and position feature extractions in accordance with Chapter 5.

6.1 Sample

The ability to detect some events using the position feature, has been examined previously in *Cost-Optimized Event Detection in Football Video* by Arpe and Ericsson [6]. The optimal results were achieved, if framing the position feature at 1 frame per second and sampling at 3 frames per sample. Therefore, equal settings for sampling the position feature are examined in this degree project exclusively. However, the duration of the referee whistle in the spectrograms are in general shorter than 3 seconds, thus sampling similar to the position feature is probably too long and may confuse a machine learning model. For example, the longer the sample length, the filters of a CNN endanger to learn patterns irrelevant to actual referee whistles. Therefore, both the spectrogram and MFCC features are sampled with 1 second sample length instead. In addition, the spectrograms are divided into frequency range $[3000, 5000]$ Hz.

To construct such samples, the sampling is performed every second in accordance with Figure 6.1. Each sample is defined by a time interval of 3 seconds, which is divided into three equidistant intervals of 1 second. The center interval consists of the spectrogram and MFCC features, while one frame of the position feature is picked at the midpoint of each equidistant intervals. If no frames are extracted within an equidistant interval, a frame of zeros is padded at the midpoint. Given an audio or position segment and three equidistant intervals, the corresponding feature extraction is performed in accordance with Algorithm 1.

6.2 Sample Generator

The sampling is performed using a sample generator, which is implemented to construct samples online as items arrive from the Spiideo API server. The sample generator is defined by a time interval and the audio and position segments of not yet sampled data. The lower bound of the interval is the timestamp to sample from next and the upper bound the minimum of the end timestamps of the segments. As an item is received, the corresponding segment is added to the sample generator, thereafter the upper bound of the interval is updated accordingly. Furthermore, the sample generator attempts sampling in accordance with Algorithm 2, which constructs samples if sampling is possible, else waits for a new item to arrive.

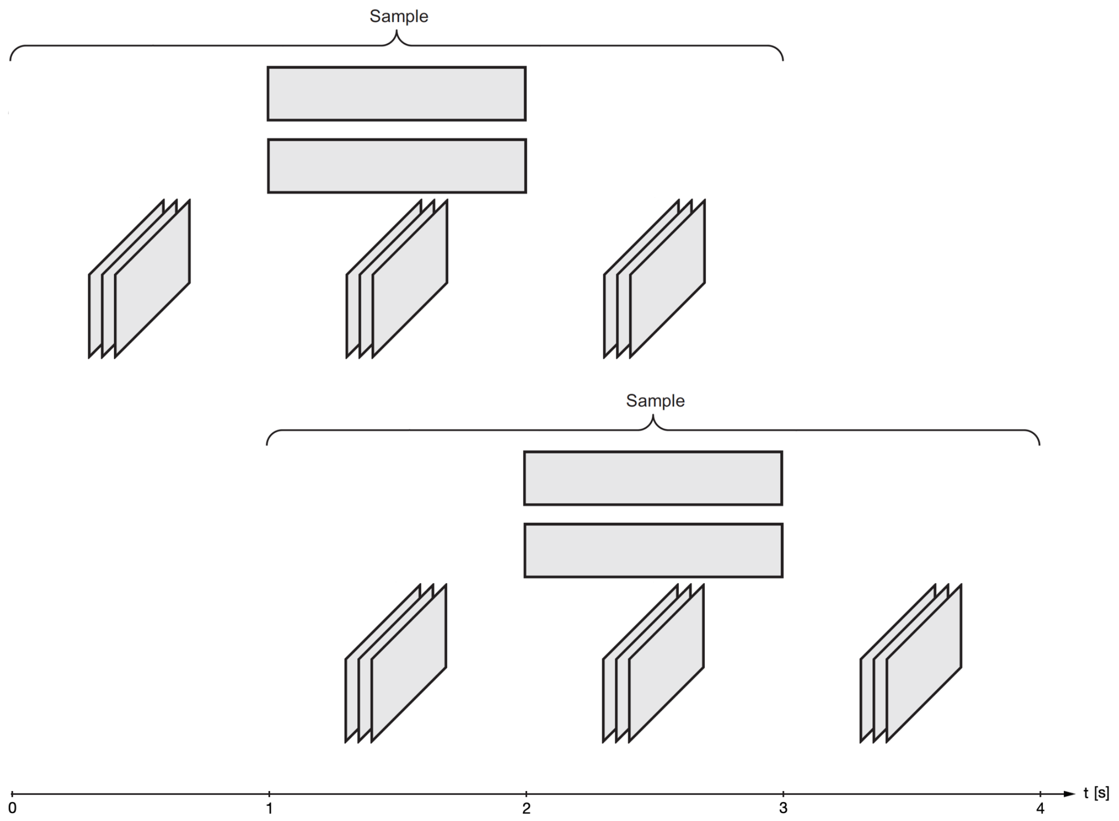


Figure 6.1: Visualization of how the sampling is performed. The sampling is performed every second, where a sample is defined by three equidistant intervals of 1 second. The upper boxes correspond to the spectrogram and MFCC features within the center interval, while the lower frames correspond to the position feature at the midpoint of each equidistant intervals. Each frame contains a foreground ratio and corresponding motion fields.

Algorithm 1: Feature extraction

```
Input: segment, equidistant intervals
if audio then
    for center interval do
        extract spectrogram feature in frequency range $[3000, 5000]$ Hz;
        extract MFCC feature;
    end
else if position then
    extract position feature;
    foreach equidistant interval do
        if no frames then
            pad frame of zeros at midpoint;
        else
            pick frame closest to midpoint;
        end
    end
end
```

Algorithm 2: Sampling

```
Input: sample generator
define empty list of samples;
while lower bound added by 3 seconds is less or equal to upper bound do
    define empty sample;
    define 3 equidistant intervals of 1 second starting from lower bound;
    foreach segment do
        extract features by Algorithm 1 and add to sample;
        remove timestamps less than lower bound from segment;
    end
    add sample to samples;
    add 1 second to lower bound;
end
return samples
```

7 Datasets

Before fitting machine learning models, the data require to be structured into some datasets. Several datasets are constructed for different purposes, depending on what the model being fit to a dataset is intended to learn. One of the feature extractions of the samples is chosen to construct a dataset, thereafter the samples are labeled suitably for the purpose. The samples of a dataset are split into training, validation and test sets, where the samples of each set are balanced, normalized and reshaped if necessary.

7.1 Attack Datasets

As mentioned in Section 5.2 and 5.3, the MFCC feature of the audio segments and the position feature of the position segments are intended to detect the events in a game. Specifically, ongoing attacks are at focus. Therefore, the MFCC and position features of the samples are chosen to construct two separate attack datasets, each containing one of the feature extractions.

Labeling

Considering the events included in the provided tags, one may argue for when an actual event is taking place. For example, is a free kick occurring when a player is fouled, the referee whistles or the free kick is taken? Therefore, one may be more interested in a time interval rather than a specific timestamp of an event, thus the occurrence of a tag is of less importance when labeling. Given this argument and the focus of detecting ongoing attacks in a game, both the MFCC and position feature of a sample are labeled as an event if the event of the tags is goal, shot, corner or penalty and a timestamp of these tags is within the whole interval of the sample, and as no event otherwise.

Splitting into Training, Validation and Test Sets

The labeled samples of the attack datasets are split into training, validation and test sets. Three of the games in Table A.1 are put in the validation set, three in the test set and the rest in the training set. The games are chosen the reflect the variation of the data, thus each of the validation and test sets includes a derby, a game from Allsvenskan and a game from Superettan.

Balancing Classes

Given how the MFCCs and position features are labeled in the attack datasets, the number of samples in each class is heavily imbalanced. To balance the samples in each class evenly, all samples labeled as an event are kept and an equal number labeled as no event. The balancing of classes is performed for each game in each set separately, where the samples labeled as no event are picked randomly. The number of samples and corresponding events in the attack datasets is given by Table 7.1, where the corresponding numbers for each game are given by Table A.2 in Appendix A. For clarification, the initial samples are the same, but the corresponding feature extractions are chosen to construct each of the attack datasets.

Normalizing

The coefficient values of the MFCCs are generally in range $[-500, 100]$, which can be notices in the MFCC feature extractions in Figure 5.2. As mentioned in Section 2.5, CMVN is shown to improve the recognition results and reduce noise. Therefore, the MFCCs of the corresponding attack dataset are normalized using CMVN. This is performed by estimating the cepstral means and standard deviations over the training set, thereafter all sets are normalized in accordance with Equation 2.20.

The values of the foreground ratios are within range $[0, 1]$ and the values of the motion fields are generally in range $[-12, 12]$, which can be noticed in position feature extractions in Figure 5.3. Therefore, as an attempt to balance all values, both the foreground ratio and motion fields of the position features of the corresponding attack dataset are normalized using minmax normalization. This is performed by estimating the minimum and maximum of the foreground ratios and each of the motion fields separately over the training set, thereafter all sets are normalized in accordance with Equation 2.18 and 2.19 respectively.

Reshaping

To feed the samples of a dataset into a machine learning model, the shapes of the samples require to be equal. Given how the MFCCs are sampled in the corresponding attack dataset, the shapes of the MFCCs may vary. Therefore, the MFCCs are reshaped using zero-padding before fed into a model. This is performed by estimating the maximum shape over the training set, thereafter the end of the MFCCs of all sets are padded with zeros or sliced into the maximum shape if necessary.

Given how the position features are extracted, the shapes of the position features in the corresponding attack dataset are equal by construction. Therefore, the position features are not required to be reshaped before fed into a model.

7.2 Whistle Dataset

As mentioned in Section 5.1, the spectrogram of the audio segments is intended to detect the referee whistles in a game. Therefore, the spectrogram feature of the samples is chosen to construct a whistle dataset.

Comparing the timestamps of the provided tags and the occurrences of the referee whistle in the corresponding spectrograms, the conclusion is that the timestamps and whistles do not coincide accurately. This can be notices in the spectrogram feature extractions in Figure 5.1. In addition, the number of tags including a referee whistle is low and does not include all whistles during the games. Therefore, to fit machine learning models for whistle detection, the referee whistles are tagged manually screening the games. This is performed using an analysis tool provided by Spiideo, thereafter the whistle tags are collected similar to the provided data. Due to the time required to tag all the games, only six of the games in Table A.1 are tagged. The games are selected among the training games in Table A.2, which are chosen to reflect the variation of the data, thus two games from each arena are picked. The games and their number of whistle tags in the whistle data are given by Table A.3 in Appendix A. Notice, the number of whistle tags varies between $72 - 82$.

Labeling

The spectrogram of a sample is labeled as an event if a timestamp of the whistle tags is within the center interval of the three equidistant intervals of the sample, and as no event otherwise.

Splitting into Training, Validation and Test Sets

The labeled samples of the whistle dataset are split into approximately 70% training, 15% validation and 15% test sets. This is performed for each label and for each game in Table A.3 separately, thereafter assembled into the corresponding set.

Balancing Classes

Given how the spectrograms are labeled in the whistle dataset, the number of samples in each class is heavily imbalanced. Therefore, the whistle dataset is balanced similar to the attack datasets. The number of samples and corresponding events in the whistle dataset is given by Table 7.2, where the corresponding numbers for each game are given by Table A.4 in Appendix A.

Normalizing

The intensities of the spectrograms are generally in range $[-80, 0]$ dB, which can be noticed in the spectrogram feature extractions in Figure 5.1. Therefore, the spectrograms of the whistle dataset are normalized using minmax normalization. This is performed by estimating the minimum and maximum over the training set, thereafter all sets are normalized in accordance with Equation 2.18.

Reshaping

Given how the spectrograms are sampled in the whistle dataset, the shape of the spectrograms may vary. Therefore, the spectrograms are reshaped using zero-padding before fed into a model, which is performed similar to the MFCCs of the corresponding attack dataset.

7.3 Hard Negative Mining of Whistle Dataset

Since the number of samples in each class is heavily imbalanced before balancing classes, a significant number of whistles from the audience may not be represented in the whistle dataset. Such whistles may be classified as false positives during a live recording of a game, thus the method of hard negative mining mentioned in Section 2.6 is examined. This as an attempt to improve the model for whistle detection. The method is initiated by feeding all samples in the whistle dataset before balancing classes into the optimal initial whistle model, which is declared in the results in Section 11.1.2. Given the model predictions, the samples are classified as true positives, true negatives, false positives or false negatives.

Labeling

The spectrogram of the samples are labeled similar to the whistle dataset, thus a sample classified as true positive or false negative is labeled as an event and true negative or false positive as no event.

Splitting into Training, Validation and Test Sets

The labeled samples of the updated whistle dataset are split into approximately 70% training, 15% validation and 15% test sets, which is performed similar to the whistle dataset.

Balancing Classes

Given how the spectrograms are labeled in the updated whistle dataset, the number of samples in each class is still heavily imbalanced. Therefore, the updated whistle dataset is balanced similar to the whistle dataset, but with some modifications in accordance with the method of hard negative mining. The samples labeled as no event are chosen randomly to include approximately 50% true negatives and 50% false positives in the training and validation sets. This in order for the updated whistle dataset to reflect the variation of the whistle data in a more representative manner, where hopefully some false positives correspond to whistles from the audience. Notice, this is not performed for the test set, due to being able to compare the initial and updated whistle models for improvement. The number of samples and corresponding events in the updated whistle dataset is given by Table 7.2, where the corresponding numbers for each game are given by Table A.4 in Appendix A. Notice, the distribution of events are equal in the whistle dataset and updated whistle dataset by hard negative mining, hence the number are shown together.

Normalizing and Reshaping

The spectrograms of the updated whistle dataset are normalized using minmax normalization and reshaped using zero-padding, which are performed similar to the whistle dataset. For clarification, the minimum and maximum values and maximum shape are estimated over the training set anew for the updated whistle dataset.

7.4 Whistle Event Dataset

Given the optimal whistle model, which is declared in the results in Section 11.1.2, the intention is to determine what event if possible. To fit machine learning models for determining the whistle event, both the events of the whistle detections and actual referee whistles are tagged manually screening the games. This is performed similar to the whistle tags in Section 7.2. The detection event tags are audience, speaker, kickoff, end of half, free kick, penalty, offside, substitution, interruption and others, and the whistle event tags the same as the detection event tags except for audience and speaker. Due to the time required to tag all the games, only three of the games in Table A.1 are tagged. The games are selected as the validation games in Table A.2. The position feature of the samples is chosen to construct the whistle event dataset. The games and their numbers of detection and whistle event tags in the whistle event data are given by Table A.5 in Appendix A. Notice, even if the numbers of tags are close for some events, the numbers are somewhat misleading as sequential detections of the same event are common.

Labeling

As noticed in Table A.5, the number of detection and whistle event tags is low for some classifications. Therefore, some classifications are assembled into a common label. Audience and speaker are commonly labeled as surroundings, free kick and penalty as free kick and end of half, offside, substitution, interruption and others as others. The position feature of a sample is labeled as the common label of a tag if the timestamp of the tag is within the center interval of the three equidistant intervals of the sample. Unlabeled samples are rejected and not included in the whistle event dataset.

Splitting into Training, Validation and Test Sets

The labeled samples of the whistle event dataset are split into approximately 70% training, 15% validation and 15% test sets. This is performed for each label and for each game in Table A.5 separately, thereafter assembled into the corresponding set. The number of samples and corresponding events in the whistle event dataset is given by Table 7.3, where the corresponding numbers for each game are given by Table A.6 in Appendix A. Notice, the number of samples is lower than the total number of tags in Table A.5, due to most whistle detections and actual referee whistles are overlapping.

Normalizing

The position features of the whistle event dataset are normalized using minmax normalization, which is performed similar to the position features of the corresponding attack dataset.

Attack Datasets		
	Samples	Events
Training	4150	2075
Validation	364	182
Test	386	193

Table 7.1: The number of samples and corresponding events in the attack datasets.

Whistle Datasets		
	Samples	Events
Training	642	321
Validation	140	70
Test	146	73

Table 7.2: The number of samples and corresponding events in the whistle dataset and updated whistle dataset by hard negative mining.

Whistle Event Dataset					
	Samples	Kickoff	Free kick	Others	Surround.
Training	256	13	66	126	51
Validation	57	4	15	27	11
Test	61	4	16	28	13

Table 7.3: The number of samples and corresponding events in the whistle event dataset.

8 Detector

After fitting machine learning models to each of the datasets in Chapter 7, a detector is constructed for detecting the events in a game. The detector is defined by the optimal models for attack detection, whistle detection and whistle event determination, which are declared in the results in Section 11.1. Furthermore, additional properties of the data are used, where the audio segments are used to detect high sound powers in a game and the position segments to sort out low movement attack detections.

8.1 Sound Detection

The audio segment of the samples is used to detect high sound powers in a game, which is performed by estimating the power of the segment. Examining the sound powers during the games in Table A.1, the conclusion is that the volume depends on the arena. The volume is low if close to no audience at the arena, such as the games recorded from Gavlevallen in Gävle. The volume is higher for the games recorded from Stadion in Malmö compared to Tele2 Arena in Stockholm, which is probably due to the distance between the field and the recording device. Therefore, the volume threshold is adjusted manually for each arena separately. The sound powers and corresponding adjusted volume thresholds during the validation games in Table A.2 are given by Figure 8.1.

8.2 Attack Detection

Given the results in Section 11.1.1, the position feature performs better compared to the MFCC feature for detecting ongoing attacks in a game. Therefore, the optimal attack model using the position feature is used exclusively for attack detection. On the other hand, the Precision-Recall threshold is useless, due to the corresponding F1 score is undefined. Therefore, the ROC threshold is used as prediction threshold.

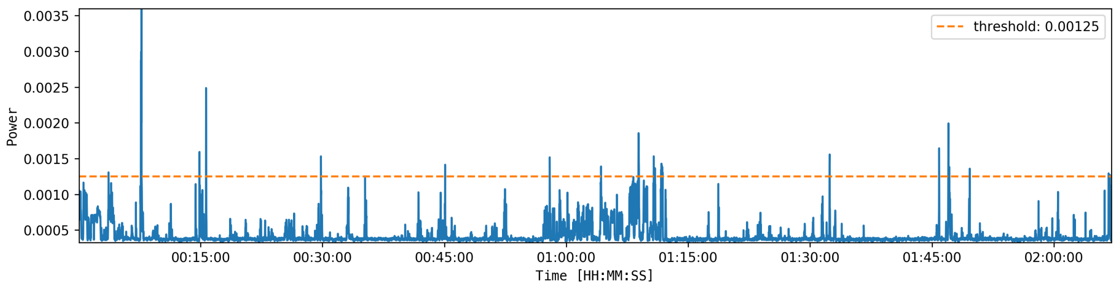
Examining the detections of the games in Table A.1, the conclusion is that the number of detections is very high. The attack model tends to detect multiple sequential detections of the same attack and sometimes when no attack is ongoing. This partly due to the ROC threshold being estimated over the balanced test set, which is naive considering the full imbalanced set during a live recording of a game. However, it is essential that there is certain movement in the field during an ongoing attack, thus the position segment of the samples is used to sort out low movement attack detections. This by summarizing the absolute values of the motion fields into one value, which is performed for each frame of the position feature of a sample, thereafter the mean is estimated over the whole sample and used as an accumulated movement measure. Examining the accumulated movements during the games in Table A.1, the conclusion is that the movements are in the same order of magnitude. Therefore, the movement threshold is adjusted manually for all games. The accumulated movements and adjusted movement threshold during the validation games in Table A.2 are given by Figure 8.2.

8.3 Whistle Detection

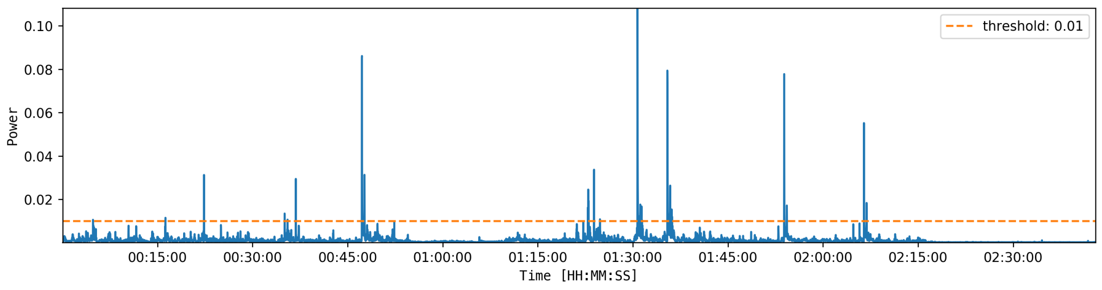
Given the results in Section 11.1.2, the optimal initial whistle model performs better compared to the updated whistle model for detecting the referee whistles in a game. Therefore, the initial whistle model is used exclusively for whistle detection. The Precision-Recall threshold is adapted to imbalanced classification, as during a live recording of a game. Therefore, the Precision-Recall threshold is used as prediction threshold. Notice, the optimal initial whistle model using the Precision-Recall threshold is the model used to construct the whistle event dataset in Section 7.4.

8.4 Whistle Event Determination

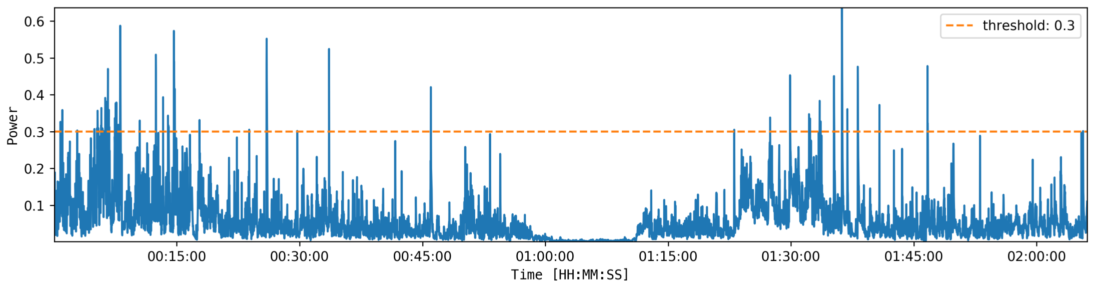
Given the results in Section 11.1.3, the intention is to comment whistle detections determined as kickoff, free kick and others and ignore surroundings. Others are commented even if not able to determine the event, but indicates that a referee whistle has occurred. Surroundings are ignored, due to not able to determine if the event is audience or speaker. Then, the Precision-Recall thresholds perform better compared the ROC thresholds. Therefore, the Precision-Recall thresholds are used as whistle event determination thresholds, where the priority is set in growing order.



(a) Dalkurd - Jönköpings Södra IF, 3 August 2019, at Gavlevallen in Gävle.

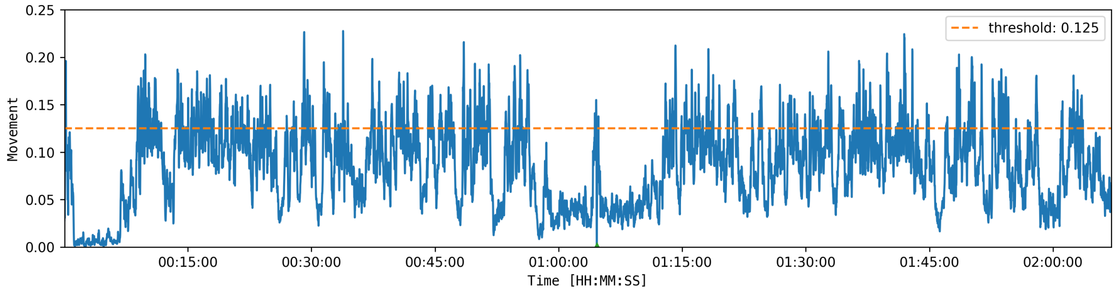


(b) Hammarby IF - Örebro SK, 30 September 2019, at Tele2 Arena in Stockholm.

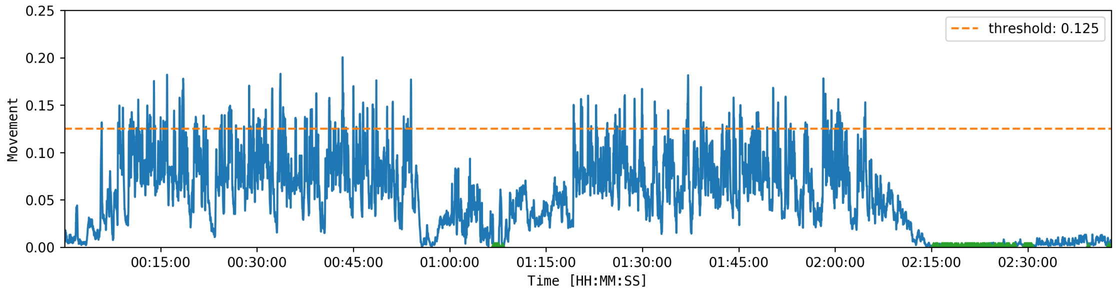


(c) Malmö FF - Djurgårdens IF, 25 August 2019, at Stadion in Malmö.

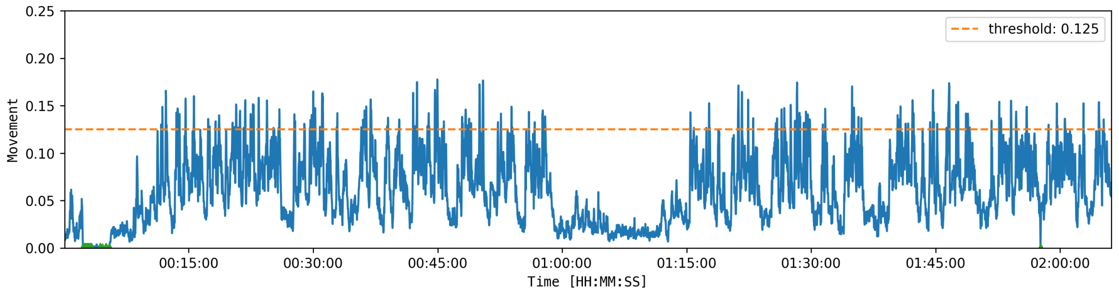
Figure 8.1: The sound powers and corresponding adjusted volume thresholds during the validation games in Table A.2. Notice, the volume depends on the arena.



(a) Dalkurd - Jönköpings Södra IF, 3 August 2019, at Gavlevallen in Gävle.



(b) Hammarby IF - Örebro SK, 30 September 2019, at Tele2 Arena in Stockholm.



(c) Malmö FF - Djurgårdens IF, 25 August 2019, at Stadion in Malmö.

Figure 8.2: The accumulated movements and adjusted movement threshold during the validation games in Table A.2. The green marks at the bottoms indicate when the movement is zero.

9 Commentator

For automatic live commenting of detected events in a game, a commentator is constructed for generating comments using natural language generation. The commentator is defined by its sample generator introduced Chapter 6, detector introduced Chapter 8, commentator state, game information and NLG system. The items and provided tags are fed into the commentator as if pushed from the Spiideo API server during a live recording of a game, thus the commentator is implemented to generate comments online. As items are received, samples are generated by the sample generator, thereafter fed sequentially into the detector. Both the sample and eventual detection are used to update the commentator state. As tags are received, the game information is updated. Furthermore, the detection or tag, commentator state and game information are used as information input into the NLG system. Only valid detections are commented, but all events are used to update the game information. A visualization of the commentator setup is given by Figure 9.1.

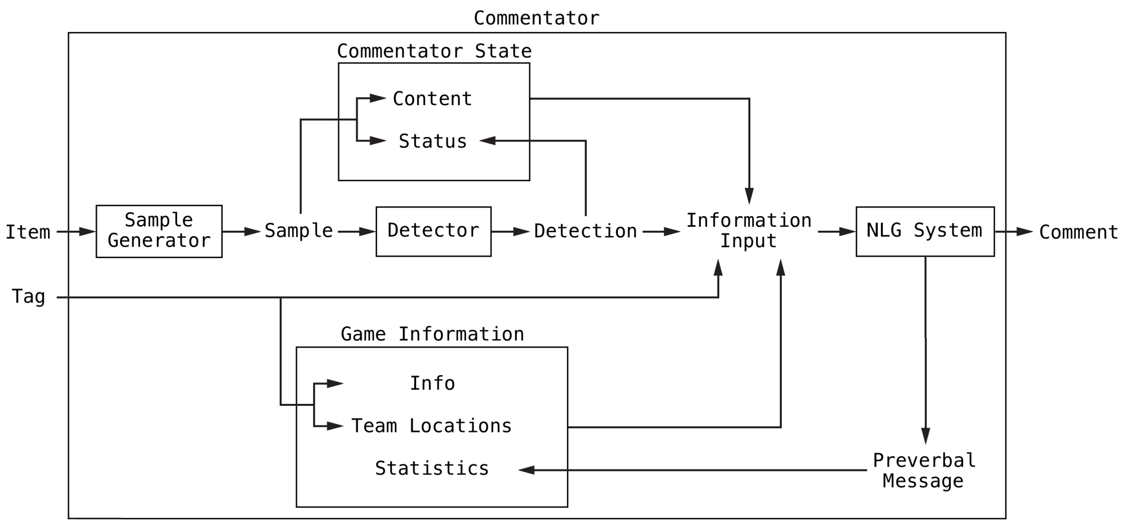


Figure 9.1: Visualization of the commentator setup. Details regarding the commentator state, game information and NLG system are given by Sections 9.1–3.

9.1 Commentator State

The commentator state is used to increase the complexity of the comments, which is defined by its current content and status. The content consists of locations and directions extracted from the position feature of the samples and the status information of ongoing detections. The content and status of the state are updated for every sample generated by the sample generator, where the status is re-updated for every detection by the detector.

9.1.1 Content

To determine valuable locations and corresponding directions for further commenting, the position feature of the samples is used. This under the assumption that the foreground pixels and tracks extracted from the player positions consist mainly of players, thus locations and directions indicate the positions and movements of the players in the field.

First one would like to know where in the field and in what direction the players are moving in general, which may be helpful for deciding which team is attacking during an ongoing attack. The center of gravity of the foreground ratio is used to determine the location coordinates. The averages of the motion fields in the x and y directions are used to determine the direction vector. Second one would like to know where in the field and in what direction the maximum movement occurs, which may be helpful for deciding a player sprint during an ongoing attack. The absolute values of the motion fields are aggregated into an image of one channel, thereafter the maximum value is used to determine the location coordinates. The location in the corresponding motion fields are used to determine the direction vector. The locations and directions of the center of gravity and maximum of movement are extracted for the center frame of each sample only, due to outer frames are similar to the center frame of sequential samples. This given how the sampling is performed, outlined earlier in Chapter 6.

Furthermore, the location coordinates and corresponding direction vectors are converted into string values in accordance with Figure 9.2 and 9.3 respectively. The field is split into two halves defined by the camera angle towards the field, namely left and right. Each half is divided into three zones defined by the attack direction towards the goal, namely left, center and right. In addition, each zone is applied by a set of directions, namely left, center, right and backward for the center zone and forward, center and backward for the left and right zones. The content of the commentator state contains the converted locations and corresponding directions of the center of gravity and maximum of movement, which is updated for every sample generated.

9.1.2 Status

The status of the commentator state contains information of ongoing detections, which consists of one status for sound, attack and whistle separately. Each status may include multiple sequential detections, if detected within a time limit. The time limits are set reasonably by hand given the performance of the detector, which are summarized by Table 9.1. An ongoing detection is interrupted and reset, if the corresponding time limit is exceeded. In addition, an ongoing attack detection is interrupted before the time limit, if the location of the center of gravity switches half. The status of the commentator state is updated for interruption every sample generated by the sample generator and re-updated for ongoing detections every detection by the detector.

Time Limits	
Status	Limit [s]
Sound	60
Attack	30
Whistle	5

Table 9.1: The time limits for ongoing detections.

In detail, each status consists of a set of attributes, which are categorized in accordance with the following list.

`in_progress`
: A boolean. States if a detection is ongoing.

`start_time`
: The timestamp of the first detection included in the ongoing detection. Defines the start time of the corresponding time limit.

`phase`
: A list of phases. Each phase corresponds to the detections included in the ongoing detection. For the sound status, the phases are categorized in accordance with the following list.

  `initial`
  : The first detection of the ongoing detection.

  `ongoing`
  : All sequential detections included in the ongoing detection.

For the attack status, the phases are categorized in accordance with the following list. The phase of a detection is determined sequentially in the given order of the list.

`initial`
: The first valid detection included in the ongoing detection. A detection is valid if the direction of the center of gravity is not backward.

`switch_gravity_zone`
: If the location of the center of gravity switches zone from the previously commented detection validly. A switch is valid if the corresponding direction is not backward and the time distance to the previously commented detection is more than 1.5 seconds.

`switch_gravity_direction`
: If the direction of the center of gravity switches direction from the previously commented detection validly. A switch is valid if the corresponding direction is not backward and the time distance to the previously commented detection is more than 1.5 seconds.

`switch_movement_zone`
: If the location of the maximum of movement switches zone from the previously commented detection validly. A switch is valid if the corresponding direction is not backward and the time distance to the previously commented detection is more than 1.5 seconds.

`switch_movement_direction`
: If the direction of the maximum of movement switches direction from the previously commented detection validly. A switch is valid if the corresponding direction is not backward and the time distance to the previously commented detection is more than 1.5 seconds.

`ongoing`
: If the phase is not determined as any of the previously mentioned phases for the given category.

For the whistle status, the phases are categorized in accordance with the following list. The phase of the detection is determined sequentially in the given order of the list.

`initial`
: The first valid detection included in the ongoing detection. A detection is valid if the whistle event of the detection is kickoff, free kick or others.

`continuation`
: If the whistle event of the detection is kickoff, free kick or others and none of these events are included in the ongoing detection previously.

`ongoing`
: If the phase is not determined as any of the previously mentioned phases for the given category.

`content`
: A list of contents. Each content corresponds to the detections included in the ongoing detection. The content of a detection corresponds to the content of the commentator state when detected.

`timestamp`
: A list of timestamps. Each timestamp corresponds to the detections included in the ongoing detection.

`to_comment`
: A list of booleans. Each boolean corresponds to if to comment the detections included in the ongoing detection. For the sound status, a boolean is true if the phase is `initial`, else false. For the attack and whistle status, a boolean is false if the phase is `ongoing`, else true.

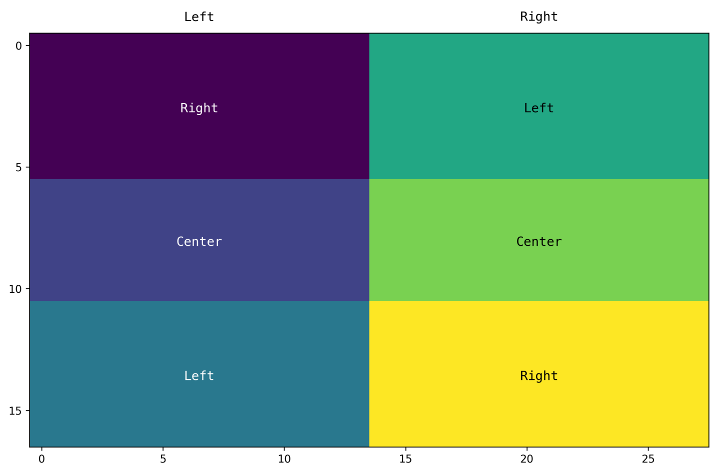
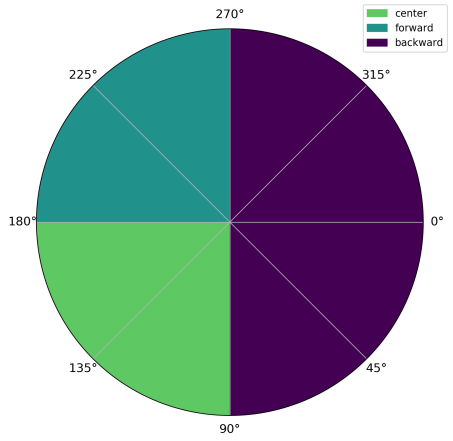
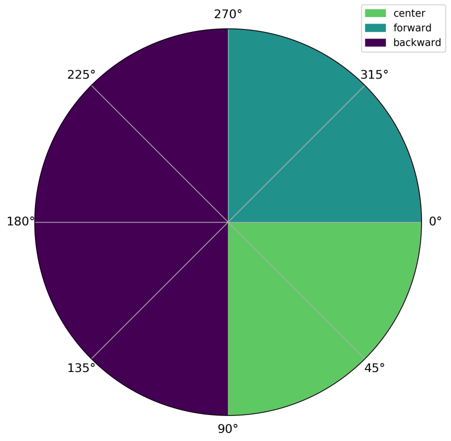


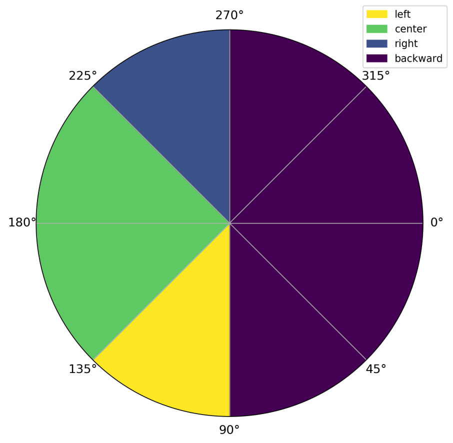
Figure 9.2: Visualization of the conversion of locations, where the coordinates are transformed into two string valued half and zone. The text above the field indicate the halves and the text in the field the corresponding zones. Notice, the axes correspond to the 17×28 grid of the foreground ratio and motion fields of the position feature.



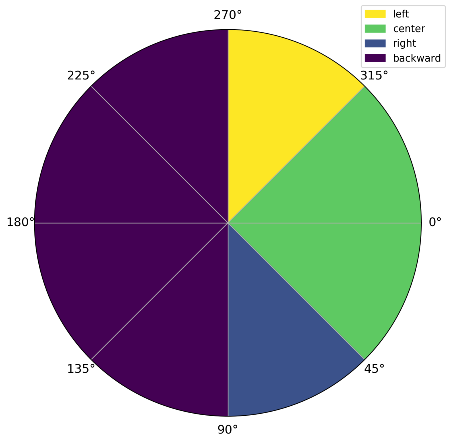
(a) Half: left, zone: right.



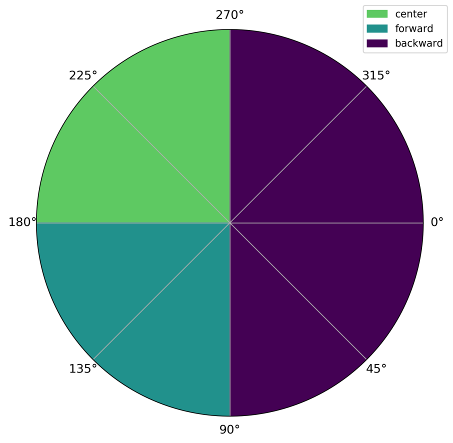
(b) Half: right, zone: left.



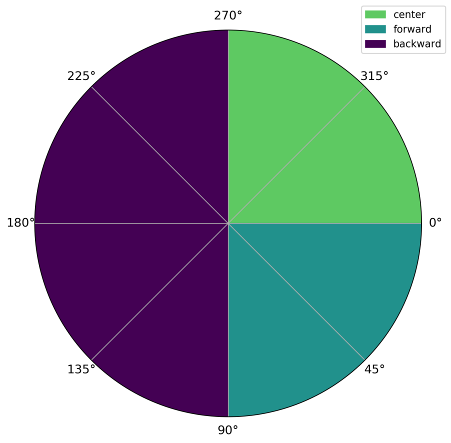
(c) Half: left, zone: center.



(d) Half: right, zone: center.



(e) Half: left, zone: left.



(f) Half: tight, zone: right.

Figure 9.3: Visualization of the conversion of directions for the corresponding halves and zones in Figure 9.2, where the direction vector is transformed into a string valued direction.

9.2 Game Information

The game information is used to increase the complexity of the comments, which is defined by its info, team locations and statistics. The info includes the participating teams, which team playing at home and away, arena and date of play, which are extracted when the commentator is initiated. In addition, the info includes if the game is ongoing and which half, which are extracted from the provided kickoff and end of half tags when received. This to determine the minute an event occurs and the period of the game, where the period is partitioned into before, first half, break, second half and after. The team locations include the location halves of the teams in the field, which are extracted from the provided tags when received and possible. The statistics include information of the detected events, provided tags and players of the provided tags, which are gathered from the NLG system during a game.

9.2.1 Team Location Determination

The provided tags are used to determine the team locations, which include the location halves of the teams in the field. For example, if an ongoing attack is detected, the variation of the comment increases if the attacking team is known. This is performed using the goal, shot, corner and penalty tags, which may contain information of the performing team if provided. In such case, the location of the center of gravity of the commentator state determines the half of occurrence. Then, the performing team is determined as the opposite half and the other team as the given half. The commentator attempts team location determination for every tag received until determination. If the end of half tag of the first half is received, the team locations switches halves.

9.2.2 Game Statistics

Apart from generating comments, the NLG system is used to gather event statistics during a game. The game statistics are partitioned into sound, whistle, left, right and rest categories, where left and right correspond to the team locations of the events. The preverbal messages, constructed by the text planner of the NLG system described in Section 9.3.1, and players of the provided tags, are the information stored in the game statistics. For ongoing sound and attack detections, only preverbal messages constructed from a detection with `initial` phase are stored in the game statistics. For ongoing whistle detections and provided tags, all preverbal messages constructed and players of the tags are stored in the game statistics.

The sound and whistle categories store the sound and whistle detections. The left and right categories store the attack detections, tags and players of the tags, if able to determine the team location of the corresponding preverbal messages and players. Otherwise, the preverbal messages and players are stored in the rest category. The preverbal messages and players stored in the rest category are reconsidered for the left and right categories for every new tag received, if able to determine the team location anew given the new information of the tags. For example, assume the rest category contains the preverbal message of a substitution including the performing team, where a tag is received determining the team location, thereafter the substitution is transferred to the team location of the performing team. Another example, assume the preverbal message of a goal is constructed including the team location and scoring player, where the rest category contains a pair of players from a free kick including the scoring player, thereafter the other player is transferred to the opposite team location. Similar logical rules can be constructed for sorting other preverbal messages and players of the game statistics.

9.3 NLG System

The NLG system is built in accordance with Chapter 3, which generates comments using natural language generation. Only valid detected events are commented, where a detection is valid if the `to_comment` attribute of the corresponding status of the commentator state is true. For clarification, the provided tags are not commented at all, due to typically not being provided live for games in lower divisions and youth games. The NLG system uses a modular pipeline architecture in accordance with Figure 3.1, thus the subproblems are solved in sequential order. In addition, the NLG system is built to provide a welcome and goodbye comment, which are generated when the commentator is initiated and the end of half tag of the second half is received respectively.

9.3.1 Text Plan

Initially, the information input is fed into the text planner, which converts the information into a preverbal message by content determination. The information input consists of the detection or tag, commentator state and game information as visualized in Figure 9.1. As mentioned, the preverbal messages and players of the tags are the information stored in the game statistics, which is the reason the provided tags are fed into the NLG system as well. The preverbal messages constructed for detections, tags, welcome and goodbye are given by Figures B.2–15 in Appendix B.

After content determination, the preverbal messages are filtered through text structuring. For clarification, one text plan is constructed for each detection or tag, which includes one preverbal message. In other words, the main part of text structuring is performed by the detector, which determines the temporal order as events are detected. Therefore, text structuring is built to pass preverbal messages to be commented, where invalid detections and tags are neglected. The result of text structuring corresponds to the text plan.

9.3.2 Sentence Plan

Further, the text plan is fed into the sentence planner, which converts the text plan into a sentence plan. This is performed by sentence aggregation, which separates the information of the text plan into multiple sentences if suitable. For text plans of ongoing attack detections, the information may include the attacking team, general movement and a player sprint. Therefore, to provide more fluid and readable comments, the information is separated into multiple sentences. In addition, the center of gravity or maximum of movement are ignored if the corresponding directions are backward, since the information is irrelevant considering the attack direction towards the goal. The maximum of movement is ignored if equal to the center of gravity, since the player sprint corresponds to the general movement.

After sentence aggregation, the information of the individual sentences are transformed into sets of alternatives by lexicalization. Each set consists of words and phrases to express the information of the individual sentence. The result of lexicalization corresponds to the sentence plan.

9.3.3 Realization

Finally, the sentence plan is fed into the realizer, which generates the output text of the comment. This is performed by linguistic realization, which combines all words and phrases of the sets of alternatives included in the sentence plan. Human handcrafted templates are used, where the words and phrases are expressed in the Swedish language.

After linguistic realization, the output text is generated by deciding a word or phrase randomly among the sets of alternatives of the templates. The result of stochastic decisions correspond to the output text of the comment.

10 Experiment Setup

To fit machine learning models for event detection and generate comments of detected events, an experiment environment is setup. The machine learning models and NLG system are implemented in accordance with Chapter 2 and 3 respectively. Given the performance of the detector, the following NLG system is adapted accordingly.

10.1 Machine Learning

For the machine learning models, the model architectures are implemented using the Keras library with the TensorFlow library as backend. The learning procedures are performed on AWS using a `g4dn.xlarge` instance, which are run on Deep Learning AMI (Ubuntu 18.04) Version 27.0. [21][22][23][24]

In accordance with the purposes of the constructed datasets in Chapter 7, two separate attack models are fit to detect ongoing attacks in a game, where the attack datasets consists of either the MFCC or position feature. Two separate whistle models are fit to detect the referee whistles in a game, where the whistle datasets consists of the spectrogram feature. As a reminder, the updated whistle dataset is constructed by the method of hard negative mining, as an attempt to improve the whistle model. One whistle event model is fit to determine the whistle event, where whistle event dataset consists of the position feature. For clarification, the optimal initial whistle model, which is declared in the results in Section 11.1.2, is the model used to construct the whistle event dataset.

10.1.1 Model Architectures

The model architectures examined are adapted to each dataset individually, which are built as an attempt to capture the patterns of the corresponding feature extraction and separate the classes in a generalized manner. CNNs are investigated for all datasets in accordance with Section 2.1, where the internal structures are experimented with.

The CNN architectures examined for the attack models, whistle models and whistle event model are summarized by Figure 10.1. 1D convolutions are applied if the corresponding dataset consists of the spectrogram or MFCC feature and 3D convolutions if the position feature. Both one and the visualized two fully-connected layers are examined. Different kernel sizes, strides and number of filters are examined for the convolutional layers, different pool sizes and strides for the maxpooling layers and different number of units for the fully-connected layers. For internal convolutional and maxpooling layers, the activation function is chosen as the ReLU in accordance with Equation 2.1. For binary classification problems, the output activation function is chosen as the sigmoid function in accordance with Equation 2.2. For multi-class classification problems, the output activation function is chosen as the softmax function in accordance with Equation 2.3.

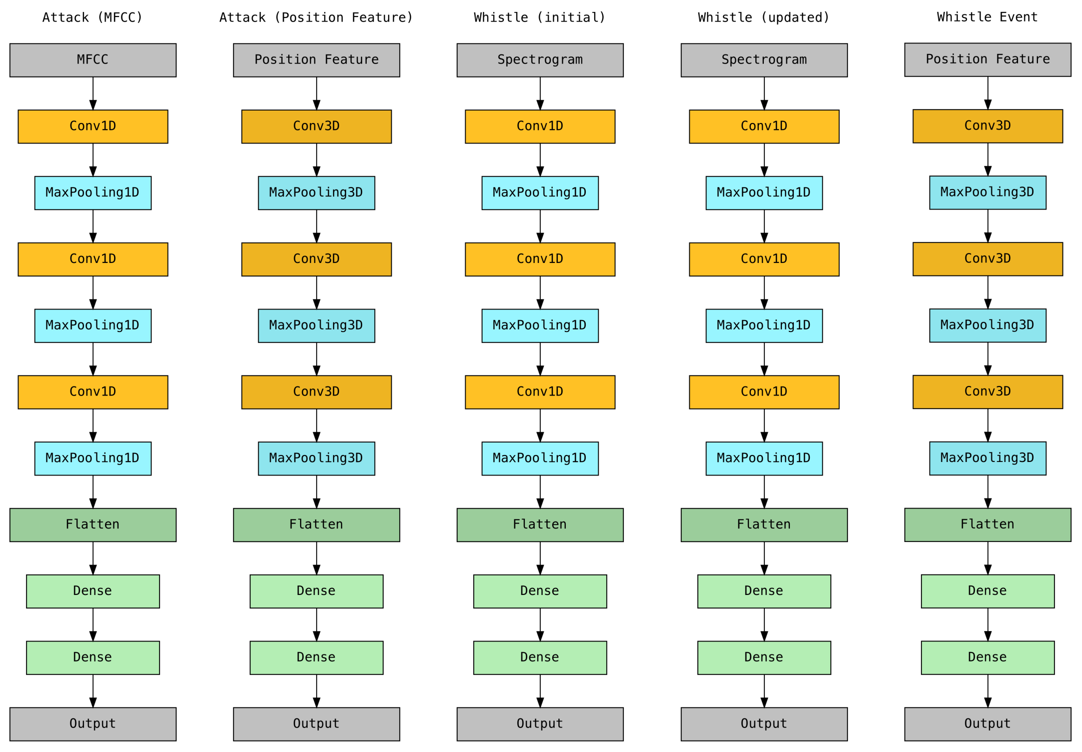


Figure 10.1: Visualization of the CNN architectures examined for the attack models, whistle models and whistle event model.

10.1.2 Learning Procedure

For optimal results during the learning procedures, different batch sizes are examined for each CNN structure. To reduce over-fitting, early stopping is used. A model is fit for 1000 epochs, but stopped after 100 epochs if no improvement in the validation set. The most accurate model on the validation set is evaluated on the test set, where the most optimal model among all fit models is saved and used further on.

The ADAM optimization method is used for all learning procedures. For binary classification problems, the binary cross-entropy is used as loss function and accuracy as metric in accordance with Equation 2.4 and 2.7 respectively. For imbalanced multi-class classification problems, the categorical cross-entropy is used loss function and F1 score as metric in accordance with Equation 2.5 and 2.8 respectively, where the targets are weighted using class weights in accordance with Equation 2.6.

10.1.3 Model Evaluation

In accordance with in Section 2.3, different evaluation methods are used to compare fit models. In addition, the loss and corresponding metric are monitored for all models during the learning procedures.

For binary balanced classification problems, the ROC curve and its optimal threshold are used estimated over the balanced test set. To get a hint on the performance of the models on samples not included in the dataset, the Precision-Recall curve and its optimal threshold are used estimated over the full imbalanced training, validation and test sets. The confusion matrices are used estimated over the balanced test set and full imbalanced training, validation and test sets, where the thresholds are set to 0.5 in both cases.

For the imbalanced multi-class classification problems, the ROC and Precision-Recall curves and their corresponding optimal thresholds for each label are used estimated over the test set. The numbers of true positives, true negatives, false positives and false negatives are calculated as one label versus the rest. The confusion matrices are used estimated over the test set, where the thresholds are set to the optimal thresholds for each label of the ROC and Precision-Recall curves respectively. The threshold priorities are set in growing order, where a sample is classified as undecided if not above any of the thresholds.

10.2 Natural Language Generation

For generating comments of detected events, a commentator is setup in accordance with Chapter 9. The comments of the NLG system are evaluated by running the commentator on new data.

10.2.1 NGL System

The NLG system examined is built in accordance with Section 9.3, which is adapted to the detector in Chapter 8. As a reminder, the detector consists of the optimal models for attack detection, whistle detection and whistle event determination, which are declared in the results in Section 11.1. In addition, the detector is implemented to detect high sound powers in a game.

10.2.2 System Evaluation

The generated comments are evaluated by running the commentator on new data, which consists of games not included in the previously provided data. The new games are recorded from Guldfågeln Arena in Kalmar during season 2020, where the items and tags are collected similar to previous data. The games and number of tags included in the evaluation data are given by Table B.1 in Appendix B. Notice, the games are recorded during the Covid-19 pandemic, thus no audience was allowed at the arena and the majority of the sound environment corresponds to the speaker and yelling of players and team staffs.

Since the NLG system is implemented to generate comments from human handcrafted templates, the variation of the comments is strictly bound the imagination of the implementer. Therefore, the complexity, fluidity and readability of the comments are not evaluated. Instead, the evaluation focuses on how well the generated comments describe the events in a game, where examples of when the comments are correct and incorrect are presented.

11 Results

This chapter presents the results, which are partitioned into two parts. The first part declares the machine learning models for event detection and the second part the generated comments of detected events. The results are produced in accordance with the experiment setup in Chapter 10.

11.1 Machine Learning Models

For the machine learning models, the results are presented in accordance with Section 10.1.3. Only the optimal models for each of the attack models, each of the whistle models and whistle event model are declared.

11.1.1 Attack Models

The performances of the optimal attack models using the MFCC and position feature on the test set are given by Table 11.1. The corresponding model evaluations are given by Figure 11.1 and 11.2 respectively, where the results are summarized by Table 11.2. The model architectures of the corresponding optimal model are given by Figure A.1 in Appendix A.

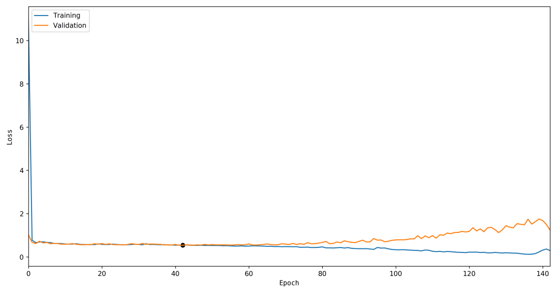
The results indicate that the optimal model using the position feature performs better compared to the MFCC feature. On the other hand, the precision and recall are zero for the optimal threshold using the position feature, thus the corresponding F1 score is undefined. In other words, all samples are classified as no event if using the Precision-Recall threshold for prediction, thus the threshold is useless for application. Therefore, the attack model using the position feature and ROC threshold as prediction threshold, is the optimal model for detecting ongoing attacks in a game.

Performances of Attack Models		
	MFCC	Position feature
Loss	0.5574	0.4872
Accuracy	0.7254	0.8575

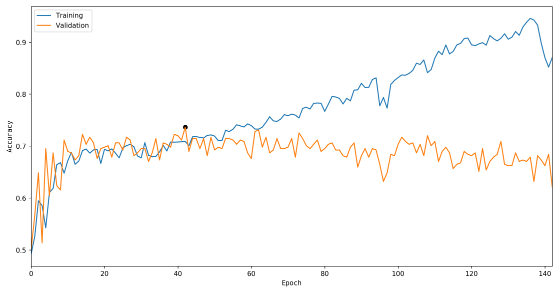
Table 11.1: The performances of the optimal attack models on the test set.

Model Evaluations of Attack Models		
	MFCC	Position feature
Loss	0.5448	0.3223
Accuracy	0.7363	0.9011
ROC		
AUC	0.7807	0.8654
G-mean	0.7239	0.8585
Threshold	0.4985	0.5706
Precision-Recall		
Precision mean	0.02416	0.03076
F1 score	0.1198	Undefined
Threshold	0.8062	0.9988

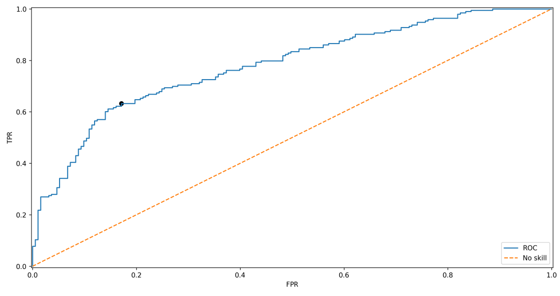
Table 11.2: The model evaluations of the optimal attack models. The numbers correspond to the black dots in Figure 11.1 and 11.2 respectively.



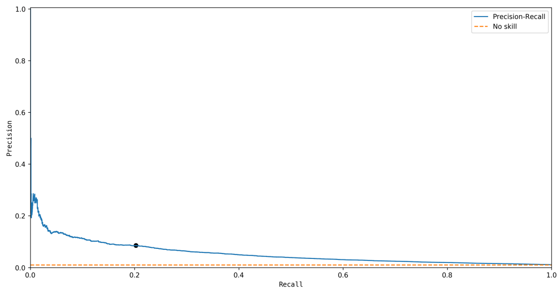
(a) Loss.



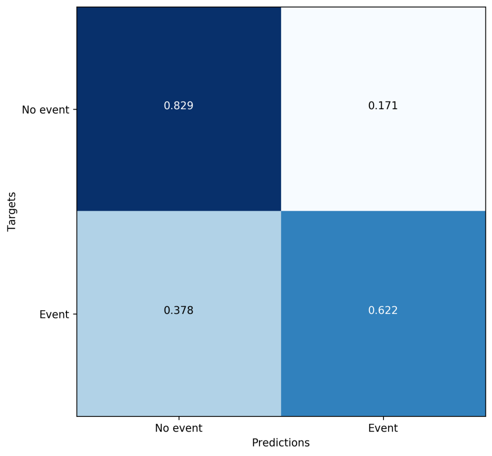
(b) Accuracy.



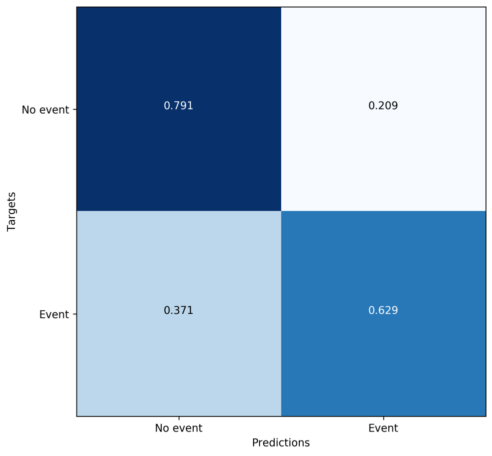
(c) ROC curve.



(d) Precision-Recall curve.

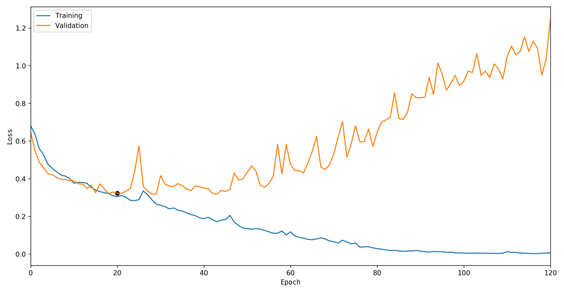


(e) Confusion matrix (ROC).

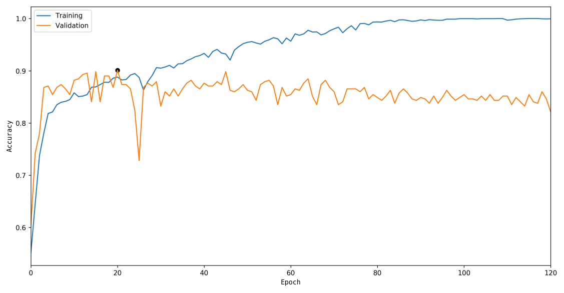


(f) Confusion matrix (Precision-Recall).

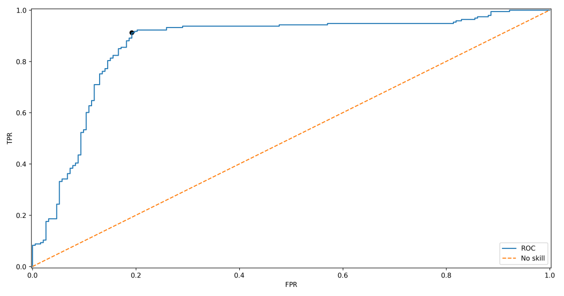
Figure 11.1: The model evaluations of the optimal attack model using the MFCC feature. The loss and accuracy curves visualize the learning procedure, where the black dots correspond to the most accurate model on the validation set. The ROC and Precision-Recall curves are estimated for the most accurate model, where the black dots correspond to the optimal thresholds using the G-mean and F1 score respectively. The confusion matrices are estimated over the samples of the ROC and Precision-Recall curves respectively, where thresholds are set to 0.5.



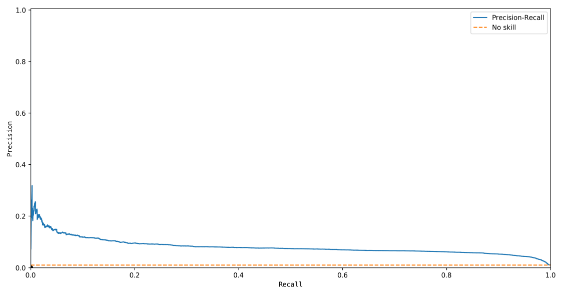
(a) Loss.



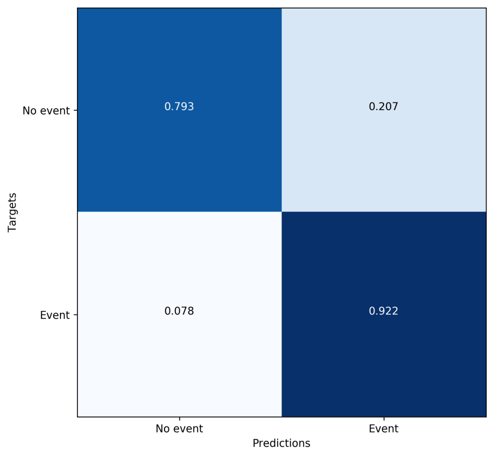
(b) Accuracy.



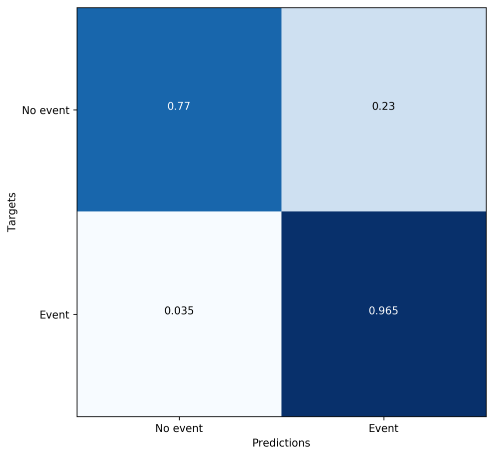
(c) ROC curve.



(d) Precision-Recall curve.



(e) Confusion matrix (ROC).



(f) Confusion matrix (Precision-Recall).

Figure 11.2: The model evaluations of the optimal attack model using the position feature. The loss and accuracy curves visualize the learning procedure, where the black dots correspond to the most accurate model on the validation set. The ROC and Precision-Recall curves are estimated for the most accurate model, where the black dots correspond to the optimal thresholds using the G-mean and F1 score respectively. The confusion matrices are estimated over the samples of the ROC and Precision-Recall curves respectively, where thresholds are set to 0.5.

11.1.2 Whistle Models

The performances of the optimal initial and updated whistle models using the spectrogram feature on the test set are given by Table 11.3. The corresponding model evaluations are given by Figure 11.3 and 11.4 respectively, where the results are summarized by Table 11.4. The model architectures of the corresponding optimal model are given by Figure A.2 in Appendix A.

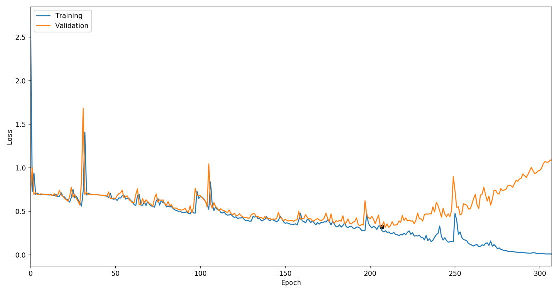
The results indicate that the optimal initial whistle model performs better compared to the updated whistle model by hard negative mining. The Precision-Recall threshold is adapted for imbalanced classification, as during a live recording of a game. Therefore, the initial whistle model using the Precision-Recall threshold as prediction threshold, is the optimal model for detecting the referee whistles in a game.

Performances of Whistle Models		
	Initial	Updated
Loss	0.3065	0.4300
Accuracy	0.8699	0.7877

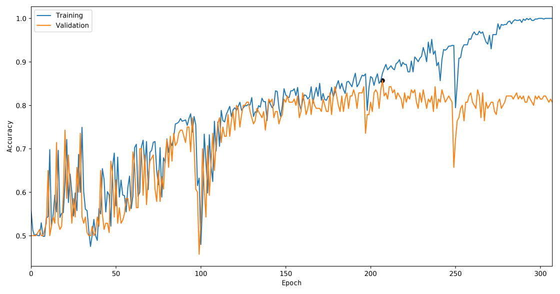
Table 11.3: The performances of the optimal whistle models on the test set.

Model Evaluations of Whistle Models		
	Initial	Updated
Loss	0.3196	0.5362
Accuracy	0.8571	0.7571
ROC		
AUC	0.9394	0.8668
G-mean	0.8766	0.8072
Threshold	0.5374	0.4547
Precision-Recall		
Precision mean	0.04971	0.03770
F1 score	0.4141	0.3782
Threshold	0.9907	0.9225

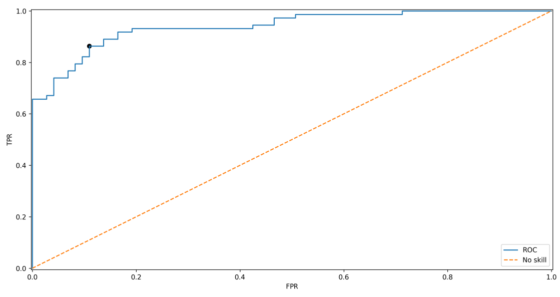
Table 11.4: The model evaluations of the optimal whistle models. The numbers correspond to the black dots in Figure 11.3 and 11.4 respectively.



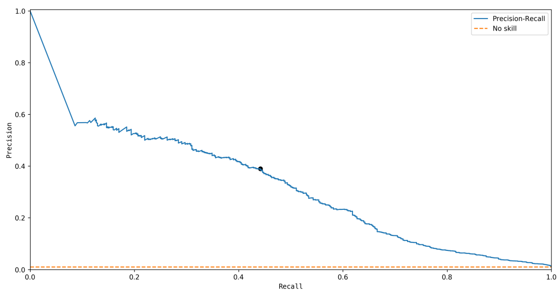
(a) Loss.



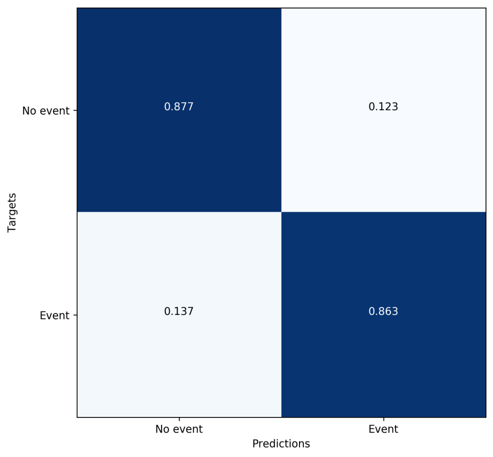
(b) Accuracy.



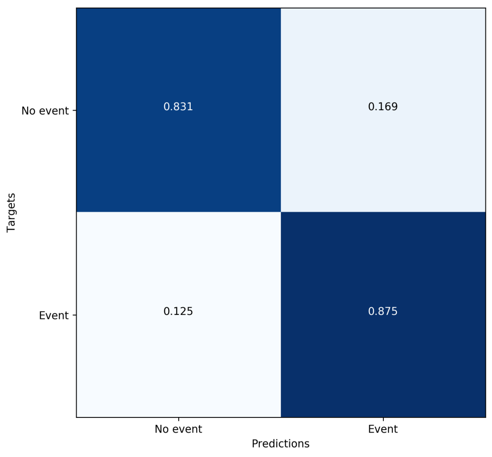
(c) ROC curve.



(d) Precision-Recall curve.

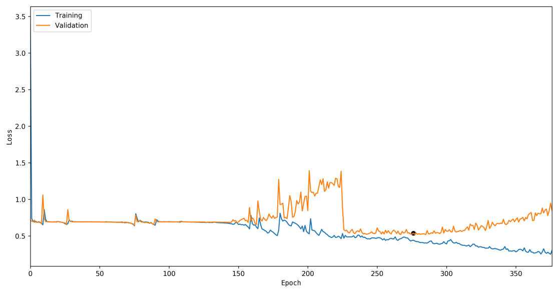


(e) Confusion matrix (ROC).

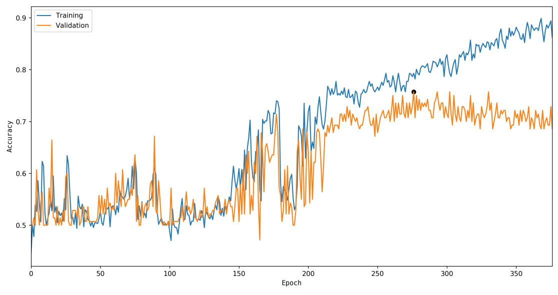


(f) Confusion matrix (Precision-Recall).

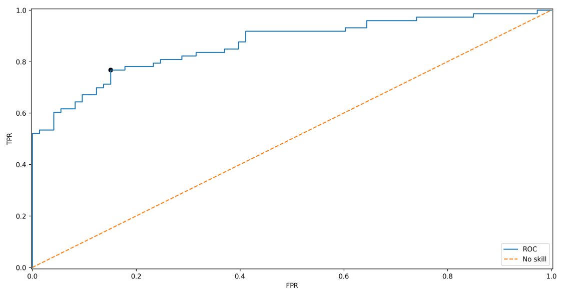
Figure 11.3: The model evaluations of the optimal initial whistle model. The loss and accuracy curves visualize the learning procedure, where the black dots correspond to the most accurate model on the validation set. The ROC and Precision-Recall curves are estimated for the most accurate model, where the black dots correspond to the optimal thresholds using the G-mean and F1 score respectively. The confusion matrices are estimated over the samples of the ROC and Precision-Recall curves respectively, where thresholds are set to 0.5.



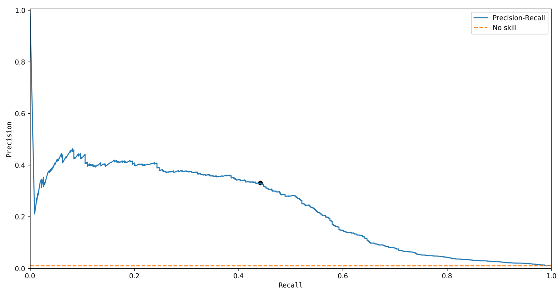
(a) Loss.



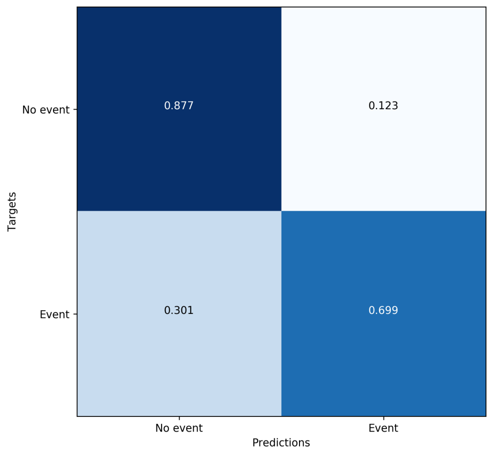
(b) Accuracy.



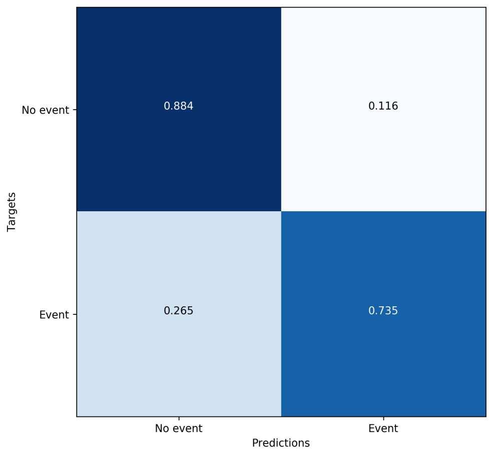
(c) ROC curve.



(d) Precision-Recall curve.



(e) Confusion matrix (ROC).



(f) Confusion matrix (Precision-Recall).

Figure 11.4: The model evaluations of the optimal updated whistle model. The loss and accuracy curves visualize the learning procedure, where the black dots correspond to the most accurate model on the validation set. The ROC and Precision-Recall curves are estimated for the most accurate model, where the black dots correspond to the optimal thresholds using the G-mean and F1 score respectively. The confusion matrices are estimated over the samples of the ROC and Precision-Recall curves respectively, where thresholds are set to 0.5.

11.1.3 Whistle Event Model

The performance of the optimal whistle event model using the position feature is given by Table 11.5. The model evaluation is given by Figure 11.5, where the results are summarized in Table 11.6. For estimating the confusion matrices, the threshold priorities are set in growing order in accordance with Table 11.7. The model architecture of the optimal model is given by Figure A.3 in Appendix A.

The results indicate that the Precision-Recall thresholds performs better compared to the ROC thresholds. This can be notices in the corresponding confusion matrices in Figure 11.5c and 11.5d respectively, where less samples are miss-classified using the Precision-Recall thresholds. Therefore, the optimal whistle event model using the Precision-Recall thresholds, is the optimal model for determining the whistle events in a game.

Performance of Whistle Event Model	
	Position feature
Loss	0.6959
Accuracy	0.8893
F1 score	0.7748

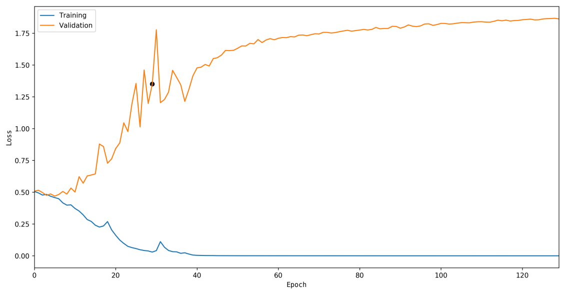
Table 11.5: The performance of the optimal whistle event model on the test set.

Model Evaluation of Whistle Event Model				
	Position feature			
Loss	1.350			
Accuracy	0.8114			
F1 Score	0.6234			
ROC	Kickoff	Free kick	Others	Surround.
AUC	1.000	0.8167	0.8171	0.8317
G-mean	1.000	0.8498	0.8218	0.7783
Threshold	0.1798	0.1930	0.6178	0.09457
Precision-Recall	Kickoff	Free kick	Others	Surround.
Precision mean	1.000	0.5514	0.7095	0.5266
F1 score	1.000	0.8000	0.8254	0.7273
Threshold	0.1798	0.4912	0.6178	0.4652

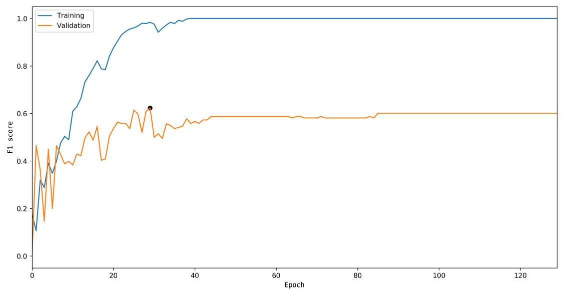
Table 11.6: The model evaluation of the optimal whistle event model. The numbers correspond to the black dots in Figure 11.5.

Thresholds for Whistle Event Determination		
ROC		
Event	Threshold	Priority
Surroundings	0.09457	1
Kickoff	0.1798	2
Free kick	0.1930	3
Others	0.6178	4
Precision-Recall		
Event	Threshold	Priority
Kickoff	0.1798	1
Surroundings	0.4652	2
Free kick	0.4912	3
Others	0.6178	4

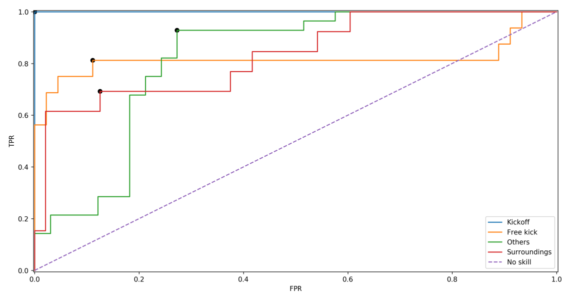
Table 11.7: The thresholds for whistle event determination and their priority.



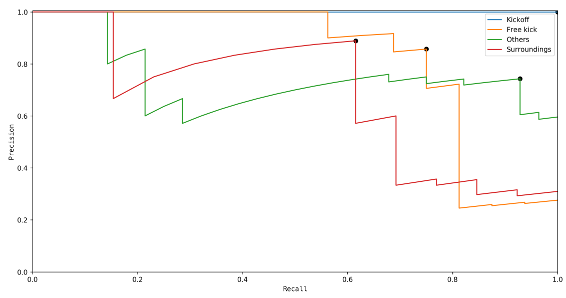
(a) Loss.



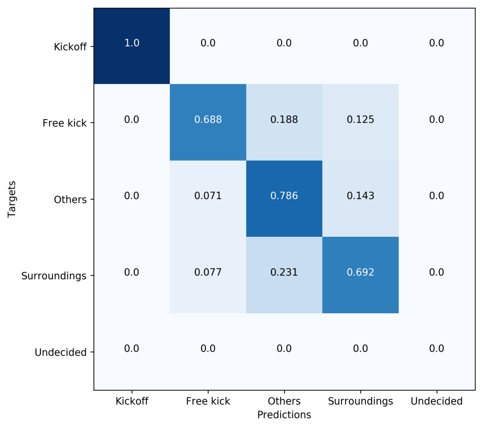
(b) F1 score.



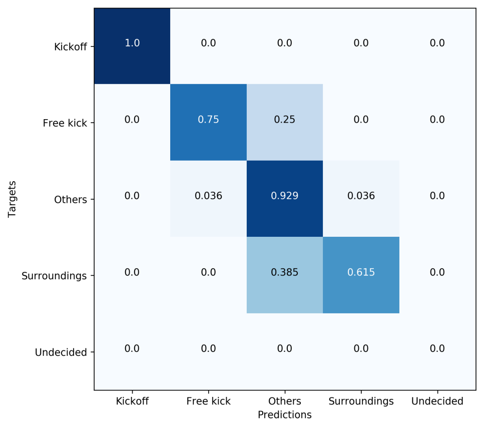
(c) ROC curve.



(d) Precision-Recall curve.



(e) Confusion matrix (ROC).



(f) Confusion matrix (Precision-Recall).

Figure 11.5: The model evaluation of the optimal whistle event model. The loss and F1 score curves visualize the learning procedure, where the black dots correspond to the most accurate model on the validation set. The ROC and Precision-Recall curves are estimated for the most accurate model for each label, where the black dots correspond to the optimal thresholds using the G-mean and F1 score respectively. The confusion matrices are estimated over the samples of the ROC and Precision-Recall curves respectively, where the threshold priorities are set in growing order in accordance with Table 11.7.

11.2 NLG System

For the generated comments of detected events, the results are presented in accordance with Section 10.2.2. Examples of when the the comments are correct and incorrect are given by Figures 11.6a–14, which are presented for the evaluation game Kalmar FF - Örebro SK, 6 August 2020, recorded from Guldfågeln Arena in Kalmar. The sound detection threshold is adjusted using the other evaluation games in Table B.1, which is set to 0.02.

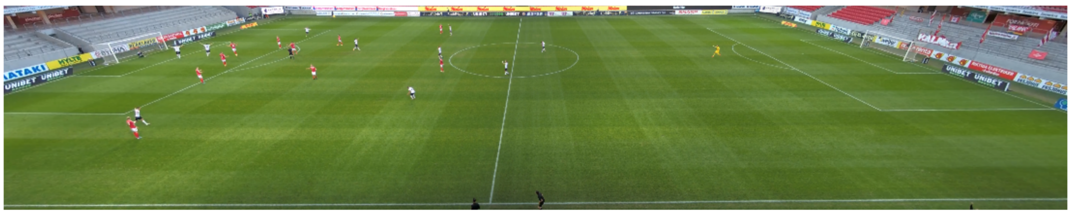
The results indicate that the whistle model performs worse for games recorded from new arenas. The number of detections increases tremendously, where the large majority corresponds to no referee whistle. This affects the whistle event model, which determines a significant number of detections as kickoff, free kick and others incorrectly. This is not the case if running the commentator on the test games in Table A.2, where the whistle model performs similar to the results in Section 11.1.2. However, the performance of the whistle model is improved, if increasing the prediction threshold manually. The number of whistle detections and corresponding generated comments for different thresholds in the evaluation game are given by Table 11.8. The printed comments of the evaluation game are given in Appendix C, where the prediction threshold of the whistle model is adjusted to 0.9999999. The results indicate that the sound detection and attack model perform similar for games recorded from new arenas, while the whistle event model performs worse in a live commentator setup in general.

Performance of Whistle Model		
Threshold	Detections	Comments
0.9907	1187	570
0.999	700	387
0.9999	449	276
0.99999	301	203
0.999999	204	141
0.9999999	145	105
0.99999999	145	105

Table 11.8: The number of whistle detections and corresponding generated comments for the game Kalmar FF - Örebro SK, 6 August 2020, recorded from Guldfågeln Arena in Kalmar. Notice, only valid detections are commented, thus the difference between the numbers.

00:06:37
Publiken hälsar lagen välkomna inför bataljen.

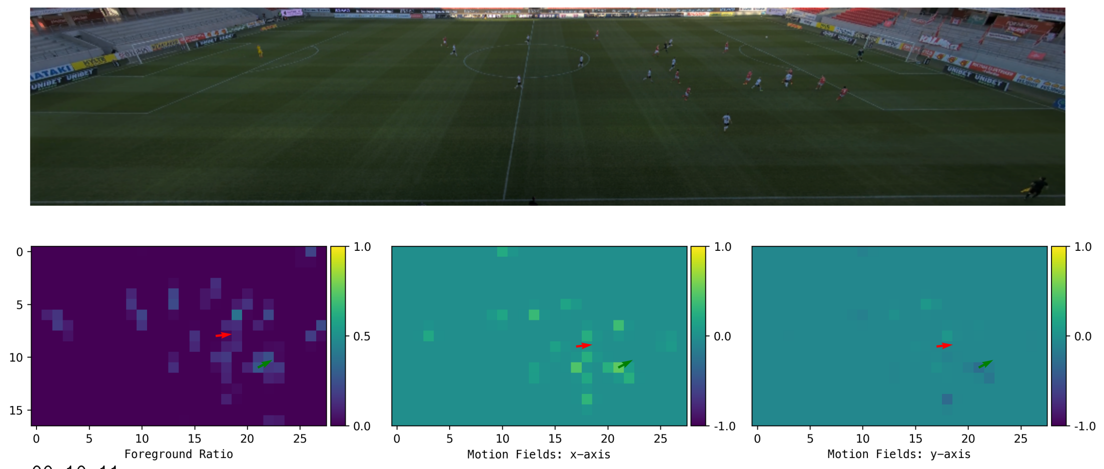
(a) The detection corresponds to the speaker, thus the comment describes the event incorrectly.



01:24:16
Omgivningen gör sig hörd!

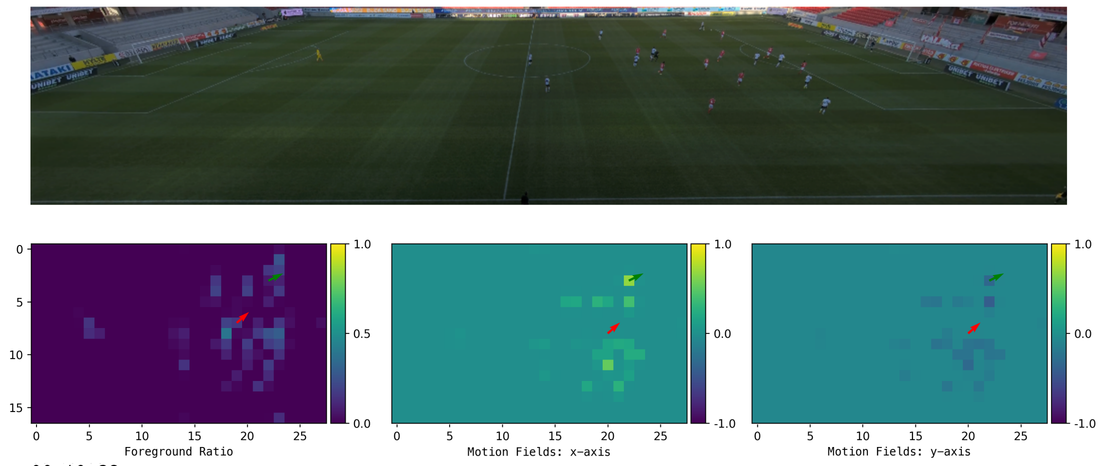
(b) The detection corresponds to a goal scored, thus the comment describes the event correctly. This as the surroundings, i.e. the players and team staff, celebrate the goal.

Figure 11.6: Examples of the generated comments of two separate sound detections, visualized by the corresponding video images. The times correspond to the occurrences into the recording.



```
00:10:11
Ny attack pågår mot mitten från mitten. Samtidigt löpning mot mål längs höger sida.
```

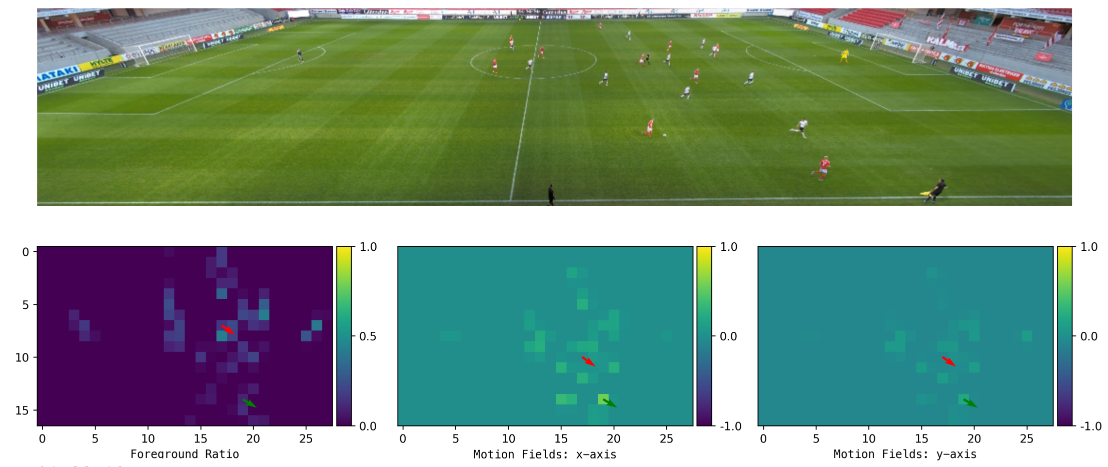
(a) The initial detection corresponds to a long ball from the defence line towards the attacking box, thus the comment describes the attack correctly. However, the player sprint may be irrelevant given the event examining the video sequence.



```
00:10:32
Spelare fyller på framåt på vänsterkanten.
```

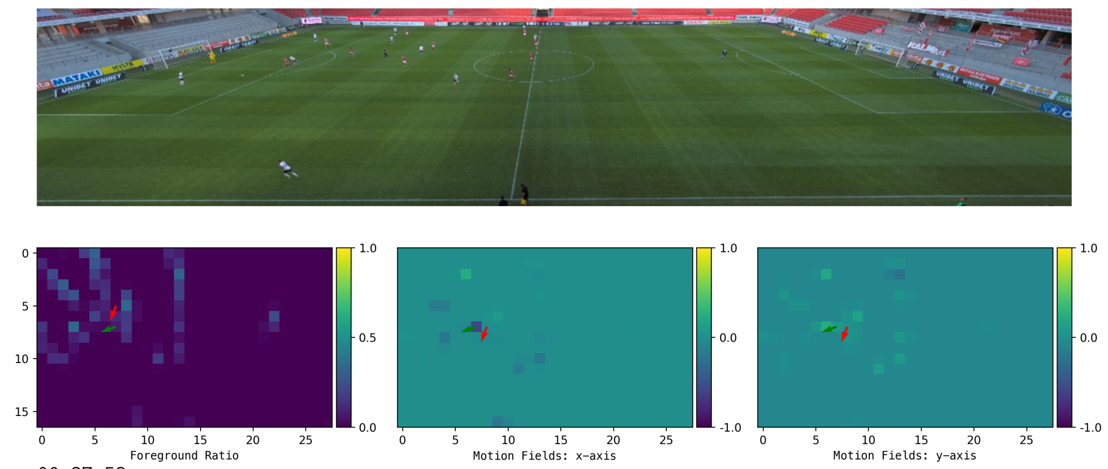
(b) The further detection corresponds to a player sprint on the left wing, which whom the ball is passed to, thus the comment describes the attack correctly.

Figure 11.7: Examples of the generated comments of an ongoing attack detection, visualized by the corresponding video images and center frames of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The times correspond to the occurrences into the recording. Notice, the team locations are not yet determined.



01:23:19
Kalmar FF startar ett nytt anfall mot mål centralt. Löpning kommer framåt på högerkanten.

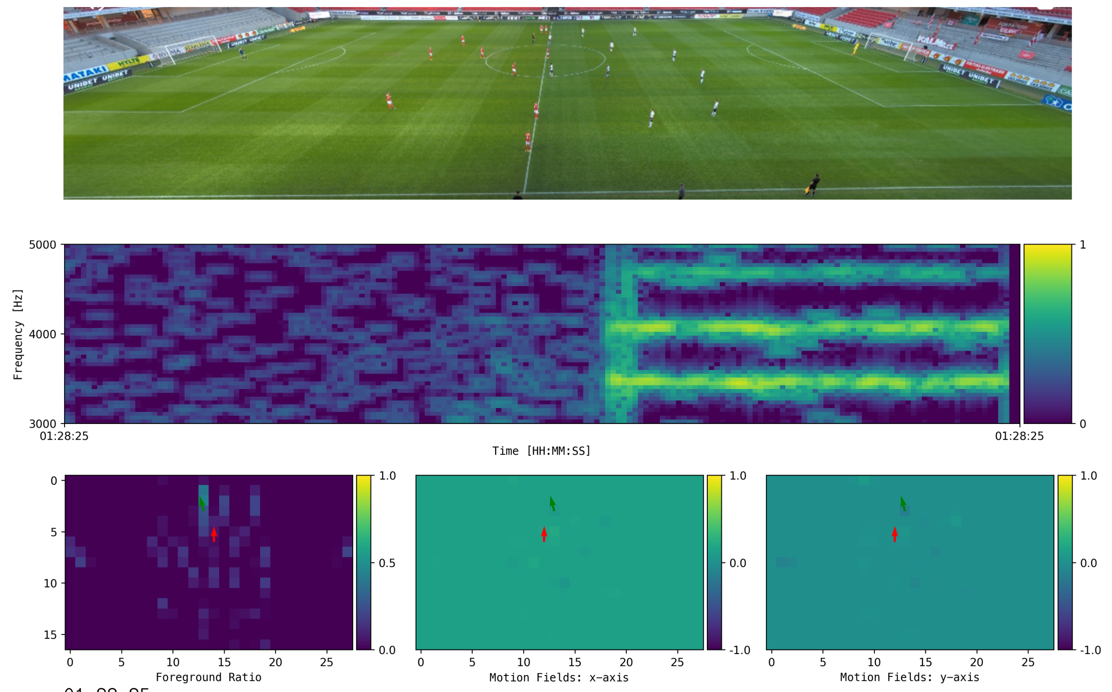
(a) The detection corresponds to the startup of an attack on the right wing, thus the comment describes the attack somewhat correctly. This as the general movement occurs in the center, but the player sprint is accurate.



00:27:58
Hemmalaget inleder ett nytt anfall mot mål från högerkanten. Spelare fyller på mot mitten centralt.

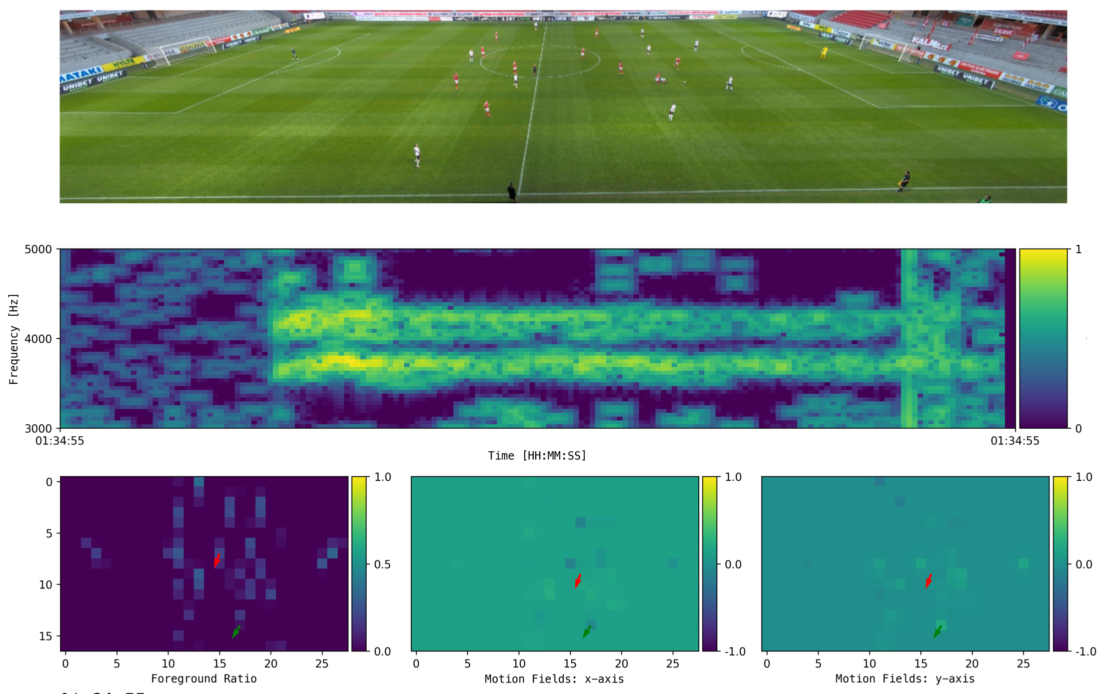
(b) The detection corresponds to Kalmar FF pressuring the defence line of Örebro SK, thus the comment describe the attack incorrectly. In extension, the detection updates the game statistics incorrectly.

Figure 11.8: Examples of the generated comments of two separate ongoing attack detections, visualized by the corresponding video images and center frames of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The times correspond to the occurrences into the recording. Notice, the team locations are determined.



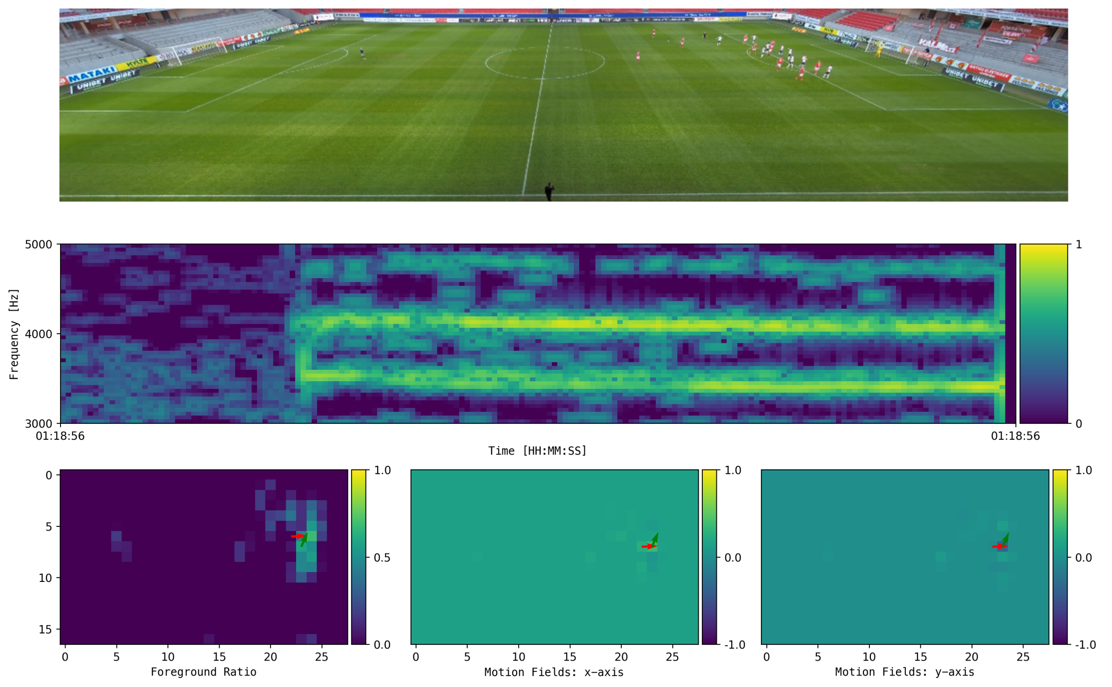
```
01:28:25
Domaren visslar med pipan för avspark. Kampen i spel återigen!
```

Figure 11.9: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the referee whistle of a kickoff after a goal is scored, thus the comment describes the event correctly.



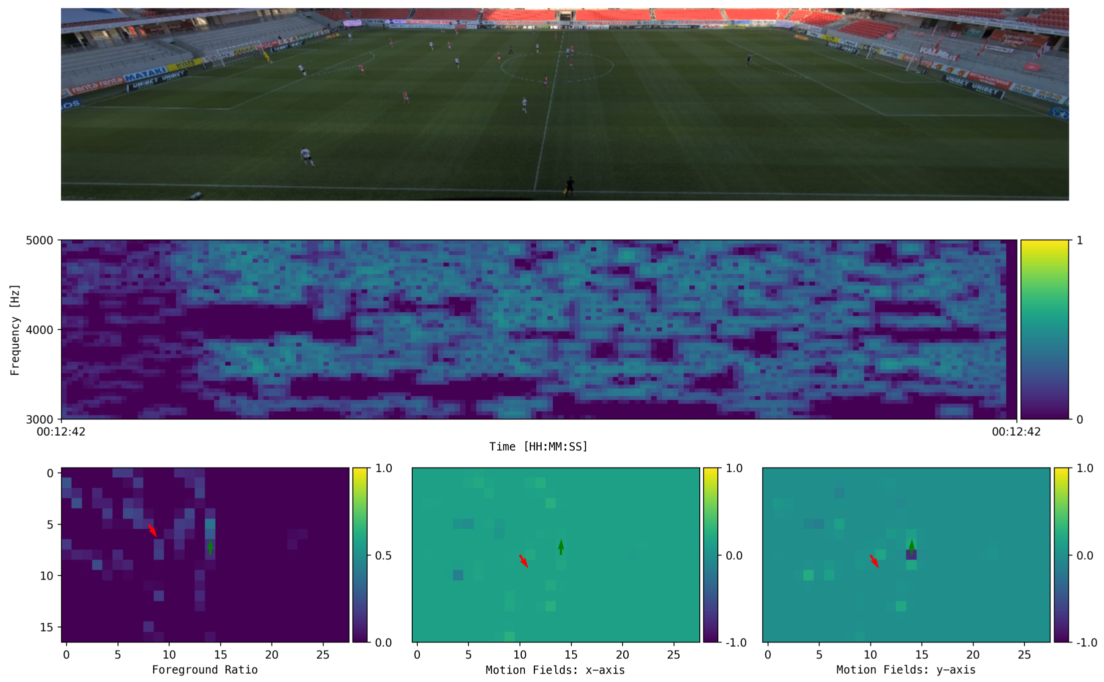
```
01:34:55
Domaren visslar med pipan! En frispark är tillgiven för händelsen.
```

Figure 11.10: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the referee whistle of a free kick, thus the comment describes the event correctly.



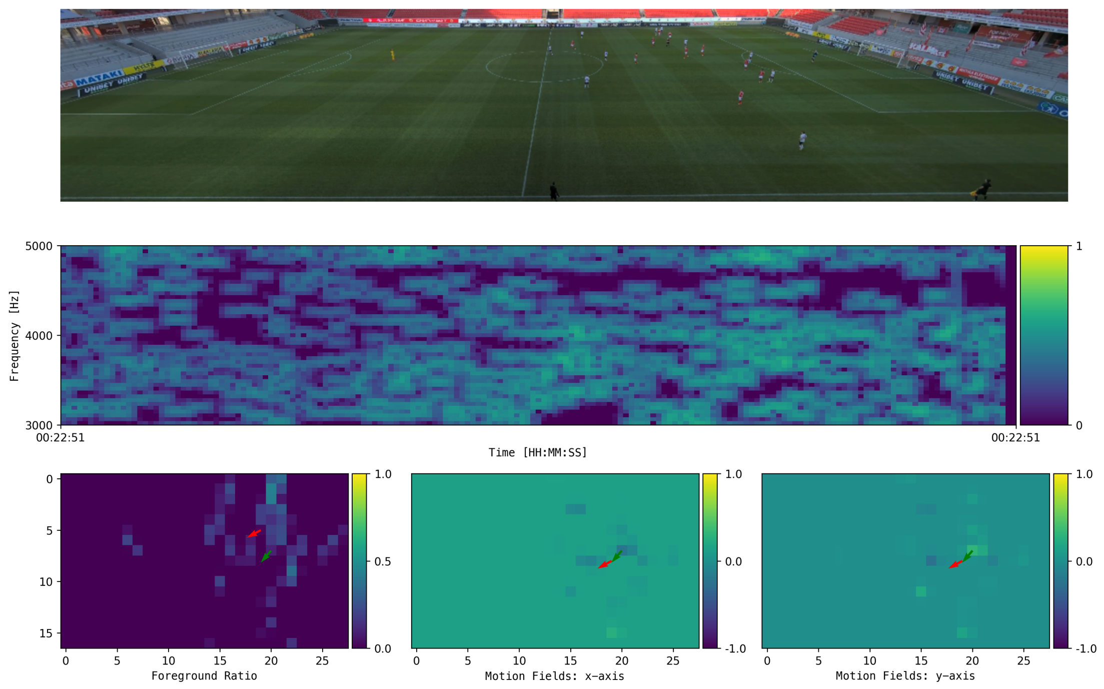
```
01:18:56
Domaren blåser.
```

Figure 11.11: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the referee whistle of allowing a free kick to be taken, thus the comment describes the event correctly.



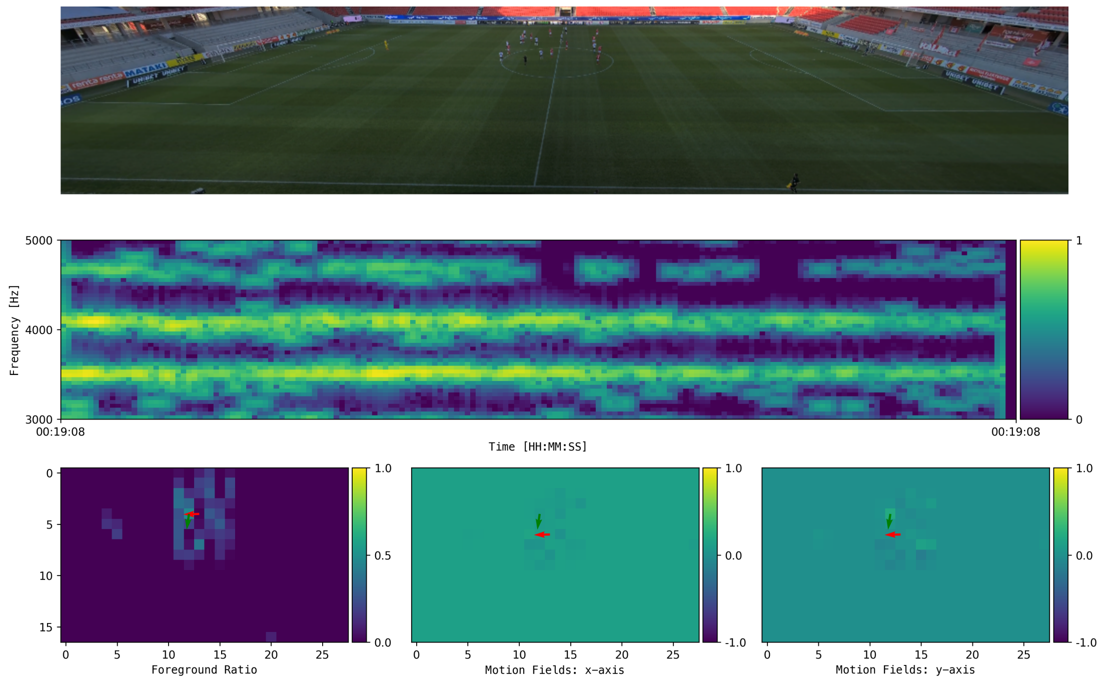
```
00:12:42
Domaren markerar med pipan!
```

Figure 11.12: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the speaker, thus the comment describes the event incorrectly. This can be notices in the spectrogram, where no referee whistle occurs visually. In extension, the detection updates the game statistics incorrectly.



```
00:22:51
Domaren markerar med pipan. Frispark delgiven för företelesen.
```

Figure 11.13: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the players yelling, thus the comment describes the event incorrectly. This can be notices in the spectrogram, where no referee whistle occurs visually. In extension, the detection updates the game statistics incorrectly.



00:19:08
Domaren blåser i pipan för avspark. Kampen är igång igen!

Figure 11.14: Example of the generated comment of a whistle detection, visualized by the corresponding video image, spectrogram and center frame of the position feature. The red arrows correspond to the center of gravity and the green to the maximum of movement. The time corresponds to the occurrence into the recording. The detection corresponds to the referee whistle of allowing a free kick to be taken, thus the comment describes the event incorrectly. In extension, the detection updates the game statistics incorrectly.

12 Discussion

In this chapter, the results of the degree project are discussed. As a reminder, the commentator was evaluated against the evaluation game Kalmar FF - Örebro SK, 6 August 2020, recorded from Gulfågeln Arena in Kalmar. The game are recorded during the Covid-19 pandemic, thus no audience was allowed at the arena and the majority of the sound environment corresponds to the speaker and yelling of players and team staffs. In addition, the commentator was run on the test games in Table A.2 for comparison. This was performed in accordance with the experiment setup and results in Section 10.2 and 11.2 respectively.

12.1 Detector

As mentioned, since the volume depends on the arena, the volume threshold for sound detection was adapted for each arena separately. This probably due to the audience and distance between the field and recording device. The conclusion is that the solution is not optimal, as the threshold requires to be set prior each game commented from new arenas.

Since no audience at the arena, the sound detections of the evaluation game correspond to the speaker and yelling. This may arise for games of low audience in general, where the volume of the speaker and yelling may be higher than the audience. However, the problem may be prevented, if adapting the threshold to sort out the speaker and yelling. On the other hand, increasing the volume threshold may result in no detections at all. To notify the problem, the sound detection threshold of the evaluation arena, was set reasonably to detect some of the speaker and yelling.

Examining the position feature in connection with the provided tags, one noticed that goal, shot, corner and penalty tags are similar visually. Therefore, the conclusion was that these events would be hard to separate individually using machine learning models. However, all events are assumed to occur during an attack, thus the decision was to classify the events collectively using an ongoing attack label.

According to the results in Section 11.1.1, the position feature performs better compared to the MFCC feature for ongoing attack detection. However, the optimal performance of the attack model on the test set of 86% is not ideal, which resulted in a significant number of detections when no attacks were ongoing. The attack model tended to detect the water sprinklers when activated and lineups before offensive free kicks and corners. The problem was partly handled by sorting out low movement detections using the accumulated movement, which resulted in detections corresponding to more accurate events included in an attack. Still, the model sometimes detects the opposing team pressuring the defence line, thus the final model is not perfect. In extension, this affects the game statistics, where the attack count for the pressuring team is increased incorrectly. Nonetheless, the conclusion is that the attack model performs accurately for detecting ongoing attacks in a game.

The performance of the attack models may be improved if fitting machine learning models to more data, which would increase the possibility of separating the events included in the ongoing attack label as well. In addition, different parameter settings for extracting the position feature may be examined to find the optimal parameters, which was not performed due to the computational cost and to constrain the degree project.

According to the results in Section 11.1.2, hard negative mining did not improve the results for whistle detection. This is rather disappointing as the method improved the recognition results in *Cost-Optimized Event Detection in Football Video* [6], even if the method was performed with some modifications in this degree project. The idea was to update the whistle model for a number of iterations until convergence, but since the model did not improve the first iteration no further attempts were performed. However, the worsened updated whistle model may be due to the tagging of referee whistles. In detail, a whistle was tagged when heard screening the recording of the games, thus the timestamp of a tag corresponds to some time during the whistle. In other words, the same whistle may be included in several samples, but only the sample including the timestamp was labeled as an event. Therefore, the updated whistle dataset includes samples of actual referee whistles categorized as false positives, which probably confused the updated whistle model. This can be avoided if picking false positives at a certain distance from the whistle tags, but was discovered when the degree project had proceeded.

Still according to the results, the whistle model performs relatively accurate if applied to games recorded from arenas the model is fit to. If running the commentator on the test games, the performance is similar to the results in Section 11.1.1. On the other, hand according to the results in Section 11.2, the whistle model performs worse for games recorded from new arenas. The conclusion is that the whistle model is not robust for application on new arenas directly. To generate reasonable comments of the whistle events in the evaluation game, the prediction threshold was adjusted manually. However, if increasing the threshold, actual referee whistles are sorted out as well. In other words, the decision is a trade-off between the number of whistle events commented and errors. Screening through the recording of the evaluation game, one noticed that the referee whistles and surroundings sound differently compared to the previous games. The empty arena introduces some echo, which does not occur in the same extent at the previous arenas. This may be a part of the explanation for the worsened performance from new arenas. Nevertheless, the results are somewhat disappointing, due to the environment being close to similar to the surroundings at Gavlevallen.

The performance of the whistle models may be improved if tagging the referee whistles manually for more games recorded from more arenas, which would probably fit a more robust model and improve the worsened performance from new arenas. The hard negative mining method may be retried in accordance with the modification above. In addition, different parameter settings for extracting the spectrogram feature may be examined to find the optimal parameters, which was not performed for the same reasons as the attack model. The samples were constructed to include a spectrogram and MFCC of 1 second and a position feature of 3 seconds, where the decision was partly based to maintain the simplicity of the sample generator. On the other hand, sampling the audio segments at 1 second, the spectrograms may include data irrelevant for a whistle. Therefore, the models may be improved if shorten the sample length, but the tagging requires higher precision when the sample length is shortened.

According to the results in Section 11.1.3, the whistle event model performs relatively accurate. On the other hand, if running the commentator on the evaluation game and test games, the model performs poorly compared to the presented results. A significant number of surroundings are miss-classified as kickoff, free kick and others. Confusion among the classes occurs in greater extent than presented. This results in errors during commenting and affects the game statistics, where the counts are updated incorrectly. However, considering the small and imbalanced whistle event dataset in accordance with Table 7.3, the results are not surprising. Examining the position feature in connection with the whistle event tags, one noticed that only kickoff tags are significantly different from the rest of the tags visually. The others label includes a large variety of patterns, due to including several different types of whistle events. Some of those patterns are similar to the free kick patterns. This may explain the poor performance of the model in a live commentator setup. The conclusion is that the whistle event model performs inaccurately, where difficult to determine the event of the referee whistle.

The performance of the whistle event model may be improved if tagging the events of the referee whistles for more games, thus thereby constructing a larger and possibly more balanced whistle event dataset. This would increase the possibility of separating the events included in the others label as well, such as offside and substitution, but was not performed due to the labour of tagging. In addition, different parameters settings for extracting the position feature may be examined to find the optimal parameters, but was not performed for the same reasons as the attack models.

12.2 Commentator

According to the results in the printed comments in Appendix C and Figure 11.6, the generated sound comments do not describe the events completely as intended for the evaluation game. This as the corresponding human handcrafted templates of the NLG system are adapted to the audience singing and cheering, thus the following comments may not be fully accurate. The problem may be prevented by adjusting the sound detection threshold in accordance with Section 12.1 or reconstruct the templates to adapt for games of low audience. However, the threshold for Guldfågeln Arena my be increased reasonably, once the audience is allowed at the arenas again.

According to the results in the printed comments in Appendix C and Figures 11.7–8, the generated attack comments describe the events accurately considering the performance of the detector. The comments depend on the provided tags, where the complexity of the comments increases if able to determine the team locations. In particular, goal, shot, corner and penalty tags are valuable for determining the team locations, where the earlier the tags are provided the better. The conclusion is that the commentator is sensitive to the live provided tags, where the complexity of the comments depends on the delay which the tags are provided. For games where tags are not provided live at all, the variation of the comments is expected to decrease, due to not being able to comment what team is attacking. Unfortunately, this may be the case for games in lower divisions and youth games in general.

As an alternative to the maximum of movement for determining the player sprint, a threshold may be used for determining if a sprint is occurring or not. Then, multiple player sprints during an ongoing attack may be determined, which would increase the information input into the NLG system and thereby the complexity of the comments. This was not performed due to the additional labour and to constrain the degree project.

According to the results in the printed comments in Appendix C and Figures 11.9–14, a significant number of the generated whistle comments describe the events inaccurately. This due to the reasons discussed earlier in Section 12.1. This is not a problem if the referee whistles are determined as others or surroundings, where the following comments are generated as 'the referee whistles' or ignored. The problem arise if the detections are miss-classified as kickoff or free kick, where the following comments are completely inaccurate. A significant number of detections are miss-classified as kickoff, thus the following comments are inaccurate. The majority of the detections are determined as others, thus the NLG system generates an unsatisfying number of 'the referee whistles' comments. The conclusion is that the generated whistle comments are inaccurate, due to the worsened performance of the whistle model for games recorded from new arenas and poor performance of the whistle event model in a live commentator setup.

In general, the complexity of the comments is rather low, due to the information able to be extracted of the detections and relatively small event domain. Therefore, more advance NLG systems are hard to apply for increasing the complexity of the comments further. However, the complexity may be increased if expanding the sets of alternatives to decide among. On the other hand, considering the method of human handcrafted templates of the NLG system, a significant increase of variation is unlikely by further expansion. Therefore, to constrain the degree project, no further attempts to increase the complexity of the comments were performed. In summary, the NLG system more or less comments that a detected event has occurred without deeper details, which is unsatisfying results overall. Nonetheless, the conclusion is that the complexity of the comments are somewhat high enough, considering the performance of the detector and the method of human handcrafted templates.

As mentioned, the preverbal messages constructed by the NLG system are used to store game statistics in a game. The initial idea was to use the statistics to increase the complexity of the comments. Unfortunately, the statistics were not able to be included in the information input into the NLG system in any greater extent, which is somewhat disappointing.

12.3 Computational Speed

When receiving the audio recordings from the Spiideo API server, the LibROSA library requires that the audio segments in AAC-format are saved before loaded and converted into a 32-floating-point time series. This introduces some delay, which turned out to be negligible compared to the time required for sampling, feature extraction and prediction in general.

According to the results in the printed comments in Appendix C, the time required to run the commentator on the whole evaluation game is 1 hour and 50 minutes. This is somewhat misleading, as the request from the server for this game was modified due to an updated data structure in the Spiideo system. The modification introduced some delay, which most likely can be reduced by Spiideo further on. For the previous games, the run times are approximately 45 minutes. How the run time affects the delay during an actual live recording of a game, has not been examined.

12.4 Future Work

Given the results of the degree project, the conclusion is that the detector is essential for automatic live commenting of football games. The results indicate the difficulty of separating the events using the position feature exclusively. However, if using both the audio and position segments as multiple inputs into a parallel neural network, some events may be separable in greater extent than the presented results. For example, the audience celebrating a home score is in general loud, thus some goals may be separable using the MFCC and position features simultaneously. This was not performed to constrain the degree project, but is a reasonable step for future work.

If the position of the ball is known, it is fair to assume that more events are separable using machine learning models. This possibly using the position feature and position of the ball as multiple inputs into a parallel neural network. For example, it is essential that a goal is scored if the position of the ball is in the net or a kickoff is to be taken if the players are lined up and the position of the ball is at the midpoint of the field. Therefore as future work, tracking of the ball may improve the event detection.

Given the results of the degree project, the conclusion is that the NLG system is strongly dependant on the information input. Unfortunately, the information able to be extracted of the detections is rather low, thus the complexity of the comments is restricted. However, assume the detector is able to detect a goal, thus the complexity of the comment would increase if able to determine the scoring player live. Therefore as future work, the step to improve the NLG system is to keep track of the players for the whole games and possibly apply the name, team or position. Then, the information input would increase and the possibility to apply more advanced NLG systems to generate more alive comments. On the other hand, implementing such advanced tracking algorithms, may come with high computational cost.

As mentioned, the game statistics including the preverbal messages, were not able to be used to increase the complexity of the comments in any greater extent. However, the game statistics may be used to construct summarized text of the events in a game using natural language generation. Such texts would only require post provided tags and possibly ongoing attack detections to get a hint on the performance of the teams during the game. This may be something that Spiideo is interested in to implement in the future.

13 Conclusion

In this degree project, automatic live comments of the detected events in a football games have been generated in text format. Concerning the machine learning task, the conclusion is that it is difficult to detect different events in general. The results indicate that the detections of sound powers and referee whistles are sensitive to the arena, where the detection thresholds require to be set manually prior commenting at new arenas. In addition, it is difficult to determine the event of the referee whistle in a live commentator setup. However ongoing attack detections are detected accurately, if sorting out low movement detections.

Concerning the natural language generation task, the conclusion is that it is possible to generate descriptive comments of the events, if regarding the performance of the detector. The results indicate that the complexity of the comments is sensitive to the detector and information able to be extracted of the events, where the variation is bound to the relatively small event domain.

A Appendix A

This appendix includes further details regarding the machine learning task of the degree project. The games and their number of tags in the corresponding data are given by Table A.1, A.3 and A.5. The number of samples and corresponding events for each game in the corresponding datasets are given by Table A.2, A.4 and A.6. The model architectures of the optimal models for each of the attack models, each of the whistle models and whistle event model are given by Figures A.1–2.

Provided Data		
Game	Tags	Arena
190609 Dalkurd - Halmstad	36	Gävlevallen
190625 Dalkurd - Västerås SK	43	Gävlevallen
190706 Malmö FF - Örebro SK	35	Stadion
190707 Hammarby - Falkenbergs FF	43	Tele2 Arena
190714 Djurgården - Malmö FF	36	Tele2 Arena
190720 Dalkurd - Varbergs BoIS FC	42	Gävlevallen
190721 Malmö FF - IK Sirius FK	42	Stadion
190722 Hammarby - IF Elfsborg	40	Tele2 Arena
190728 Djurgården - BK Häcken	35	Tele2 Arena
190803 Dalkurd - Jönköpings Södra IF	34	Gävlevallen
190810 Djurgården - IK Sirius FK	40	Tele2 Arena
190811 Hammarby - Helsingborgs IF	27	Tele2 Arena
190817 Hammarby - GIF Sundsvall	38	Tele2 Arena
190818 Dalkurd - GAIS	34	Gävlevallen
190818 Malmö FF - Falkenbergs FF	32	Stadion
190825 Malmö FF - Djurgården	33	Stadion
190901 Dalkurd - Norrby IF	38	Gävlevallen
190915 Hammarby - IFK Göteborg	32	Tele2 Arena
190915 Malmö FF - IFK Norrköping	36	Stadion
190916 Djurgården - Helsingborgs IF	37	Tele2 Arena
190921 Dalkurd - Östers IF	40	Gävlevallen
190922 Hammarby - AIK	20	Tele2 Arena
190924 Djurgården - Falkenbergs FF	35	Tele2 Arena
190926 Malmö FF - Helsingborgs IF	38	Stadion
190929 Dalkurd - Trelleborgs FF	40	Gävlevallen
190930 Hammarby IF - Örebro SK	36	Tele2 Arena
191006 Djurgården - Hammarby	71	Tele2 Arena
191006 Dalkurd - IK Frej Täby	84	Gävlevallen
191006 Malmö FF - IFK Göteborg	73	Stadion
191020 Hammarby - Malmö FF	70	Tele2 Arena
191028 Malmö FF - AIK	44	Stadion
191028 Djurgårdens IF FF - Örebro SK	32	Tele2 Arena
191102 Hammarby - BK Häcken	85	Tele2 Arena

Table A.1: The games and their number of tags in the provided data.

Attack Datasets		
Training	Samples	Events
190609 Dalkurd - Halmstad	146	73
190625 Dalkurd - Västerås SK	186	93
190706 Malmö FF - Örebro SK	132	66
190707 Hammarby - Falkenbergs FF	166	83
190714 Djurgården - Malmö FF	138	69
190721 Malmö FF - IK Sirius FK	184	92
190722 Hammarby - IF Elfsborg	156	78
190810 Djurgården - IK Sirius FK	156	78
190811 Hammarby - Helsingborgs IF	96	48
190817 Hammarby - GIF Sundsvall	156	78
190818 Dalkurd - GAIS	136	68
190818 Malmö FF - Falkenbergs FF	126	63
190901 Dalkurd - Norrby IF	134	67
190915 Hammarby - IFK Göteborg	116	58
190915 Malmö FF - IFK Norrköping	144	72
190916 Djurgården - Helsingborgs IF	138	69
190921 Dalkurd - Östers IF	152	76
190922 Hammarby - AIK	78	39
190924 Djurgården - Falkenbergs FF	138	69
190929 Dalkurd - Trelleborgs FF	154	77
191006 Djurgården - Hammarby	168	84
191006 Dalkurd - IK Frej Täby	246	123
191006 Malmö FF - IFK Göteborg	234	117
191020 Hammarby - Malmö FF	144	72
191028 Malmö FF - AIK	106	53
191028 Djurgårdens IF FF - Örebro SK	122	61
191102 Hammarby - BK Häcken	298	149
Validation	Samples	Events
190803 Dalkurd - Jönköpings Södra IF	108	54
190825 Malmö FF - Djurgården	104	52
190930 Hammarby IF - Örebro SK	152	76
Test	Samples	Events
190720 Dalkurd - Varbergs BoIS FC	100	50
190728 Djurgården - BK Häcken	144	72
190926 Malmö FF - Helsingborgs IF	142	71

Table A.2: The number of samples and corresponding events for each game in the attack datasets.

Whistle Data	
Game	Whistle tags
190625 Dalkurd - Västerås SK	82
190818 Dalkurd - GAIS	75
190915 Malmö FF - IFK Norrköping	84
190922 Hammarby - AIK	78
191006 Djurgården - Hammarby	72
191028 Malmö FF - AIK	74

Table A.3: The games and their number of whistle tags in the whistle data.

Whistle Datasets		
Training	Samples	Events
190625 Dalkurd - Västerås SK	114	57
190818 Dalkurd - GAIS	104	52
190915 Malmö FF - IFK Norrköping	116	58
190922 Hammarby - AIK	106	53
191006 Djurgården - Hammarby	100	50
191028 Malmö FF - AIK	102	51
Validation	Samples	Events
190625 Dalkurd - Västerås SK	24	12
190818 Dalkurd - GAIS	22	11
190915 Malmö FF - IFK Norrköping	26	13
190922 Hammarby - AIK	24	12
191006 Djurgården - Hammarby	22	11
191028 Malmö FF - AIK	22	11
Test	Samples	Events
190625 Dalkurd - Västerås SK	26	13
190818 Dalkurd - GAIS	24	12
190915 Malmö FF - IFK Norrköping	26	13
190922 Hammarby - AIK	24	12
191006 Djurgården - Hammarby	22	11
191028 Malmö FF - AIK	24	12

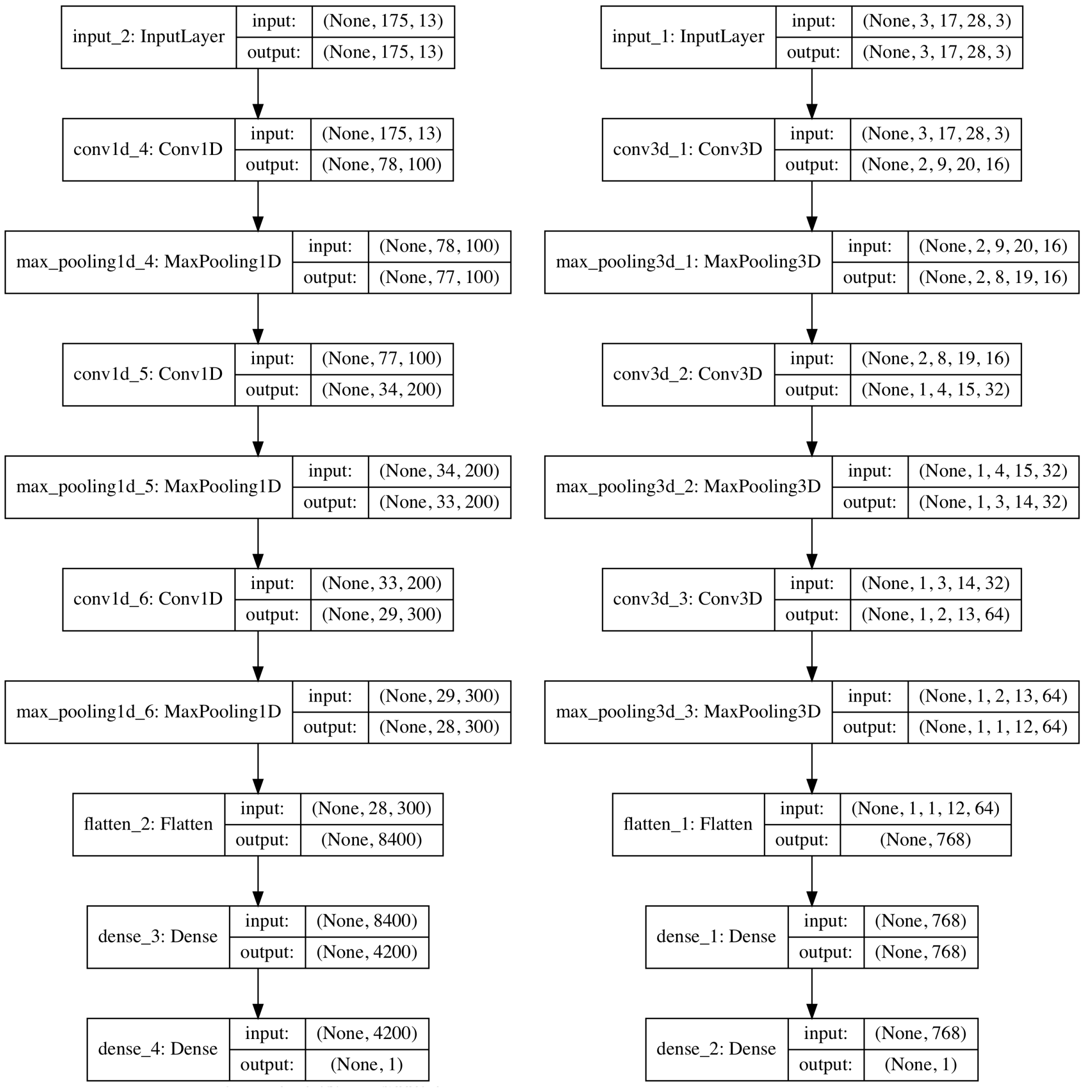
Table A.4: The number of samples and corresponding events for each game in the whistle dataset and the updated whistle dataset by hard negative mining.

Whistle Event Data		
190803 Dalkurd - Jönköpings Södra IF	Detection tags	Whistle tags
Audience	5	-
Speaker	0	-
Kickoff	4	4
End of half	4	6
Free kick	32	31
Penalty	0	0
Offside	4	3
Substitution	2	4
Interruption	3	4
Others	47	57
190825 Malmö FF - Djurgården	Detection tags	Whistle tags
Audience	22	-
Speaker	5	-
Kickoff	2	3
End of half	2	5
Free kick	24	24
Penalty	0	0
Offside	3	4
Substitution	4	4
Interruption	4	1
Others	22	37
190930 Hammarby IF - Örebro SK	Detection tags	Whistle tags
Audience	26	-
Speaker	17	-
Kickoff	6	8
End of half	1	4
Free kick	10	12
Penalty	0	0
Offside	1	2
Substitution	5	4
Interruption	4	3
Others	19	27

Table A.5: The games and their number of detection and whistle event tags in the whistle event data.

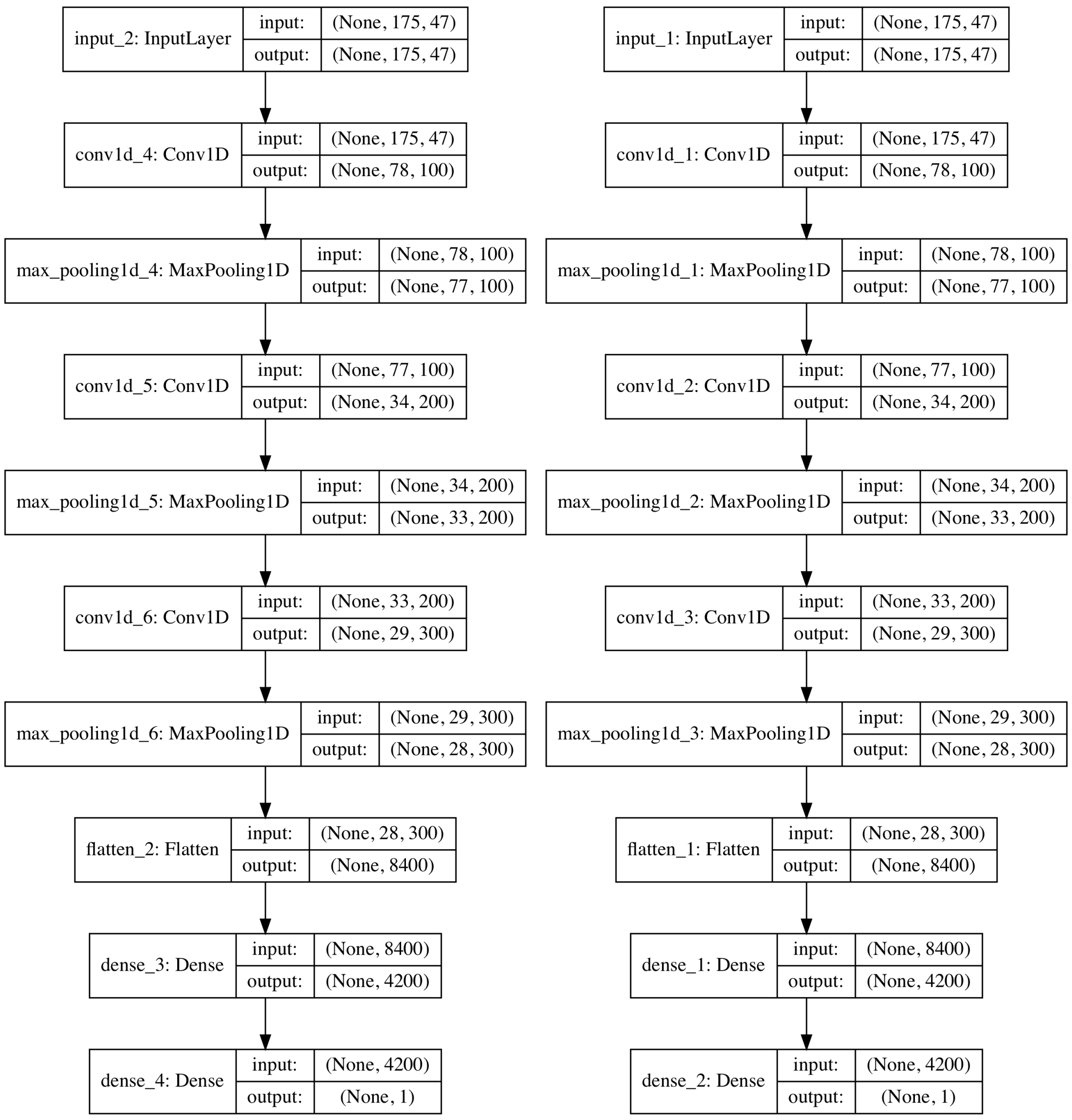
Whistle Event Dataset					
Training	Samples	Kickoff	Free kick	Others	Surround.
190803 Dalkurd - Jönköpings Södra IF	93	4	31	55	3
190825 Malmö FF - Djurgården	85	1	25	41	18
190930 Hammarby IF - Örebro SK	78	8	10	30	30
Validation	Samples	Kickoff	Free kick	Others	Surround.
190803 Dalkurd - Jönköpings Södra IF	21	1	7	12	1
190825 Malmö FF - Djurgården	20	1	6	9	4
190930 Hammarby IF - Örebro SK	16	2	2	6	6
Test	Samples	Kickoff	Free kick	Others	Surround.
190803 Dalkurd - Jönköpings Södra IF	21	1	7	12	1
190825 Malmö FF - Djurgården	21	1	6	9	5
190930 Hammarby IF - Örebro SK	19	2	3	7	7

Table A.6: The number of samples and corresponding events for each game in the whistle event dataset.


input_2: InputLayer | input: (None, 175, 13) | output: (None, 175, 13)
conv1d_4: Conv1D | input: (None, 175, 13) | output: (None, 78, 100)
max_pooling1d_4: MaxPooling1D | input: (None, 78, 100) | output: (None, 77, 100)
conv1d_5: Conv1D | input: (None, 77, 100) | output: (None, 34, 200)
max_pooling1d_5: MaxPooling1D | input: (None, 34, 200) | output: (None, 33, 200)
conv1d_6: Conv1D | input: (None, 33, 200) | output: (None, 29, 300)
max_pooling1d_6: MaxPooling1D | input: (None, 29, 300) | output: (None, 28, 300)
flatten_2: Flatten | input: (None, 28, 300) | output: (None, 8400)
dense_3: Dense | input: (None, 8400) | output: (None, 4200)
dense_4: Dense | input: (None, 4200) | output: (None, 1)
input_1: InputLayer | input: (None, 3, 17, 28, 3) | output: (None, 3, 17, 28, 3)
conv3d_1: Conv3D | input: (None, 3, 17, 28, 3) | output: (None, 2, 9, 20, 16)
max_pooling3d_1: MaxPooling3D | input: (None, 2, 9, 20, 16) | output: (None, 2, 8, 19, 16)
conv3d_2: Conv3D | input: (None, 2, 8, 19, 16) | output: (None, 1, 4, 15, 32)
max_pooling3d_2: MaxPooling3D | input: (None, 1, 4, 15, 32) | output: (None, 1, 3, 14, 32)
conv3d_3: Conv3D | input: (None, 1, 3, 14, 32) | output: (None, 1, 2, 13, 64)
max_pooling3d_3: MaxPooling3D | input: (None, 1, 2, 13, 64) | output: (None, 1, 1, 12, 64)
flatten_1: Flatten | input: (None, 1, 1, 12, 64) | output: (None, 768)
dense_1: Dense | input: (None, 768) | output: (None, 768)
dense_2: Dense | input: (None, 768) | output: (None, 1)

(a) MFCC feature. (b) Position feature.

Figure A.1: The model architectures of the optimal attack models using the MFCC and position feature extraction.


input_2: InputLayer | input: (None, 175, 47) | output: (None, 175, 47)
conv1d_4: Conv1D | input: (None, 175, 47) | output: (None, 78, 100)
max_pooling1d_4: MaxPooling1D | input: (None, 78, 100) | output: (None, 77, 100)
conv1d_5: Conv1D | input: (None, 77, 100) | output: (None, 34, 200)
max_pooling1d_5: MaxPooling1D | input: (None, 34, 200) | output: (None, 33, 200)
conv1d_6: Conv1D | input: (None, 33, 200) | output: (None, 29, 300)
max_pooling1d_6: MaxPooling1D | input: (None, 29, 300) | output: (None, 28, 300)
flatten_2: Flatten | input: (None, 28, 300) | output: (None, 8400)
dense_3: Dense | input: (None, 8400) | output: (None, 4200)
dense_4: Dense | input: (None, 4200) | output: (None, 1)
input_1: InputLayer | input: (None, 175, 47) | output: (None, 175, 47)
conv1d_1: Conv1D | input: (None, 175, 47) | output: (None, 78, 100)
max_pooling1d_1: MaxPooling1D | input: (None, 78, 100) | output: (None, 77, 100)
conv1d_2: Conv1D | input: (None, 77, 100) | output: (None, 34, 200)
max_pooling1d_2: MaxPooling1D | input: (None, 34, 200) | output: (None, 33, 200)
conv1d_3: Conv1D | input: (None, 33, 200) | output: (None, 29, 300)
max_pooling1d_3: MaxPooling1D | input: (None, 29, 300) | output: (None, 28, 300)
flatten_1: Flatten | input: (None, 28, 300) | output: (None, 8400)
dense_1: Dense | input: (None, 8400) | output: (None, 4200)
dense_2: Dense | input: (None, 4200) | output: (None, 1)

(a) Initial whistle model. (b) Updated whistle model.

Figure A.2: The model architectures of the optimal initial and updated whistle models using the spectrogram feature extraction.

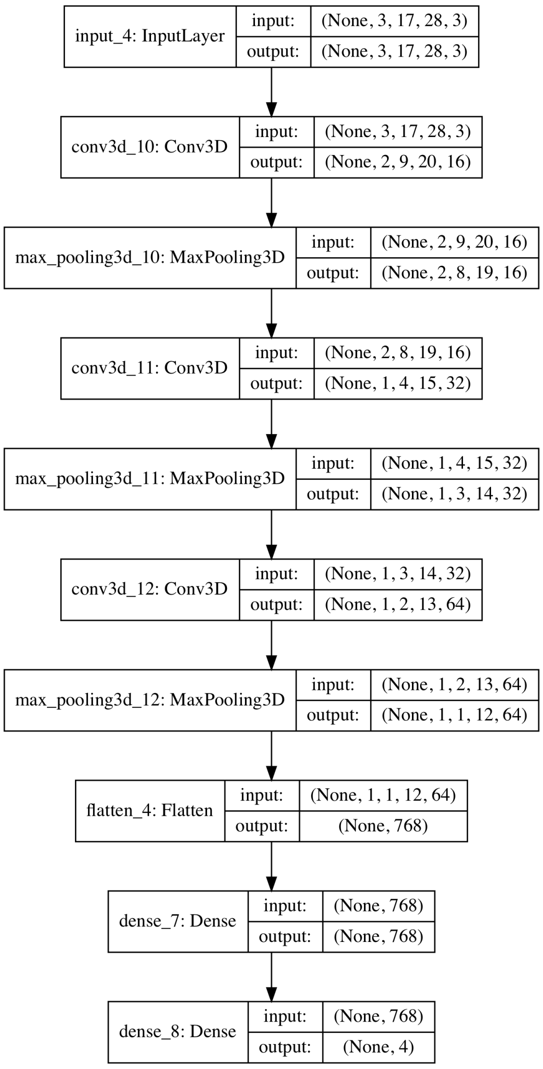

input_4: InputLayer
input: (None, 3, 17, 28, 3)
output: (None, 3, 17, 28, 3)
conv3d_10: Conv3D
input: (None, 3, 17, 28, 3)
output: (None, 2, 9, 20, 16)
max_pooling3d_10: MaxPooling3D
input: (None, 2, 9, 20, 16)
output: (None, 2, 8, 19, 16)
conv3d_11: Conv3D
input: (None, 2, 8, 19, 16)
output: (None, 1, 4, 15, 32)
max_pooling3d_11: MaxPooling3D
input: (None, 1, 4, 15, 32)
output: (None, 1, 3, 14, 32)
conv3d_12: Conv3D
input: (None, 1, 3, 14, 32)
output: (None, 1, 2, 13, 64)
max_pooling3d_12: MaxPooling3D
input: (None, 1, 2, 13, 64)
output: (None, 1, 1, 12, 64)
flatten_4: Flatten
input: (None, 1, 1, 12, 64)
output: (None, 768)
dense_7: Dense
input: (None, 768)
output: (None, 768)
dense_8: Dense
input: (None, 768)
output: (None, 4)

Figure A.3: The model architecture of the optimal whistle event model using the position feature extraction.

B Appendix B

This appendix includes further details regarding the natural language generation task of the degreee project. The preverbal messages constructed for detections are given by Figures B.1–3, tags by Figures B.4–13, welcome by Figure B.14 and goodbye by B.15. The games and their number of tags in the evaluation data are given by Table B.1.

$$\begin{bmatrix} \text{category:} & \texttt{detection} \\ \text{event:} & \texttt{attack} \\ \text{phase:} & \begin{array}{l}\texttt{initial/}\\ \texttt{switch\_gravity\_location/}\\ \texttt{switch\_gravity\_direction/}\\ \texttt{switch\_movement\_location/}\\ \texttt{switch\_movement\_direction/}\\ \texttt{ongoing}\end{array} \\ \text{minute:} & \texttt{int/None} \\ \text{team:} & \begin{bmatrix}\text{home/away/half:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\\ \text{score:} & \texttt{int}\end{bmatrix}\end{bmatrix} \\ \text{centerOfGravity:} & \begin{bmatrix}\text{zone:} & \texttt{str}\\ \text{direction:} & \texttt{str}\end{bmatrix} \\ \text{maximumOfMovement:} & \begin{bmatrix}\text{zone:} & \texttt{str}\\ \text{direction:} & \texttt{str}\end{bmatrix} \end{bmatrix}$$

Figure B.1: The preverbal message of an attack detection. If the game is not ongoing, the minute is set to `None`. The score of the team corresponds to the number of attacks carried out, which is increased if the phase is `initial`. If the team locations are determined, the team is set to home or away with the corresponding name. Otherwise, the team is set to the half of occurrence and the name to `None`. The zones and directions of the center of gravity and maximum of movement correspond to the content of the commentator state. Notice, the half of the team can be determined given the location of the center of gravity and the score given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{detection} \\ \text{event:} & \texttt{sound} \\ \text{phase:} & \texttt{initial/ongoing} \\ \text{power:} & \texttt{float} \\ \text{period:} & \texttt{before/1st\_half/break/2nd\_half/after} \\ \text{minute:} & \texttt{int/None} \end{bmatrix}$$

Figure B.2: The preverbal message of a sound detection. If the game is not ongoing, the minute is set to `None`.

$$\begin{bmatrix} \text{category:} & \texttt{detection} \\ \text{event:} & \texttt{whistle} \\ \text{type:} & \texttt{kickoff/freekick/others/surroundings/undecided} \\ \text{phase:} & \texttt{initial/continuation/ongoing} \\ \text{score:} & \texttt{int} \\ \text{minute:} & \texttt{int/None} \end{bmatrix}$$

Figure B.3: The preverbal message of a whistle detection. If the game is not ongoing, the minute is set to `None`. The score corresponds to the number of whistles detected for the corresponding type. Notice, the score can be determined given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{kickoff} \\ \text{type:} & \texttt{1st\_half/2nd\_half} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \begin{bmatrix} \text{home/away/half/None:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int/None} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure B.4: The preverbal message of a kickoff tag. The score of the team corresponds to the number of kickoffs taken. If able to determine the half of the team and the team locations are determined, the team is set to home or away, else the half. If unable to determine the half of the team, the team and corresponding score are set to `None`. If the team is not provided, the name is set to `None`. Notice, the half of the team can not be determined given the location of the center of gravity. The score of the team can be determined and the half of the team possibly given the gathered statistics during the game.

$$\left[\begin{array}{ll} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{end\_of\_half} \\ \text{type:} & \texttt{1st\_half/2nd\_half} \\ \text{minute:} & \texttt{int} \end{array}\right]$$

Figure B.5: The preverbal message of an end of half tag.

$$\left[\begin{array}{ll} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{goal} \\ \text{type:} & \texttt{shot/header/freekick/penalty/own\_goal} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \left[\begin{array}{ll} \text{home/left:} & \left[\begin{array}{ll} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{array}\right] \\ \text{away/right:} & \left[\begin{array}{ll} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{array}\right] \end{array}\right] \\ \text{player:} & \left[\begin{array}{ll} \text{home/away/half:} & \left[\begin{array}{ll} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int/None} \end{array}\right] \end{array}\right] \end{array}\right]$$

Figure B.6: The preverbal message of a goal tag. If the type is not provided, the type is set to `shot`. The scores of the teams and player correspond to the number of goals scored respectively. If the team locations are determined, the teams are set to home and away with the corresponding name. Otherwise, the teams are set to left and right and the corresponding name to `None`. If the team locations are determined, the player is set to home or away, else the half of occurrence. If the player is not provided, the name and score are set to `None`. Notice, the half of the teams and player can be determined given the location of the center of gravity and the scores given the gathered statistics during the game.

$$\left[\begin{array}{ll} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{corner} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \left[\begin{array}{ll} \text{home/away/half:} & \left[\begin{array}{ll} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{array}\right] \end{array}\right] \end{array}\right]$$

Figure B.7: The preverbal message of a corner tag. The score of the team corresponds the number of corners conceded. If the team locations are determined, the team is set to home or away, else the half of occurrence. If the team is not provided, the name is set to `None`. Notice, the half of the team can be determined given the location of the center of gravity and the score of the team given the gathered statistics during the game.

$$\begin{bmatrix}
\text{category:} & \texttt{tag} \\
\text{event:} & \texttt{shot} \\
\text{type:} & \texttt{shot/header/freekick/penalty} \\
\text{action:} & \texttt{wide/blocked/saved/post/bar/None} \\
\text{minute:} & \texttt{int} \\
\text{team:} & \begin{bmatrix} \text{home/away/half:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{bmatrix} \end{bmatrix} \\
\text{player:} & \begin{bmatrix} \text{home/away/half:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int/None} \end{bmatrix} \end{bmatrix}
\end{bmatrix}$$

Figure B.8: The preverbal message of a shot tag. If the type is not provided, the type is set to `shot`. The scores of the team and player correspond to the number of finishes taken respectively. If the team locations are determined, the team is set to home or away, else the half of occurrence. If the team is not provided, the name is set to `None`. If the team locations are determined, the player is set to home or away, else the half of occurrence. If the player is not provided, the name and score are set to `None`. Notice, the half of the team and player can be determined given the location of the center of gravity and the score given the gathered statistics during the game.

$$\begin{bmatrix}
\text{category:} & \texttt{tag} \\
\text{event:} & \texttt{penalty} \\
\text{minute:} & \texttt{int} \\
\text{team:} & \begin{bmatrix} \text{home/away/half:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{bmatrix} \end{bmatrix} \\
\text{player:} & \begin{bmatrix} \text{home/left/0:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \end{bmatrix} \\ \text{away/right/1:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \end{bmatrix} \end{bmatrix}
\end{bmatrix}$$

Figure B.9: The preverbal message of a penalty tag. The score of the team corresponds to the number of penalties conceded. If the team locations are determined, the team is set to home or away, else the half of occurrence. If the team is not provided, the name is set to `None`. If able to determine the teams of the players and the team locations are determined, the players are set to home and away, else left and right. If unable to determine the teams of the players, the players are set to an `int`. If a player is not provided, the name is set to `None`. Notice, the half of the team can be determined given the location of the center of gravity and the score of the team and the halves of the players possibly given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{freekick} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \begin{bmatrix} \text{home/away/half/None:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int/None} \end{bmatrix} \end{bmatrix} \\ \text{player:} & \begin{bmatrix} \text{home/left/0:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \end{bmatrix} \\ \text{away/right/1:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure B.10: The preverbal message of a free kick tag. The score of the team corresponds to the number of free kicks conceded. If able to determine the half of the team and the team locations are determined, the team is set to home or away, else the half. If unable to determine the half of the team, the team and corresponding score are set to `None`. If the team is not provided, the name is set to `None`. If able to determine the teams of the players and the team locations are determined, the players are set to home and away, else left and right. If unable to determine the teams of the players, the players are set to an `int`. If a player is not provided, the name is set to `None`. Notice, the half of the team can not be determined given the location of the center of gravity. The score of the team can be determined and the halves of the team and players possibly given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{yellow\_card} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \begin{bmatrix} \text{home/away/half/None:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int/None} \end{bmatrix} \end{bmatrix} \\ \text{player:} & \begin{bmatrix} \text{home/away/half/None:} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & 1 \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure B.11: The preverbal message of a yellow card tag. The scores of team and player correspond to the number of yellow cards conceded respectively, which is always one for the player. If able to determine the half of the team and the team locations are determined, the team is set to home or away, else the half. If unable to determine the half of the team, the team and corresponding score are set to `None`. If the team is not provided, the name is set to `None`. If able to determine the half of the player and the team locations are determined, the player is set to home or away, else the half. If unable to determine the half of the player, the player is set to `None`. If the player is not provided, the name is set to `None`. Notice, the half of the team and player can not be determined given the location of the center of gravity. The score of the team can be determined and the halves of the team and player possibly given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{red\_card} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \begin{bmatrix}\text{home/away/half/None:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\\ \text{score:} & \texttt{int/None}\end{bmatrix}\end{bmatrix} \\ \text{player:} & \begin{bmatrix}\text{home/away/half/None:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\\ \text{score:} & \texttt{1}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

Figure B.12: The preverbal message of a red card tag, which is constructed similar to the preverbal message of a yellow card.

$$\begin{bmatrix} \text{category:} & \texttt{tag} \\ \text{event:} & \texttt{substitution} \\ \text{minute:} & \texttt{int} \\ \text{team:} & \begin{bmatrix}\text{home/away/half/None:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\\ \text{score:} & \texttt{int}\end{bmatrix}\end{bmatrix} \\ \text{player:} & \begin{bmatrix}\text{in/0:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\end{bmatrix}\\ \text{out/1:} & \begin{bmatrix}\text{name:} & \texttt{str/None}\end{bmatrix}\end{bmatrix}\end{bmatrix}$$

Figure B.13: The preverbal message of a substitution tag. The score of the team corresponds to the number of substitutions performed. If able to determine the half of the team and the team locations are determined, the team is set to home or away, else the half. If unable to determine the half of the team, the team and corresponding score are set to `None`. If the team is not provided, the name is set to `None`. If able to determine the player being substituted, the players are set to in and out, else an `int`. If a player is not provided, the name is set to `None`. Notice, the half of the team can not be determined given the location of the center of gravity. The score of the team can be determined and the player being substituted possibly given the gathered statistics during the game.

$$\begin{bmatrix} \text{category:} & \texttt{welcome} \\ \text{team:} & \begin{bmatrix}\text{home} & \begin{bmatrix}\text{name:} & \texttt{str}\end{bmatrix}\\ \text{away} & \begin{bmatrix}\text{name:} & \texttt{str}\end{bmatrix}\end{bmatrix} \\ \text{arena:} & \texttt{str} \\ \text{date:} & \texttt{str} \\ \text{season:} & \texttt{spring/summer/autumn/winter}\end{bmatrix}$$

Figure B.14: The preverbal message of welcome.

$$\begin{bmatrix} \text{category:} & \texttt{goodbye} \\ \text{team:} & \begin{bmatrix} \text{home/left} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{bmatrix} \\ \text{away/right} & \begin{bmatrix} \text{name:} & \texttt{str/None} \\ \text{score:} & \texttt{int} \end{bmatrix} \end{bmatrix} \end{bmatrix}$$

Figure B.15: The preverbal message of goodbye. The scores of the teams correspond to the number of goals scored. If the team locations are determined, the teams are set to home and away with the corresponding name. Otherwise, the teams are set to left and right and the corresponding name to `None`. Notice, the scores of the teams can be determined given the gathered statistics during the game.

Evaluation Data		
Game	Tags	Arena
200806 Kalmar FF - Örebro SK	82	Guldfågeln Arena
200813 Kalmar FF - IFK Göteborg	66	Guldfågeln Arena
200824 Kalmar FF - Mjällby AIF	60	Guldfågeln Arena
200914 Kalmar FF - IFK Norrköping	68	Guldfågeln Arena
200928 Kalmar FF - Varbergs BoIS FC	85	Guldfågeln Arena

Table B.1: The games and their number of tags in the evaluation data.

C Appendix C

This appendix presents the generated comments of the commentator in Chapter 9, if run on the game Kalmar FF - Örebro SK, 6 August 2020, recorded from Guldfågeln Arena in Kalmar. The times correspond to occurrences into the recording and corresponding texts to the generated comments of the detected events. In addition, the occurrences of the team location determinations are printed. Notice, the prediction threshold of the whistle model is adjusted manually in accordance with Section 11.2 and the comments are generated in the Swedish language.

```
Commentator
--------------------------------------------------------------------------------
game_id      1859a748-6b19-4833-97f6-4c4c01b1ef9b
title        200806 Kalmar FF - Örebro SK
--------------------------------------------------------------------------------

None
Vi hälsar välkommen till dagens drabbning som står mellan Kalmar FF-Örebro SK
på Guldfågeln Arena. Vi ser fram emot en målrik batalj denna sommardag.

00:00:47
Domaren blåser.

00:00:48
Supportrarna sjunger igång sina lag inför dagens kamp.

00:06:37
Publiken hälsar lagen välkomna inför bataljen.

00:07:01
Domaren blåser i visselpipan.

00:07:07
Domaren markerar med visselpipan! En frispark är tilldelad för händelsen.

00:07:29
Domaren visslar. Frispark tilldelad för händelsen.
```

00:07:46
Domaren blåser.

00:07:47
Publiken hejar igång sina lag inför kampen!

00:08:08
Domaren markerar med pipan.

00:10:04
Domaren blåser för avspark. Kampen är igång!

00:10:11
Ny attack pågår mot mitten från mitten. Samtidigt löpning mot mål längs höger sida.

00:10:32
Spelare fyller på framåt på vänsterkanten.

00:10:34
Spelare fyller på mot box från mitten av banan.

00:11:03
Domaren markerar för avspark. Kampen är nu igång igen!

00:11:05
Ny attack startar mot box från mitten.

00:11:16
Nytt anfall startar mot mitten centralt i banan. Löpning kommer samtidigt mot box från vänster sida.

00:11:48
Nytt anfall pågår mot mitten från vänster. Samtdigt fyller spelare på framåt från vänster.

00:12:42
Domaren markerar med pipan!

00:13:29
Ny attack mot mitten längs högerkanten.

00:13:32
Attacken fortsätter i ny riktning framåt i banan från höger.

00:13:45
Domaren visslar.

```
00:15:11
Attack startar mot mål i mitten av plan.

00:16:51
Domaren markerar för avspark. Kampen är igång!

Determination: team locations
home    right   Kalmar FF
away    left    Örebro SK

00:18:28
Bortalaget startar en attack mot mål centralt i banan. Spelare fyller på framåt
i banan längs högerkanten.

00:18:30
Spelare kommer i riktning mot box längs höger sida.

00:18:34
Domaren markerar med pipan.

00:19:08
Domaren blåser i pipan för avspark. Kampen är igång igen!

00:19:24
Örebro SK påbörjar en ny attack mot box längs vänster sida. Spelare fyller på
framåt från vänster.

00:19:32
Ny löpning framåt i banan på höger sida.

00:19:34
Attacken fortgår mot mitten i mitten av plan. Löpning samtidigt framåt i plan
längs höger sida.

00:19:44
Domaren visslar i pipan!

00:21:19
Kalmar FF startar en attack mot mitten centralt i plan.

00:22:05
Örebro SK påbörjar en attack mot mitten centralt i plan. Samtidigt fyller
spelare på mot mitten på högerkanten.

00:22:22
Domaren blåser i visselpipan!
```

00:22:36
Örebro SK inleder ett anfall mot mål i mitten av plan. Löpning samtidigt mot mål på högerkanten.

00:22:51
Domaren markerar med pipan. Frispark delgiven för företeelsen.

00:23:14
Kalmar FF startar ett nytt anfall mot mitten centralt i banan. Spelare kommer samtidigt mot box på vänsterkanten.

00:23:25
Domaren markerar med pipan!

00:24:03
Domaren markerar med pipan!

00:25:54
Domaren markerar!

00:26:17
Bortalaget inleder en ny attack mot mitten centralt i plan. Spelare kommer samtidigt mot mål från högerkanten.

00:26:55
Domaren visslar.

00:27:15
Domaren visslar!

00:27:25
Supportrarna uttrycker deras stöd!

00:27:51
Domaren blåser i pipan!

00:27:58
Hemmalaget inleder ett nytt anfall mot mål från högerkanten. Spelare fyller på mot mitten centralt.

00:29:09
Domaren visslar med pipan! Frispark är tilldömd för händelsen.

00:30:07
Domaren visslar! Frispark är visad för regelvidrigheten.

00:30:16
Örebro SK startar en ny attack mot box centralt i banan. Samtidigt löpning framåt i banan från höger sida.

00:30:18
Spelare kommer mot box längs vänster sida.

00:30:31
Hemmalaget påbörjar en attack mot mitten från mitten av plan. Löpning mot box längs vänster sida.

00:31:39
Bortalaget inleder en ny attack framåt från vänsterkanten. Spelare fyller på mot box centralt i plan.

00:33:26
Domaren markerar!

00:34:04
Domaren visslar!

00:34:20
Hemmalaget inleder ett anfall mot mål centralt i banan. Samtidigt kommer spelare mot mål längs vänsterkanten.

00:34:22
Ny löpning kommer framåt längs vänsterkanten.

00:34:27
Domaren visslar med pipan!

00:34:33
Domaren blåser i pipan!

00:35:03
Bortalaget påbörjar ett anfall mot mål i mitten. Samtidigt löpning mot mitten på högerkanten.

00:35:05
Ny löpning mot box centralt.

00:36:25
Domaren blåser. En frispark är tilldömd för händelsen.

00:36:55
Domaren visslar. En frispark är tilldelad för regelvidrigheten.

00:37:04
Kalmar FF startar ett nytt anfall mot mitten från mitten av plan.

00:37:22
Domaren markerar med pipan!

00:37:52
Domaren blåser i visselpipan! En frispark tilldömd för regelbrottet.

00:38:13
Domaren visslar med pipan för avspark. Matchen i spel.

00:38:26
Kalmar FF startar en ny attack mot mitten från mitten av plan.

00:38:29
Ny spelare kommer i riktning mot höger centralt.

00:39:13
Bortalaget startar en attack mot mål i mitten av plan.

00:39:15
Ny löpning kommer mot box längs vänsterkanten.

00:39:42
Ny spelare fyller på mot mitten från mitten.

00:39:43
Örebro SK påbörjar en attack mot box från mitten av plan. Löpning mot mitten på höger sida.

00:40:24
Domaren markerar.

00:40:43
Hemmalaget startar ett nytt anfall mot mål centralt.

00:40:53
Örebro SK startar ett nytt anfall mot mål från mitten. Spelare fyller samtidigt på mot box från höger sida.

00:40:54
Domaren blåser!

00:41:43
Örebro SK startar ett anfall mot box centralt i banan.

00:41:45
Ny spelare fyller på mot box från högerkanten.

00:42:27
Örebro SK startar ett anfall framåt i banan från vänster sida. Löpning kommer mot vänster sida från mitten.

00:44:00
Hemmalaget startar ett anfall mot mål på högerkanten. Spelare kommer samtidigt mot box centralt i banan.

00:45:59
Domaren blåser i visselpipan!

00:46:20
Hemmalaget startar ett anfall framåt från högerkanten.

00:46:33
Bortalaget inleder en ny attack mot box från mitten av banan. Löpning mot box från höger.

00:47:00
Domaren visslar i pipan!

00:47:25
Domaren blåser i visselpipan.

00:47:40
Domaren visslar med pipan.

00:48:06
Kalmar FF påbörjar ett nytt anfall mot box längs höger sida.

00:48:08
Löpning kommer mot mitten från mitten.

00:48:45
Domaren visslar med pipan! En frispark visad för regelvidrigheten.

00:50:19
Bortalaget inleder ett nytt anfall mot mitten i mitten. Löpning mot mål från högerkanten.

00:50:55
Domaren visslar med pipan!

00:51:31
Domaren markerar med pipan.

00:53:32
Kalmar FF påbörjar en attack mot box från mitten av plan. Löpning kommer samtidigt framåt längs höger sida.

00:54:18
Domaren visslar i pipan.

```
00:54:45
Bortalaget startar ett anfall mot mål i mitten av plan.

00:54:49
Attacken fortgår i riktning mot vänster sida centralt. Löpning kommer framåt i
banan längs vänster sida.

00:55:17
Omgivningen uttrycker sin åsikt!

Switch: team locations
home    left    Kalmar FF
away    right   Örebro SK

00:58:21
Domaren markerar!

01:11:19
Domaren visslar med pipan för avspark. Matchen är igång igen!

01:11:53
Domaren visslar med pipan.

01:12:16
Domaren blåser!

01:12:56
Domaren visslar.

01:13:42
Domaren visslar i pipan.

01:14:57
Domaren visslar i pipan.

01:15:06
Domaren visslar.

01:18:29
Domaren markerar med visselpipan.

01:18:37
Kalmar FF inleder ett nytt anfall mot vänsterkanten centralt i plan. Spelare
kommer samtidigt framåt från vänster sida.

01:18:42
Domaren blåser i visselpipan!
```

01:18:56
Domaren blåser.

01:19:11
Domaren blåser i pipan!

01:20:04
Domaren blåser!

01:20:29
Domaren blåser i pipan för avspark. Matchen är nu igång igen!

01:20:56
Kalmar FF inleder ett nytt anfall mot mitten från mitten. Spelare kommer samtidigt mot box på vänster sida.

01:20:58
Löpning kommer framåt i plan från höger sida.

01:21:10
Domaren visslar med pipan!

01:21:51
Domaren markerar med visselpipan!

01:22:13
Domaren markerar med visselpipan!

01:22:53
Domaren markerar.

01:23:19
Kalmar FF startar ett nytt anfall mot mål centralt. Löpning kommer framåt på högerkanten.

01:24:10
Bortalaget startar en attack mot mitten från mitten. Löpning samtidigt mot vänster från mitten.

01:24:13
Löpning kommer mot mål från vänsterkanten.

01:24:16
Omgivningen gör sig hörd!

01:24:18
Domaren visslar i pipan!

```
01:25:39
Domaren visslar.

01:25:46
Domaren visslar med pipan! En frispark är tilldelad för händelsen.

01:26:11
Domaren blåser i pipan!

01:26:26
Domaren markerar.

01:27:06
Bortalaget startar ett anfall mot box längs högerkanten.

01:27:08
Supportrarna uttrycker deras stöd!

01:27:46
Domaren blåser i pipan.

01:28:25
Domaren visslar med pipan för avspark. Kampen i spel återigen!

01:28:46
Hemmalaget påbörjar ett anfall mot mål centralt i banan. Löpning kommer
samtidigt mot mitten på högerkanten.

01:28:48
Anfallet fortsätter i riktning mot vänsterkanten centralt i banan.

01:29:11
Bortalaget inleder ett nytt anfall mot mitten i mitten.

01:32:17
Domaren visslar.

01:32:52
Hemmalaget påbörjar en ny attack mot mitten i mitten.

01:33:55
Domaren markerar med visselpipan!

01:34:55
Domaren visslar med pipan! En frispark är tillgiven för händelsen.

01:35:43
Örebro SK inleder ett anfall mot mitten på höger sida.
```

01:35:45
Spelare kommer mot box från vänster sida.

01:35:47
Anfallet fortsätter mot box från mitten av banan. Samtidigt kommer löpning mot mål längs högerkanten.

01:35:51
Domaren blåser i pipan.

01:36:41
Örebro SK startar en attack mot mål i mitten av plan.

01:36:45
Supportrarna reagerar!

01:37:24
Domaren visslar i pipan.

01:37:33
Domaren markerar med visselpipan för avspark. Matchen nu igång igen.

01:37:46
Kalmar FF startar en attack mot mål längs vänster sida. Samtidigt löpning mot mitten från högerkanten.

01:38:10
Kalmar FF påbörjar ett nytt anfall mot box i mitten av plan. Löpning samtidigt mot mitten från höger sida.

01:38:12
Löpning kommer mot box centralt i banan.

01:39:27
Domaren markerar.

01:40:09
Örebro SK startar ett nytt anfall mot mål från mitten av banan. Samtidigt kommer spelare mot box från vänster sida.

01:40:14
Domaren visslar med pipan. Frispark tillgiven för regelbrottet.

01:41:13
Domaren visslar med pipan. En frispark är delgiven för regelbrottet.

01:42:42
Domaren blåser i visselpipan. En frispark tillgiven för företelesen.

```
01:42:52
Kalmar FF påbörjar ett nytt anfall mot mitten från mitten av banan. Spelare
fyller på framåt i banan längs vänster sida.

01:42:54
Anfallet fortsätter mot vänsterkanten centralt i banan. Samtidigt kommer
löpning framåt i banan längs vänster sida.

01:43:12
Domaren blåser i pipan.

01:43:35
Domaren blåser i visselpipan!

01:45:21
Domaren markerar!

01:46:00
Domaren blåser för avspark. Matchen igång!

01:46:28
Domaren visslar.

01:46:53
Domaren blåser i pipan. Frispark är tillvisad för regelvidrigheten.

01:47:06
Domaren markerar med visselpipan!

01:47:15
Domaren visslar i pipan!

01:47:20
Domaren markerar!

01:47:55
Hemmalaget påbörjar ett anfall mot mål centralt. Spelare kommer samtidigt mot
mål längs vänsterkanten.

01:48:02
Domaren markerar med visselpipan!

01:48:14
Domaren markerar med visselpipan! Frispark är tillgiven för regelvidrigheten.

01:48:38
Domaren visslar i pipan.
```

```
01:48:54
Domaren visslar i pipan!

01:49:44
Domaren visslar.

01:50:21
Domaren blåser!

01:51:10
Örebro SK påbörjar ett nytt anfall mot mål från mitten av plan. Spelare kommer
framåt i banan längs höger sida.

01:51:51
Domaren markerar med pipan.

01:52:19
Kalmar FF inleder ett nytt anfall mot mitten i mitten av plan. Samtidigt
löpning framåt i banan från höger sida.

01:54:35
Domaren blåser i pipan! En frispark tillgiven för regelbrottet.

01:54:54
Domaren markerar med pipan.

01:55:28
Domaren visslar med pipan!

01:56:17
Kalmar FF startar ett nytt anfall mot box från mitten. Spelare kommer mot box
längs vänsterkanten.

01:56:55
Domaren blåser.

01:57:04
Domaren visslar med pipan!

01:57:44
Domaren visslar med pipan.

01:58:04
Domaren visslar i pipan.

01:59:28
Domaren markerar med visselpipan!
```

```
None
Matchen är över och lagen har krigat väl. Resultatet blev 0-3 till Örebro SK.

02:00:26
Domaren blåser i visselpipan!

02:02:30
Domaren blåser! Frispark tillgiven för företelesen.

02:02:50
Domaren blåser i pipan.

-----------------------------------------------------------------
time        01:50:19
```

Bibliography

[1] CBS Pittsburgh. *KDKA Firsts*. `https://pittsburgh.cbslocal.com/2010/04/01/kdka-firsts/`. (Accessed: 2020-11-16). Apr. 2010.

[2] Junfei Qiu et al. "A Survey of Machine Learning for Big Data Processing". In: *EURASIP Journal on Advances in Signal Processing* (May 2016). DOI: `10.1186/s13634-016-0355-x`.

[3] Haohao Jiang, Yao Lu, and Jing Xue. "Automatic Soccer Video Event Detection Based on a Deep Neural Network Combined CNN and RNN". In: *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*. 2016, pp. 490–494. DOI: `10.1109/ICTAI.2016.0081`.

[4] Grigorios Tsagkatakis, Mustafa Jaber, and Panagiotis Tsakalides. "Goal!! Event detection in sports video". In: *Electronic Imaging* 2017 (Jan. 2017), pp. 15–20. DOI: `10.2352/ISSN.2470-1173.2017.16.CVAS-344`.

[5] Min Xu et al. "Creating Audio Keywords for Event Detection in Soccer Video". In: *2003 International Conference on Multimedia and Expo. ICME '03. Proceedings (Cat. No.03TH8698)*. Vol. 2. 2003, pp. II–281. DOI: `10.1109/ICME.2003.1221608`.

[6] Joakim Arpe and Richard Ericsson. "Cost-Optimized Event Detection in Football Video". MA thesis. Lund, Sweden: Lund University, 2020.

[7] Mariët Theune et al. "From Data to Speech: A Generic Approach". In: *Natural Language Engineering* 7.1 (2001), pp. 47–86. ISSN: 1351-3249.

[8] Kumiko Tanaka et al. "MIKE: An Automatic Commentary System for Soccer". In: Aug. 1998, pp. 285–292. ISBN: 0-8186-8500-X. DOI: `10.1109/ICMAS.1998.699067`.

[9] Kevin P. Murphy. *Machine learning : A Probabilistic Perspective*. Cambridge, Massachusetts, London, England: MIT Press, 2013.

[10] Ian J. Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. Cambridge, Massachusetts, London, England: MIT Press, 2016.

[11] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Cambridge, United Kingdom: Springer, 2006.

[12] François Chollet. *Deep Learning with Python.* Shelter Island, New York: Manning Publications Co., 2017.

[13] Richard G. Lyons. *Understanding Digital Signal Processing.* 3rd. USA: Addison-Wesley Longman Publishing Co., Inc., 2011.

[14] Steven B. Davis and Paul Mermelstein. "Comparison of Parametric Representations for Monosyllabic Word Recognition in Continuously Spoken Sentences". In: *IEEE Transactions on Acoustics, Speech, and Signal Processing* 28.4 (1980), pp. 357–366. DOI: `10.1109/TASSP.1980.1163420`.

[15] Robert Rehr and Timo Gerkmann. "Cepstral noise subtraction for robust automatic speech recognition". In: *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP).* 2015, pp. 375–378. DOI: `10.1109/ICASSP.2015.7177994`.

[16] Albert Gatt and Emiel Krahmer. "Survey of the State of the Art in Natural Language Generation: Core Tasks, Applications and Evaluation". In: *Journal of Artificial Intelligence Research* 61.1 (2018), pp. 65–170. ISSN: 1076-9757.

[17] Ehud Reiter and Robert Dale. "Building Applied Natural Language Generation Systems". In: *Natural Language Engineering* 3.1 (1997), pp. 57–87. DOI: `10.1017/S1351324997001502`.

[18] Brian McFee et al. "LibROSA: Audio and Music Signal Analysis in Python". In: *Proceedings of the 14th Python in Science Conference.* Vol. 8. 2015, pp. 18–25.

[19] Robin Gustavsson. "Region of Interest Prediction in Football using Artificial Neural Networks". MA thesis. Lund, Sweden: Lund University, 2017.

[20] Shivam Soni, Sudipta Dey, and M. Sabarimalai Manikandan. "Automatic Audio Event Recognition Schemes for Context-Aware Audio Computing Devices". In: *2019 Seventh International Conference on Digital Information Processing and Communications (ICDIPC).* 2019, pp. 23–28. DOI: `10.1109/ICDIPC.2019.8723713`.

[21] *Keras.* URL: `https://keras.io`.

[22] *TensorFlow.* URL: `https://www.tensorflow.org/`.

[23] *Amazon EC2 G4 Instances.* URL: `https://aws.amazon.com/ec2/instance-types/g4/`.

[24] *Deep Learning AMI (Ubuntu 18.04).* URL: `https://aws.amazon.com/marketplace/pp/Amazon-Web-Services-AWS-Deep-Learning-AMI-Ubuntu-1/B07Y43P7X5`.

Master’s Theses in Mathematical Sciences 2021:E5
ISSN 1404-6342

LUTFMA-3437-2021

Mathematics
Centre for Mathematical Sciences
Lund University
Box 118, SE-221 00 Lund, Sweden

http://www.maths.lth.se/