Modeling and Interpreting CTG Curves from Labor Using Machine Learning and Pattern Recognition

Sandra Kandefelt,
Sara Miranda Perklev

Master's thesis
2022:E50

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematical Statistics

Abstract

The main monitoring method during labor is cardiotocography, CTG, which measures the fetal heartbeat, FHR, as well as the uterine contractions, TOCO. The CTG is a valuable tool in assessing the fetal status and it is evaluated intermittently by clinicians during the progress of the labor. However, abnormalities in the CTG are often non-specific which makes the interpretation difficult and inconsistent. Therefore, a technical solution that could assist and simplify the assessment would be beneficial. By the use of different machine learning methods, this master's thesis investigates the properties and liaisons between the CTG data and the condition of the newborn baby, i.e., the Apgar score. All model-performance measures are evaluated based on 10-fold cross-validation to reduce bias. The thesis covers three linked areas. Firstly, a peak detection model was implemented to indicate peaks in the uterine contraction curve. The best model was a k-nearest neighbors that had a mean sensitivity of 98.70 (0.42) percent. This model was used for the second part; identifying different stages of labor with two approaches. For the classification of labor stage, the best model was a convolutional neural network with 23 consecutive layers. This model had a mean accuracy of 78.80 (0.80) percent. This step was conducted to enable the machine learning algorithms to be trained and evaluated on CTG data from the same stage of labor. In the last part, the data from the second part were used for predicting the fetal status. The results were discouraging, as the machine learning models did not perform considerably better than the fortuity. The best mean F1 score for classifying high and low Apgar scores was 5.26 (0.43) percent, and it was achieved by a decision tree model. The conclusion of this thesis is that additional research is required to establish the conjunction between CTG data and the status of the fetus. Furthermore, a data set with auxiliary samples of low Apgar scores would presumably be valuable in the aspiration of developing a supportive tool for fetal assessment.

Acknowledgements

We would like to extend our deepest gratitude to Andreas Jakobsson, Karl Åström, and Ida Arvidsson, our supervisors at the Faculty of Engineering. Without the weekly meetings, interest in our project, and support throughout the process, this thesis would never have reached satisfying results. Furthermore, we would like to pay our special regards to Karel Maršál and Karin Källén for your guidance within the obstetric field and for helping us understand and navigate the difficulties of monitoring and interpreting CTG during labor. Lastly, a special thanks to Josefine Öder who supported and guided us in the beginning of our project.

Sandra Kandefelt & Sara Perklev, Lund, June 2022

Contents

1. Introduction

The number of births per year in Sweden has varied between 110 000 and 120 000 over the last decade [1]. Continuous development of obstetric and antenatal care, together with the emergence of modern medicine, has since the 1970s significantly reduced the number of perinatal deaths from 1.40 percent in 1973 to 0.41 percent in 2011. The majority of the cases occurs in premature or multiple birth pregnancies. If considering full-time single births, (37+0 weeks) the perinatal mortality rate is 0.60 percent and 0.20 percent, respectively [2].

Labor includes different risks for both mother and infant. The uterine contractions can cause mechanical stress on the fetus affecting the brain or the umbilical cord. This can lead to hypoxia, which is a condition where the tissue lacks sufficient oxygen supply. If the circulation in the placenta decreases, partially or fully, this might lead to metabolic stress since the gas exchange between the mother and fetus is compromised. During the pregnancy, the fetus develops resilience, which enables it to manage these sorts of stresses during labor. However, if the stress levels are unusually high or if the resilience is compromised, the fetus might be in distress. Distress can lead to hypoxia, asphyxia, brain injuries or, in the worst case, death since it affects the vital functions of the fetus [3, 4].

To monitor the fetal status during labor, the most commonly used method is cardiotocography, CTG. CTG consists of continuous and simultaneous measuring of the fetal heart rate and the uterine contractions. The recordings are visually interpreted and evaluated intermittently by clinicians during labor, to find indicators of fetal distress. The assessment is based on the fetal heart rate baseline and variability combined with the duration, intensity, and frequency of the uterine contractions. Fetal tachycardia, i.e., when the baseline heart rate exceeds 160 bpm, and bradycardia, i.e., when the baseline heart rate is less than 100 bpm, could indicate hypoxia. The variability is a heart rate response to the changes in the environment, which originate from the pressure variation due to the contractions. Long term absence of variability is therefore an indicator that the fetus is not responding to the environmental changes, which could indicate that the fetal health is compromised [4].

When the infant is born, an assessment is made based on five parameters: breathing, heart rate, skin color, muscle tone, and response to stimulation. These aspects are given points between zero and two, and the optimal total score is ten. This is known as the Apgar score, and it is evaluated one, five, and ten minutes after the baby is born. A low score increases the risk of neonatal morbidity as well as brain injuries. The Apgar score has been used since the early 1950s and was developed to assess the physical condition of the newborn [5].

Today, there is no technical tool that can support clinicians in evaluating the CTG during labor and assessing fetal status. Since the CTG is relatively hard to interpret [6], and the clinicians often work under tight time constraints, a technical solution that could assist and simplify the assessment of the fetal status would be desirable. The gains of enabling the possibility to identify an alarming situation at an earlier stage are substantial, for both the mother and infant as well as the staff of the hospital.

1.1 Project Goals and Objectives

Considerable expertise is required to interpret CTG curves, and thus determine whether the fetal response is consistent with the uterine contractions or if there is a sign of asphyxia or fetal distress. Even senior obstetricians tend to disagree with the evaluation of CTG curves due to the difficulty of interpretation [6]. This results in an undesired existence of subjectivity and variance in the assessment. With this in mind, there are several incentives to automate and develop a computer system that could visually interpret the CTG curves and give an objective evaluation. The utmost goal is to develop an algorithm that takes the prior CTG curve as input and predicts the outcome of the labor. For example, the output could be the Apgar score or the risk of emergency cesarean section. If the development of a bedside CTG-interpreting computer program is successful, it could help obstetricians in their evaluation of the fetal status, and thus reduce the risk of fetuses suffering from some degree of asphyxia during labor. As a result, it could reduce the risk of incidences of perinatal deaths. Furthermore, an algorithm does not have the constraints of the human mind or vision and could recognize patterns in the data not detectable to the human eye. Accordingly, it could capture and detect aspects in the CTG curves and clinical data that would not be considered in the evaluation made by an obstetrician.

The aim of this project is to construct a peak detector for uterine contractions through different machine learning and deep learning approaches. Deep learning applications master the task of solving complex tasks where there is a large amount of diverse, unstructured, and interconnected data. An accurate classification algorithm for uterine contractions is used to determine the frequency of uterine contractions. The frequency is then used as a tool to classify different stages of labor. These are the first steps in, hopefully, a later project in developing an algorithm that considers both the fetal heart rate and the uterine contractions to predict the outcome of labor.

The objectives of this thesis can be concluded as follows:

- Construct a peak detection algorithm for uterine contractions based on machine learning approaches with a high F1 score and sensitivity.
- Use the best model from the peak detection to classify the labor stages using machine learning approaches with a high F1 score.
- Do initial testing on predicting Apgar score and leave a solid foundation and documentation for continued research on the project.

2. Labor and Cardiotocography

This chapter presents an overview of the use of cardiotocography in modern maternity care. It also covers the five variables used in CTG evaluation, and introduces some artifacts to be aware of. Lastly, a brief explanation of the different stages of labor is presented.

2.1 Cardiotocography

Cardiotocography has been used during labor since the 1970s. It registers the fetal heart rate (cardiography, FHR) via an ultrasound transducer and the uterine contractions (tocography, TOCO) via a pressure sensor, both attached to the mother's abdomen. In figure 2.1, a CTG curve is visualized, where the upper signal is fetal heart rate and the lower signal is the uterine activity.

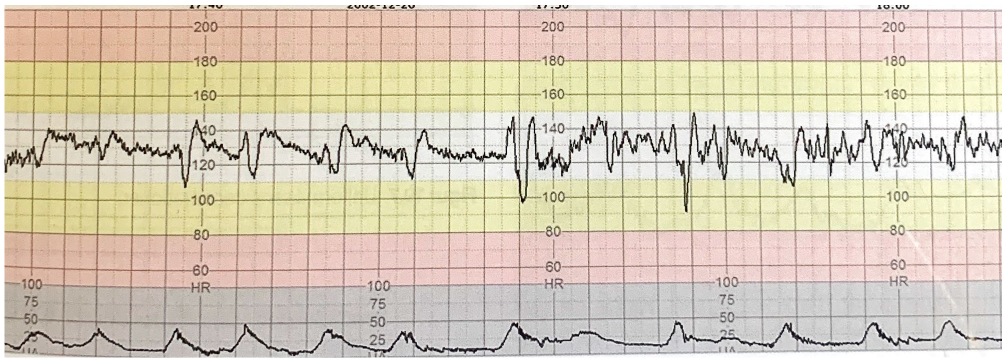


Figure 2.1: Cardiotocography with FHR signal registered at the top and TOCO signal registered at the bottom. Timescale: 1 cm equals 1 minute. [4]

The method aims to monitor fetal well-being and detect fetal distress. In particular, the method is used to diagnose hypoxia and predict asphyxia of the fetus. Asphyxia is the condition of lack of oxygen in the blood and tissue due to impaired gas exchange. During labor and uterine contractions, the fetus is exposed to extra pressure because of the reduced blood flow to the placenta as well as reduced oxygen supply, and thus emergent asphyxia could arise. A healthy fetus has a good capacity to tolerate asphyxia, however, with increased asphyxia the gas exchange is compromised resulting in reduced oxygen concentration and increased carbon dioxide concentration in the blood. Depending on the degree of asphyxia and the fetal vulnerability, the outcome varies [3]. Besides increased mortality, perinatal asphyxia can result in neurological consequences such as cognitive disorder, cerebral palsy, and epilepsy [7]. Accordingly, the use and evaluation of CTG during labor is of great importance. There are high requirements of competence in interpreting CTG among midwives and doctors, to avoid asphyxia and to minimize unnecessary interference during labor. Evaluation of the CTG is a subjective action and experience plays an important role. Nonetheless, there are five variables from the FHR and the uterine activity that the evaluation is based on [4].

2.1.1 Fetal Heart Rate

Four different features of fetal heart rate are evaluated: baseline (bpm), variability (bpm), decelerations, and accelerations [8]. The baseline, measured in beats per minute (bpm), is the average heart rate of the fetus. A normal fetal heart rate lies within 110-160bpm, whereas 100-109 and 161-180 are considered non-reassuring [9]. Baseline bpm below 100 or above 180 is considered abnormal. The variability is also measured in beats per minute and should be at least five. Below five for 40-90 minutes is considered non-reassuring, and if it stays below five for more than 90 minutes or above 25 for more than 30 minutes, it is considered abnormal. During normal labor and delivery, decelerations should not be present. Single prolonged decelerations for up to three minutes, as well as variable decelerations with over 50 percent of contractions, occurring for over 90 minutes, are labeled non-reassuring. Atypical variable decelerations with over 50 percent of contractions or late decelerations, both for over 30 minutes or a single prolonged deceleration for more than three minutes are considered abnormal. Accelerations should be present during labor. If everything appears normal, the absence of accelerations is of uncertain significance. For a CTG to be labeled 'normal', all four evaluation areas should be within the normal range [8].

2.1.2 Uterine Contractions

The uterine contraction is the tightening and relaxing of the uterine muscle. The contractions accomplish two things during labor; effacement and dilation of the cervix, and descending of the fetus into the birth canal. Uterine activity is evaluated through the frequency, intensity, and duration of the contractions [4]. An individual contraction is seen as a peak in the CTG curve measuring uterine activity, known as the TOCO curve. The frequency should not exceed four to five contractions per ten minutes, as this could affect the oxygen supply [4]. Moreover, the uterine contraction should be evaluated in combination with the FHR. They are used in combination to determine the character of the decelerations and evaluate if abnormalities occur. Early decelerations are a mirror of the contractions, while late decelerations begin when the contraction peaks and end after the contraction is over. If late declarations are present in the CTG curve, especially in combination with tachycardia and reduced variability, it is an indication of possible asphyxia, and the labor should be terminated [4].

2.1.3 Artifacts

A CTG measurement records fetal heart rate and uterine activity simultaneously. The FHR signal can be registered via an external doppler transducer placed on the mother's abdomen wall, or internally with an electrode on the fetal scalp. The former method is used to a greater extent due to simplicity. However, it is more prone to register artifacts from the surroundings or from the mother. To reduce the risk of misinterpretation of the FHR there is a filter function that only registers heart rates from 50 to 210 bpm [4]. Moreover, there is a risk that the external doppler sensor registers the maternal heart rate and mistakenly outputs it as the fetal heart rate. This can have great consequences as it would imply that the fetus is not monitored during labor and therefore the risk of missing possible fetal distress is increased. If the quality of the external signal is questionable there is an option to switch to an electrode, given that the cervix is dilated at least 1.5 cm. This would ensure that the fetus is monitored instead of the mother.

The uterine activity is measured with an external pressure sensor on the mother's abdomen wall. This transducer measures the frequency and duration, but not the strength as it can only be measured via an intrauterine pressure catheter [4]. The external pressure sensor is attached with a belt and fastened manually. Given the tightness of the belt and the mother's thickness of the abdomen wall, the sensor outputs different contraction strengths and is more or less prone to register artifacts. For example, when the mother coughs the pressure on the sensor increases, and thus the result is a peak in the TOCO curve. Even though this would be an artifact, this can be manually filtered since the duration of a cough is not in the interval of a contraction. However, this must be considered when filtering and indicating contraction peaks in the TOCO curve.

2.2 Stages of labor

Normal labor is induced spontaneously with uterine contractions leading to cervical changes or water breaking from 37+0 weeks (259 days) to 41+6 weeks (293 days). It is low-risk and proceeds without complications throughout the labor and delivery [10]. The majority of births is of normal character and proceeds through three stages [11]. In the following section, the stages of labor are explained in more detail.

2.2.1 The First Stage of Labor

The first stage of labor is defined as the period from the beginning of labor to full cervical dilation. It can be divided into two subphases, the latent and active phase. The latent stage is when the labor has started but the contractions are yet to come at regular intervals. During the latent phase, the cervix shortens until it is fully effaced and dilates from zero to three centimeters, and the duration can vary between a few hours and days [11]. The active phase is a part of the opening stage, and it is the period from three centimeters to full cervical dilation. It is often shorter and more rapid with established labor work [12]. During the active phase, the cervix often dilates at a rate of one cm per hour. Uterine contractions come in an interval of three to five minutes with a duration of 30-90 seconds. Throughout the opening stage, the contractions increase in intensity [11].

2.2.2 The Second Stage of Labor

The second stage of labor is the time from complete cervix dilation of ten centimeters to the fetus is out of the birth canal [12]. This stage is also known as the expulsion phase and can be divided into two subphases. The passive phase, where the baby's head moves through the cervix, rotates, and descends to the pelvic floor, and the active phase where the woman wants to push spontaneously and actively. The expulsion phase is physically demanding, and 45-60 minutes of active pushing increases the probability of exhaustion, and reduces the chance of spontaneous vaginal labor. The duration of the second stage varies between one and three hours and is dependent on if the woman is a first-time mother and the use of epidural, both increasing the duration. During the expulsion phase, the uterine contractions come more frequently with a duration of 60-180 seconds. The frequency should not exceed four to five contractions per ten minutes, because of the higher risk of compromising the oxygen supply [11].

2.2.3 The Third Stage of Labor

The final stage of labor commences when the child is delivered and proceeds until the delivery of the placenta. The duration is on average a few minutes but it may last up to 30 minutes. If the delivery time is greater than 30 minutes, the risk of postpartum hemorrhage is increased and there may be a need for manual removal of the placenta or other interventions [12]. After the delivery of the placenta, a visual examination is made to ensure full delivery of membrane and tissue.

3. Machine Learning

Machine learning has over the last couple of years become a course of matter for many research projects within modeling and managing large data. It is a sub-field of artificial intelligence that has the capability to self-learn without being explicitly programmed by a human. It can be described as the ability of a machine to resemble intelligent human behavior, make predictions, and make decisions based on the given training data. It can be used in several applications such as speech recognition, computer vision, and medicine where it is complicated and impractical to develop algorithms that can perform complex assignments. In this chapter, different machine learning algorithms are explained briefly and a justification for their use is given.

3.1 Justify the Use of Machine Learning

There are several reasons why machine learning is an appropriate method for CTG interpretation and fetal classification. First, a computerized CTG interpretation would reduce the subjectivity in the interpretation. It would also free up a lot of valuable time as the obstetrician would get continuous information from the trained neural network or machine learning classifier which would classify the outcome of the birth. The unbiased and robust classifier could be a complement to the human expertise of obstetricians and midwives. Moreover, the large data set of CTG curves, in particular, is a good reason for using machine learning and deep neural networks, as it is a requirement for the deep learning approaches to be feasible. Further, there are several research papers dealing with machine learning within the interpretation of CTG in general, as shown in [13, 14], and peak recognition specifically, as shown in [15, 16]. This strengthens the belief in achieving satisfactory results when using deep learning or machine learning on this type of data.

3.2 Choice of Machine Learning Algorithms

To choose the appropriate machine learning model, for the given problem, four different approaches are tested. Depending on the type of data and the initial information regarding the problem, different models are more or less appropriate. Machine learning can be divided into supervised and unsupervised learning. Supervised learning is useful when the problem is clearly described in advance and the responses to the data are known, for example examining large data sets of historical data to predict future outcomes. It develops predictive models based on both input and output data and can use both regression and classification techniques to develop the models. Examples of supervised learning techniques are Support Vector Machines, SVM, Nearest Neighbors, NN, and Decision Trees, DT. Unsupervised learning, on the other hand, is useful to identify hidden patterns in the data, which is beneficial when the problem is exploratory [17]. The unsupervised learning models groups and interprets data based solely on input data, and the most common technique is clustering [18]. Examples of unsupervised learning techniques are Neural Networks, NN, Hidden Markov Model, and Gaussian Mixture.

3.3 Support Vector Machine

Support Vector Machines (SVM) are supervised learning models that use regression or classification techniques to find hyperplanes in the N-dimensional space that distinctively can classify the data points with two separated classes [19]. Many different hyperplanes can classify the data but the goal is to choose the hyperplane that is represented by the largest separation of data points [20]. This would result in a maximization of a margin between the two classes, and a lower generalization error of the classifier. The hyperplane dimension is dependent on the number of features. A hyperplane in R^2 is just a line while a hyperplane in R^3 is a plane. Looking at the mathematical representation of the linear SVM hyperplane, it is depicted in the following way [20]:

Given a training set of n points

$$(x_1, y_1), ..., (x_n, y_n)$$

where y_i = -/+1 represents each category x_i belongs to. The objective is to find the maximum margin hyperplane which divides the group of x_i, for which y_i = -1, from x_i, for which y_i = +1, the most. The hyperplane is defined as:

$$w^T x - b = 0$$

where b is a real number and w is the normal vector to the hyperplane.

If the training data is linearly separable the best separating hyperplane is found with the following optimization problem:

Minimize $||w||$ subject to $y_i(w^T x_i - b) \geq 1$ for $i = 1, ..., n$.

This implies that the w and b which solve the problem, determine our classifier $x \mapsto sgn(w^T x - b)$, where $sgn(.)$ is the sign function. So the support vectors are the x_i satisfying $y_i(w^T x_i - b) = 1$, implying that they are on the boundary. In figure 3.1, a visual representation of support vectors in R^2 is presented.

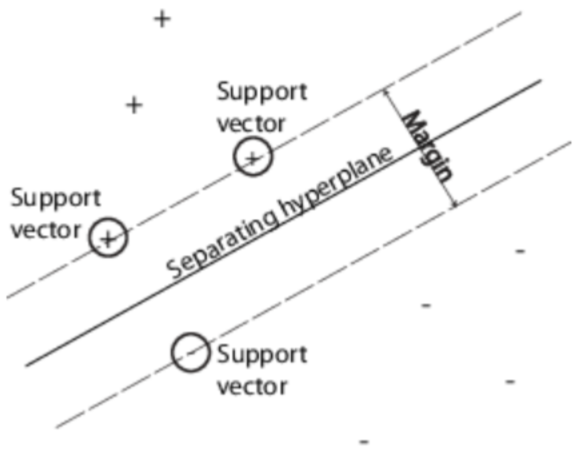


Figure 3.1: Illustration of support vectors, with + indicating data points of type +1 and - indicating data points of type -1. [20]

For a binary problem where the data has exactly two classes, SVM is an appropriate machine learning model for classifying data with potentially great accuracy.

3.4 k-Nearest Neighbors

k-Nearest Neighbors (k-NN) is a non-parametric supervised learning algorithm used for classification or regression. It is a type of distance-based learning algorithm used for heterogeneous data. Depending on if k-NN is used for classification or regression the output differs. In the first case, the output is a class membership, where the object is put in the class most similar to its k nearest neighbors. In the latter case, the output is a value, defined by the average of the values of its k nearest neighbors. It is specifically the k-NN classifier that was chosen for this project. The k-NN algorithm is one of the most widely used techniques for classifying data objects, one reason being its simplicity [21]. The main objective is to identify an object's nearest neighbors by calculating the distance between the testing and training data samples. Calculating the distance between two points, in n-dimensional space, can be done with different distance metrics, in this case, the Euclidean distance, defined by [21]:

$$d(x,y) = \sqrt{\sum_{i=1}^{n}(x_i - y_i)^2}$$

k is a user-defined constant resembling the number of nearest neighbors. The choice of k results in different clustering outcomes as it affects the smoothness of the density estimate in which the class belonging is determined [22]. A larger value of k reduces the effect of noise and makes the predictions more stable, however, it is more biased and less flexible. A smaller value of k increases the variance, but reduces the stability and the bias. In this project, k was set to one, by default. The one nearest neighbor classifier assigns the point x to the class of its nearest neighbor in the feature space [21].

As previously mentioned, the k-NN algorithm is a simple and widely used classification method, and thus a potentially good method for the given classification problem in this project.

3.5 Decision Tree

Similar to the two prior machine learning algorithms, decision trees, DTs, are supervised learning techniques used for classification or regression. Classification trees are applied when the output is a discrete set of values and regression trees when the output is a continuous set of values. The decision tree model is obtained by recursively splitting the data set, and with each partitioning fitting a prediction model [23]. This type of partitioning results in a treelike structure with root and leaf nodes. The root node of the tree is represented by the source set, and it is split into subsets, children, based on decision rules for standard CART (Classification and Regression Tree) [24]. The CART algorithm uses Gini's impurity index to base the binary splitting decision. The algorithm can be specified as follows [24]:

For all predictors $x_i, i = 1, ..., p$, the weighted impurity of node t, i_t is computed. The probability that an observation is in node t is estimated using:

$$P(T) = \sum_{j \in T} w_j$$

where w_j is the weight of observation j and T is the set of all observation indices in node t. The default for the weight w_j is $w_j = \frac{1}{n}$, where n is the sample size.

Next, x_i is sorted in ascending order, where each element of the sorted predictor is a splitting candidate or cut point. The best way to split the node t is determined by maximizing the impurity gain ΔI over all splitting candidates x_t. The observation t is split into one left and one right child node t_L and t_R. The impurity gain for the splitting candidates is determined by:

$$\Delta I = P(T)i_t - P(T_L)i_{t_L} - P(T_R)i_{t_R}$$

when x_i does not contain missing values, otherwise it is determined by:

$$\Delta I_U = P(T - T_U)i_t - P(T_L)i_{t_L} - P(T_R)i_{t_R}$$

where T_L and T_R are sets of observation indices for t_L and t_R, respectively. Finally, the predictor is split at the cut point that maximizes the impurity gain. This algorithm is then applied to the children until no further gain can be made or some stopping criterion is met. Generally, tree-based model partitions are conceptually simple whilst concurrently powerful [19]. However, there are other procedures despite CART that could be applied for decision trees. On the other hand, since CART uses a binary split, it partitions data quickly and could thus be appropriate for the large data set in this project.

3.6 Convolutional Neural Network

A convolutional neural network, CNN, is a network architecture of deep learning which uses large data sets to find information and hidden patterns in order to predict future data points. CNN is a class of artificial neural networks, ANN, which more generally is a computing system resembling the neurons in the human brain. An ANN receives and processes a signal before passing on the information to its connected neurons through a connected collection of nodes. In particular, a CNN is useful for image processing and recognizing objects, but also for the classification of non-image data such as audio and times series [25]. Further, CNN models have shown to be successful for peak detection in CTG data, and are thus a potentially useful architecture for this project [16].

A neural network has several types of layers that each have different functions to improve overall performance. Each layer learns how to detect and differentiate between features in the image. For a CNN, filters are applied to each image and the output of each convolved image is the input in the next layer. Some layers have the intent of extracting features, specific to the input data, by altering the data with different operations while other layers intend to make classifications. Below are explanations of a few frequently utilized layers for a CNN.

3.6.1 The Convolutional Layer

This layer convolves the input and passes it through a filter or kernel which reshapes the input from a tensor to a feature map. With a convolutional layer, the shape changes from (# of inputs) x (input height) x (input width) x (input channels) to (# of inputs) x (feature map height) x (feature map width) x (feature map channels). In more detail, this component performs feature extraction with a combination of linear and nonlinear operations. A kernel, a small array of numbers also known as weights, is applied across the input, tensor, and for each location of the tensor, the element-wise product between the element of the kernel and the input tensor is calculated and summed, resulting in a single value [26]. A visualization of a convolutional operation on a $5x5$ input layer with a $3x3$ kernel is shown in figure 3.2.

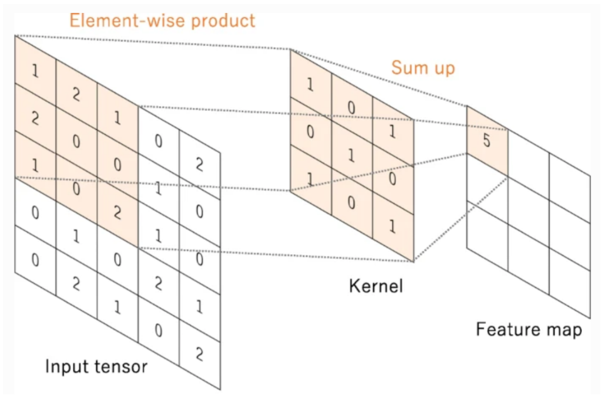


Figure 3.2: Visualization of a convolution operation with 3x3 kernel size. The element-wise product between each element of the kernel and the corresponding input tensor is calculated and summed for the corresponding output tensor in the feature map. [26]

This is a repeated procedure for an arbitrary number of kernels. The two-dimensional array of output values symbolizes different feature extractors in the feature map, and is thus the filtering of the input. The mathematical representation of convolution for a two-dimensional image I as the input and the two-dimensional kernel K is [27]:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n)$$

which may alternatively be written as follows, because of the commutative property.

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n)$$

In Matlab, the filter size can be specified as a two-dimensional vector, with height and width as input parameters. The number of filters per group is decided as a positive integer, which determines the number of channels of the output layer.

3.6.2 The ReLU Layer

The following step after a linear operation, such as a convolution, is to pass the output through a nonlinear activation function. The most common nonlinear activation function is the rectified linear unit, ReLU, perhaps because of the speed without a significant penalty of generalization accuracy [26]. The aim of using a ReLu layer is to make the model easier and faster to train, and to improve the performance [28]. In specific, it removes negative values from the feature map and replaces them with a zero value. The mathematical representation of the activation function is:

$$f(x) = max(0,x)$$

where x is the input to a neuron. This implies that input values less than zero are set to zero, and positive input values remain unchanged.

3.6.3 The Pooling Layer

A pooling layer reduces the dimensionality of the data or feature map by conducting a down-sampling operation. The operation combines the output of a neuron cluster into one single neuron, or one layer, in the next layer. There are two types of pooling operations: max pooling and average pooling. The former operation extracts clusters from the feature map and uses the maximum value as output, while average pooling takes the average value of all elements in the feature map as output [26].

The dimensions of the neuron clusters can be determined manually by the pool size in Matlab. Pool size is a two-dimensional array, with two positive integer values. Further, the step size for traversing the feature map, horizontally and vertically, can be decided with the stride. If the stride is set to less than the pooling dimensions, the pooling regions have an overlap [29].

After feature learning has been completed, often with several different connected layers, the next step for CNN is classification. Classification often follows the given structure of a fully connected layer, a softmax layer, and finally an output layer.

3.6.4 The Fully Connected Layer

A fully connected layer has many advantages as it makes no assumptions about the input, and can thus be used in many applications. The layer connects every neuron in one layer with every neuron in another layer, implying that every input in one layer is connected to every output in another [26]. Specifically in classification problems, the input for the fully connected layer is the output from the last pooling or convolutional layer, which is then used to classify the image. Accordingly, the output size of the fully connected layer is equal to the number of classes in the data set, which is specified manually in Matlab [30]. However, for a regression problem, the output size is equal to the number of response variables. Mathematically, a fully connected layer is a function from R^m to R^n. Let $x \in R^m$ represent the input and $y_i \in R$ be the i:th output from the fully connected layer. Then y_i can be written as follows:

$$y_i = \sigma(w_1 x_1 + ... + w_m x_m)$$

where σ is a nonlinear activation function and w_i are the learnable parameters or weights assigned to each neuron in the layer [31]. In Matlab, the fully connected layer also adds a bias vector besides multiplying the input by a weight matrix. It can be represented as follows:

$$Y = WX_t + b$$

where Y is the output array, W is the weight matrix, X_t is the input at time step t, and b is the bias vector [30].

3.6.5 The Softmax Layer and Classification Layer

In the final layer of a neural network classifier, a softmax layer is often applied. The softmax layer transfers a neuron value to a value between zero and one, which can be used for classification [16]. It uses a transfer function for each class using a normalized exponential function [32].

$$y_r(x) = \frac{exp(a_r(x))}{\sum_{j=1}^{k} exp(a_j(x))}$$

where $0 \leq y_r \leq 1$ and $\sum_{j=1}^{k} = 1$, $a_r(x)$ is the r:th class element from the prior output vector, and $y_r(x)$ is the probability that the network is associated with the inputs in the respective class.

For a classification network, the softmax layer is often followed by a classification layer. The classification layer takes the output from the softmax layer as input, and assigns it into K mutually exclusive classes by computing the cross-entropy loss [33]:

$$loss = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K} w_i t_{n_i} ln(y_{n_i})$$

where N is the number of samples, K is the number of classes, w_i is the weight for class i, t_{n_i} is the indicator that the n:th sample belongs to the i:th class, and y_{n_i} is the output for sample n for class i, implying the value from the softmax function. In the binary case, there are two classes which the samples can be assigned to.

3.6.6 Hyperparameters and Training Options

Along with deciding which layers will best translate into high classification accuracy of the features in the data, one important step is to choose the optimal hyperparameters and training options. There is no given best setup of hyperparameters, instead, it is a trial and error process [34]. Different data sets require different combinations of hyperparameters and training options which are configured manually. These options can be optimized by freezing the features and/or network architecture and then optimizing one hyperparameter and/or training option at a time. There are two types of hyperparameters; one determines how the network is structured and the other determines how it is trained [34]. Below are short explanations of the most frequently used hyperparameters and training options in this project.

Hyperparameters that determine how the network is structured:

1. **Kernel size** determines the size of the filter used in the convolutional layer, and it is specified by a two-dimensional vector of integers. If the kernel size is set to 1x1 this results in a feature map with the same size as the input, since the filter only has a single weight. On the contrary, if the kernel size is set to the same size as the input there would be one weight for each pixel and the output in the feature map would be one single pixel. Generally, a small kernel size is preferred over a larger one as it reduces the computational cost and thus requires far less training time. Moreover, an even-numbered kernel size, for example, 2x2 or 4x4, is not generally preferred as it does not symmetrically divide the previous layer around the output pixel. This could lead to distortion across the layers when symmetry is not present [35]. Therefore, odd-numbered kernels are most commonly used.

2. **Stride** is the rate at which the kernel passes through the input image. It is specified as a vector of two positive integer values, representing the vertical and horizontal step size, respectively. The choice of stride has an effect on the resulting feature map as it affects how the filter is applied to the input image. A higher value, for example [2 2], would imply that the filter would move 2 pixels right in every horizontal movement and 2 pixels down in every vertical movement. Accordingly, the output feature map is downsampled and the computational cost is reduced.[36]

3. **Padding** is a tool to ensure that the kernel passes over the edge of the input image by adding layers of zeros. By default, the kernel starts in the top left corner of the input image with the filter set on the top-left pixel. The filter passes through the image from left to right until the right side of the filter is sitting on the far right pixel. Therefore, the pixels laying on the edge have no opportunity to be in the center of the filter and thus have less interaction with the filter. To alleviate this problem, padding is used, implying that a border of zeros is added to the edge pixels and thus artificially putting

the edge pixel in the middle. This has no effect on the element-wise product in the feature map, however, it gives the border of the image more opportunity for feature detection [36].

Hyperparameters that determine how the network is trained [34]:

1. **Mini batch size** is a subset of the training data set which is specified as a positive integer value. It defines the number of samples that the algorithm works through before updating the internal model parameters, which is done by comparing the expected output variables with the predictions and calculating an error. If the batch size is equal to the size of the training set, implying only one batch and thus the number of iterations and epoch size is equivalent, the algorithm is called batch gradient descent [37, 38]. On the opposite, if the batch size is equal to one, the internal model parameters are updated after each sample and the learning algorithm is called stochastic gradient descent [37, 38]. The former algorithm can be used on smoother, convex curves and it has a better chance of converging directly to the global minimum. However, the approach is computationally expensive since much data is kept in the memory. Therefore, it is not an optimal way of training large networks. The latter algorithm is a good option for larger data sets, as it updates the parameters more frequently and thus results in faster convergence. However, the cost function is not reaching the actual minimum but instead oscillating around it. The third option of batch size, which combines the advantages of batch gradient descent and stochastic gradient descent, is mini-batch gradient descent [37, 38]. It uses a fixed number of training samples, $1 < batchsize < n$ where n equals the full sample size. This option has a good chance of converging to a value close to the minimum without the high computational expense.

2. **Number of epochs** is a positive integer value that specifies the maximum number of epochs used for training. More specifically, the number of epochs defines the number of times the algorithm passes through the entire training data set, and consequently for how long the network is trained [38]. One epoch implies that each sample in the training data set is iterated through once, and thus the internal model parameters are updated one time. The number of epochs is often set to a large integer value, often a few hundred or thousands, as it allows the algorithm to update the model parameters multiple times, and hence minimize the model error sufficiently.

3. **Learning rate** is a positive scalar value that determines the step size at each iteration [39]. Accordingly, it controls how fast the model adapts to the problem. A lower rate requires a larger number of training epochs, since the weights have smaller changes in each update, whereas a larger rate requires fewer training epochs as there are more rapid changes to the weights. Decreasing the learning rate might be a good option as it makes it easier for the network to find the local minimum. However, this results in a slower network as it requires more training epochs for satisfactory outcomes.

3.7 Prevent Overfitting

Overfitting is a common problem in machine learning algorithms and it results in unsatisfactory errors when transferring from training data to test data. The frequent problem arises when the model tries to fit the data completely and remembers the given structure, and assumes generalizations that do not exist. Overfitting could be decomposed into bias and variance, two different generalization errors [40]. The former describes the model's tendency to constantly learn the same thing, and it measures the difference between targeted and predicted values. More bias would imply an oversimplified model. The latter describes the tendency to learn random things that are not present in the data, and thus measures the inconsistency of different predictors. More variance could be an indication of an overfitted model, where the model has lost its capability to generalize [41].

To detect models with overfitting properties the data needs to be tested. There are different techniques to overcome the problem of overfitting [41]. Below are a few techniques briefly explained.

1. **More data**
   One option to mitigate the risk of overfitting is to increase the training data set. With more data, it is easier for the model to recognize attributes between the input and output variable, and it is therefore less prone to make assumptions that cannot be generalized. However, it requires the data to be clean. An increase in the data set does not mean adding more noisy data as this would counteract the benefits of additional data.

2. **Augmenting data**
   Data augmentation is an alternative technique to increase the training data set. In cases when gathering more data is no option, augmentation of data is an appropriate method to artificially increase the amount of training data. The technique changes the sample data slightly in each iteration of model processing. So even though the given data set is constrained to the initial size, the model can still train on a larger data set and thus reduce the risk of overfitting.

3. **Addition of noise to the input data**
   Similar to data augmentation is the option of adding noise to the input and output data. The goal is to increase the stability by adding noise to the input data without affecting the quality and to increase the diversity by adding noise to the output data. However, the technique should be applied with caution as adding noise should not change the essence of the data or reduce its quality.

4. **Simplify data**
   This technique aims to simplify the model in order to decrease the complexity and thus reduce the risk of overfitting. There are several procedures to fulfill the aim, including parameter reduction, adding dropout on a neural network, and pruning a decision tree.

5. **Feature selection**
   The more complex a model is, the higher the risk of overfitting the model. Accordingly, reducing the number of features is one way of eliminating unnecessary complexity. One model has several features and parameters depending on the network architecture. Some of these features may be redundant and irrelevant, and should thus be removed. There are different feature selection heuristics that may be used to identify features for removal.

6. **Regularization**
   Regularization is one technique to artificially force the model to be simpler, and thus reduce the risk of overfitting because of model complexity. There are different methods for regularization. For example, adding a penalty term on the cost function (L1 & L2 regularization) to limit the model's variance.

7. **Ensembling**
   This technique produces an optimal predictive model by combining several base models. The two most common ensemble methods are bagging and boosting, which have two different approaches. The former attempts to reduce the risk of overfitting complex models by combining unconstrained learners in order to average their predictions.

Boosting attempts to improve the predictive flexibility of simple models. The method combines several constrained learners, where each model is trained in sequence into one unconstrained learner and thus boosts the aggregated complexity of the final model.

8. **Adding dropout layers**
Adding dropout layers is a type of regularization as it simplifies the model artificially by ignoring a subset of units in the network structure with a given probability. When reducing the complexity of the model, by probabilistically dropping out nodes, the risk of overfitting is mitigated or prevented, and it thus increases the stability and accuracy of the machine learning model.

9. **Early stopping**
Early stopping is a method where the aim is to stop the training when the validation loss starts to degrade and the model starts to memorize noise and randomness in the data. It is a balance between not stopping the model too soon, as this would lead to underfitting, and stopping it too late, leading to overfitting. The optimal model can be found by monitoring the loss graph or setting an early stop trigger.

10. **K-fold cross-validation**
This is one of the more popular techniques to evaluate machine learning models and detect overfitting. It is a robust resampling procedure that uses the initial data set to generate k train splits. The procedure partitions the data in k folds, then the algorithm iteratively trains on k-1 folds while using the remaining fold as a test set. The neural network can train and tune hyperparameters and then test them on completely unseen data. This procedure is repeated until every individual k-fold has been used as a training set.

3.8 Evaluate Models

In order to assess the results of the different models and machine learning algorithms, the following statistical measures are used: *Sensitivity*, *Specificity*, *Accuracy*, and *F1 score*. The output of each of the conducted tests is made binary, meaning that there is a positive or a negative result. Hence, the following four outcomes are possible:

- True positive (TP)
- True negative (TN)
- False positive (FP)
- False negative (FN)

The first two are correctly identified, either positively or negatively, whereas the last two are identified wrong, either positively or negatively. The binary values are used to calculate the statistical measures. Sensitivity is the true positive rate (TPR) implying the ability to correctly classify true cases, i.e., all true positives divided by the total number of positive cases (P). Specificity, on the other hand, is the true negative rate (TNR), which represents the ability to correctly classify negative cases. Both measures can be represented as the difference between the false negative rate and false positive rate. Their mathematical representation is as follows:

$$TPR = \frac{TP}{P} = \frac{TP}{TP + FN} = 1 - FNR$$

$$TNR = \frac{TN}{N} = \frac{TN}{TN + FP} = 1 - FPR$$

There is often a trade-off between sensitivity and specificity, meaning that a higher sensitivity could lead to lower specificity, and vice versa. The overall performance can be measured with accuracy, which is equal to the fraction of correct classifications. The F1 score is another measure of a models accuracy, and it is the harmonic mean of the precision and recall. Precision is the fraction of true positives among all of the samples that the model classified as positive, i.e., the number of true positives divided by the false positives plus the true positives. The recall is equivalent to the sensitivity. A perfect model has an F1 score of one. The mathematical representations of the accuracy and F1 score are presented below:

$$ACC = \frac{TP + TN}{P + N} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$F_1 = \frac{2}{\frac{1}{TPR} \cdot \frac{1}{PPV}} = 2 \cdot \frac{PPV \cdot TPR}{PPV + TPR} = \frac{2 \cdot TP}{2 \cdot TP + FP + FN}$$

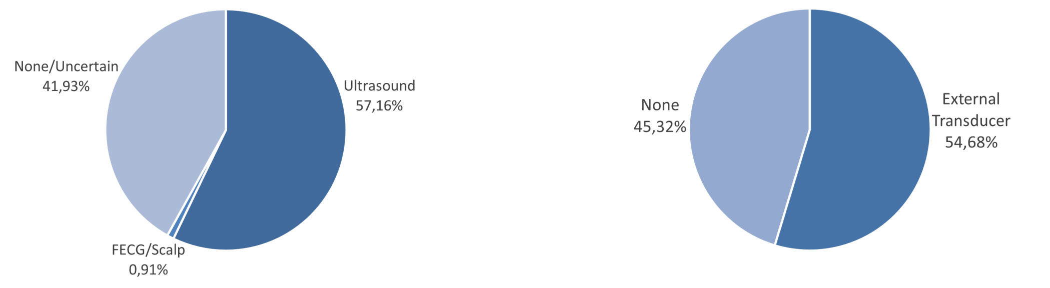
4. Data Collection

In this chapter, the data set is introduced and explained, both regarding the CTG files and the corresponding clinical data. To make the data set appropriate for further analysis, there are several pre-processing steps that need to be conducted. The data have to be cleaned from irrelevant, noisy and empty files. Further, the data need to be segmented with regards to time interruptions and discontinuances in the TOCO curve, in order to be relevant in this thesis. In the following sections, the pre-processing and segmentation aspects are presented in more detail.

4.1 Data set

The obtained data set contains 98 328 childbirths, including clinical values as well as digitized CTG curves for each individual labor. The linked clinical data and CTG data were obtained from deliveries between May 2001 and December 2013. The quality of the CTG curves is varying, and most of the files contain interruptions in the signals, due to monitoring not being conducted continuously. The lengths of the files also vary, from 1 to 769 720 data points, which is equivalent to more than 53 hours of data. The time-difference is mostly depending on the duration of the delivery, but also to which extent the labor has been monitored with CTG-equipment.

The CTG data contain the patient ID, delivery date, a time vector with 4 values per 1 second, a FHR-vector, a TOCO-vector, information regarding the FHR sensor type, TOCO sensor type, and signal quality. Among the total 98 328 samples, 56 209 samples have the FHR measured using ultrasound and 892 using an electrode on the fetal scalp. The remaining 41 227 samples are either non-connected or uncertain. For the TOCO data, 53 764 samples are measured with an external TOCO transducer placed on the mothers' abdomen, and the remaining 44 564 samples have no transducer connected. The distribution of sensor types are presented in figure 4.1.

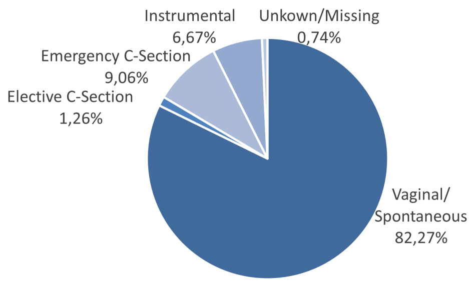


(a) Distribution of sensor type for FHR. **(b)** Distribution of sensor type for TOCO.

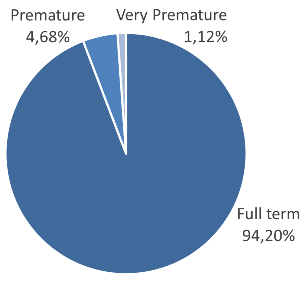
Figure 4.1: Distribution of sensor types.

The clinical data hold information regarding the mother, the birth, and the status of the newborn baby. The data contain 19 clinical values per id. Regarding the mother, there is information about smoking habits and whether she is a type one diabetic or not. There is also information regarding parity, implying how many previous times the mother has given birth to a child that is more than 24 weeks old. Further, the clinical data contain information regarding if the mother has suffered from placental abruption, ablatio placentae, which means that the placenta separates from the inner wall of the uterus before birth. This condition is dangerous for both mother and fetus, as it causes the risk of the fetus being deprived of oxygen and nutrients. There is also a major risk of heavy hemorrhaging for the mother [42]. Preeclampsia is another dangerous condition that is listed in the data set. Preeclampsia is characterized by high blood pressure and signs of damage to the organs, in most cases the kidneys and liver [43]. Both of these conditions can cause early delivery.

Moreover, the data contain information regarding the delivery type. There are ten different delivery types that can be categorized into four groups; vaginal birth, elective cesarean section, emergency cesarean section, and instrumental vaginal delivery. Vaginal birth is the most common way of delivery. Approximately 80 percent of the births in the data set are vaginal. The elective cesarean sections constitute just over one percent of the data set, while the emergency or urgent cesarean sections count for approximately nine percent. Instrumental vaginal deliveries are conducted with forceps or a ventouse suction cup. The two types of instrumental vaginal deliveries stand for around six percent of births in the data set. In some cases the birth type is unknown or missing, this category is shy of one percent. The distribution between the different delivery methods in the data set is visualized in figure 4.2a.



(a) Distribution of delivery type.



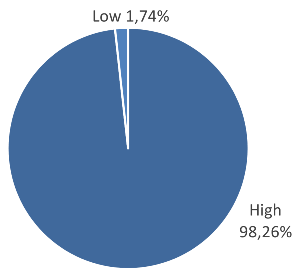
(b) Distribution of gestational age.

Figure 4.2: Distribution of delivery type and gestational age in the full data set.

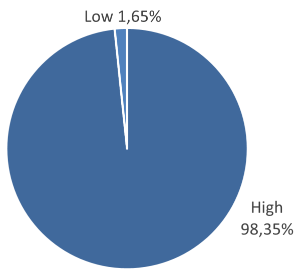
The gestational age is also included in the data set. A normal, full-term, pregnancy is considered to last for about 40 weeks (280 days). A baby born before week 37 (259 days) is classified as premature, and if it is born before week 32 (224 days) it is classified as very premature. Almost five percent of the babies in the data set is premature and around one percent is very premature. The remaining 94 percent of the pregnancies are considered full-term. The distribution of the gestational age in the data set is visualized in figure 4.2b.

The sex, weight, and weight deviation of the baby are also included in the data. The weight deviation depends on the length of the pregnancy as well as the sex of the child. Moreover, the vital status is included. The vital status is divided into four categories; not dead, dead before delivery, dead during delivery, and dead after delivery. The first category contains more than 99.8 percent of the group in total. Further, the Apgar scores at one and five minutes are available in the data. At the one-minute Apgar, scores below four are considered bad, and scores below seven are considered bad at the five-minute Apgar. Approximately 98 percent of the scores are above the threshold values at one and five minutes, respectively. This is visualized in figure 4.3. The data set also contains information regarding the pH value. It is taken from the baby's umbilical cord and should be at least 7.15.

The pH value was measured for approximately 65 percent of the samples in the data set. Among them, 83 percent, or 54 456 babies, have a pH value of 7.15 or higher. The data set also contains information regarding neonatal care, cerebral hemorrhage, neurological impact, spasms, and malformations.



(a) Distribution of Apgar score after 1 minute.



(b) Distribution of Apgar score after 5 minutes.

Figure 4.3: Distribution of Apgar scores in the full data set.

All information can be seen in a more compressed form in Appendix A.

4.2 Segmentation

The CTG data are of varying quality. Some files contain large amounts of information, while others are almost empty. The majority of the files also contain interruptions, since the standard procedure is not to continuously measure the CTG but rather to measure approximately 30-60 minutes intermittently. Clinicians state that a continuous CTG curve of a minimum of 20 to 30 minutes is needed to assess the current status. Hence, a large number of the data in the CTG files are not usable for analysis. Figure 4.4 displays several CTG files of various quality, where the bottom two are the only ones containing useful data.

To find relevant data for this thesis, the CTG files were investigated with the following aspects in consideration. The maximum allowed time jump was set to five seconds. The set limit is a compromise between not excluding too many files and not allowing long interruptions that might cause problems in the analysis. Various limits were tested before establishing the threshold at five seconds. The files that contained less than 20 minutes of continuous data were excluded from the set, since the clinical standard is to assess at least 20 minutes of continuous data. Files that contained one or more segments of 20 minutes of continuous data were temporarily saved. The segments that contained less than 50 percent of the TOCO values were considered unusable and were therefore removed. Some of the segments had missing values in the TOCO curve, and for other files, the TOCO curve was equal to zero for large consecutive parts. For both these cases, the assumption is that something has occurred with the measuring equipment which has caused an interruption or faulty data, and therefore, the segment can no longer be considered continuous. The limit on how many consecutive zeros were allowed was, once again, a compromise between not excluding too much data and not including data that might be inaccurately retrieved. The threshold was set to 120 consecutive zeros, equivalent to 30 seconds. The segments that contained interruptions were split into new segments, not including the interruption, where the segments that were at least 20 minutes long were saved and used during the project. The indices for these segments are labeled as ‘good segments’ further on in this thesis. In total there were 205 439 ’good segments’ meeting the set requirements.

Furthermore, by using the indices from the 'good segments' new segments were extracted from the 20 000 first CTG files. These segments have the same structure as a measure, with a time vector with 4 values per 1 second, a FHR-vector, a TOCO-vector, information regarding the FHR sensor type, TOCO sensor type, and signal quality. These segments are subsets of each mat file. In total, 52 344 segments were extracted from the first 20 000 CTG files. Some CTG files had one segment meeting the requirements of length and continuity, while others had none or several. These segments were used as foundations for the data sets in further analysis.

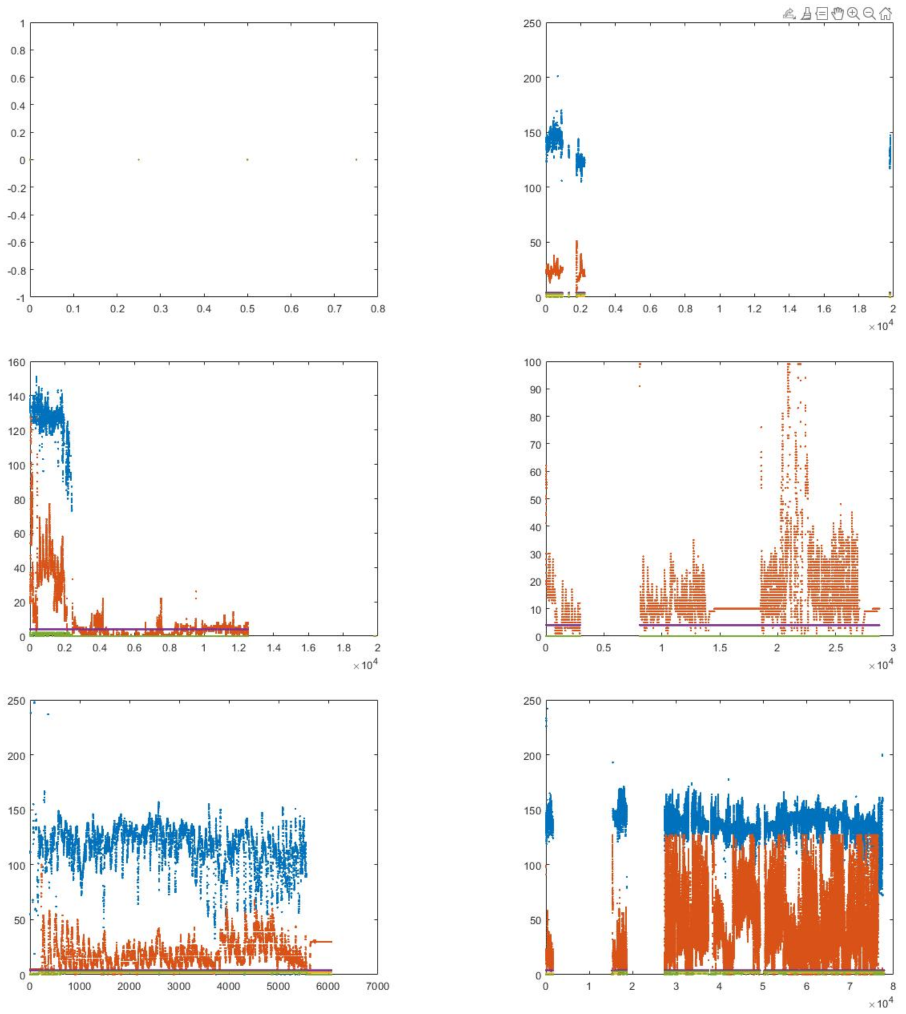


Figure 4.4: Various CTG files. The first four do not contain any usable segments, whereas the bottom two are useful files.

5. Peak Detection

In this chapter, the aim is to detect uterine contractions in the TOCO curve, which later is used to determine the contraction frequency. The uterine contractions are seen as peaks in the TOCO curve, and the objective is to create a peak detection algorithm that detects and indicates the contractions. The chapter contains a brief explanation of the use and application of Gaussian blur, how the peak detection algorithm was initialized, the segmentation of the data, the method, and finally, results from the machine learning models.

5.1 Gaussian Blur

When processing and analyzing signals, there are several techniques to detect different components of the given signal. If the aim is to identify peaks in a measured signal, one approach is to use a Gaussian filter. The filter is useful when attenuating high-frequency noise, and it is thus a low-pass filter [44]. More specifically, convolving an image with a Gaussian filter is a commonly used method that decreases sharpness to reduce noise and details in the image. Hence, it is appropriate for edge detection. This effect is often referred to as Gaussian blur or Gaussian smoothing. The filter uses a Gaussian function to blur the image by calculating the transformation and applying it to each pixel in the image. The procedure builds on applying a Gaussian filter to the signal which convolves the two functions $(f * g)$, where g is the Gaussian filter, the kernel, and f is the measured signal, expressed as an integral transform:

$$(f * g)(t) := \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau$$

The convolution expresses the modified shape when a filter is applied to the initial signal. The formula for the Gaussian function is defined as:

$$g(t;\sigma) \equiv \frac{1}{\sqrt{2\pi}\sigma}e^{-t^2/2\sigma^2}$$

where σ is the parameter denoting the Gaussian half-width [44]. In this project, the Gaussian smoothing function is applied to the TOCO curve where the convolution is used for peak detection. The parameter σ is manually set to 20 and t is a vector between $(-3 * \sigma : 3 * \sigma)$.

5.2 Peak Detection

Detecting peaks is an important step in signal processing, and it can be conducted with several different approaches including artificial neural networks, filtering, and transformation [45]. As an initial step in detecting peaks in the TOCO curve, the signal was concatenated as one continuous signal before being processed using the Gaussian smoothing filter. After convolving the signal using the low-pass filter, the peaks were determined using the following algorithm.

Assume y is the convolution $(f * g)$. Potential peaks are defined as:

$$\begin{aligned} & y_{m-1} < y_m > y_{m+1} \\ \wedge \;\; l < m \;\; \wedge \;\; y_{l-1} > y_l < y_{l+1} \;\; \wedge \;\; & y_j < y_{j+1} j \in \{l, ..., m-1\} \\ \wedge \;\; r > m \;\; \wedge \;\; y_{r-1} > y_r < y_{r+1} \;\; \wedge \;\; & y_j > y_{j+1} j \in \{m, ..., r-1\} \end{aligned}$$

with $p = \{l, m, r\}$ defining the indices for the peak, where the triplet encapsulates the local maximum by its nearest minimum in both directions.

5.3 t-distributed Stochastic Neighbor Embedding

To get a visual interpretation of high-dimensional data, one commonly used statistical method is the t-distributed stochastic neighbor embedding, t-SNE. It is a method that converts the high-dimensional data set into two- or three-dimensional data which can be displayed and visualized in a scatter plot. The method aims to preserve as much of the structure of the high-dimensional object as possible, but reduces it to a two- or three-dimensional point. Similar objects are modeled as nearby points, and dissimilar objects are modeled as far distant points, visualized as clusters [46]. Accordingly, t-SNE is an appropriate method to analyze whether the data set and its binary division have a significantly different structure, and thus if the binary outcome of the data set has a high probability of being predicted by the machine learning algorithm.

The t-SNE algorithm consists of several stages. First, the Euclidean distance of each point to all other points is calculated. These distances are transformed into conditional probabilities [47], which explains the probability that x_i would have x_j as a neighbor given being picked proportionally to their probability density under a Gaussian centered at x_i [46]. The Mathematical representation is as follows:

for $i \neq j$:

$$p_{j|i} = \frac{exp(-||x_i - x||^2/2\sigma_i^2)}{\sum_{k \neq i} exp(-||x_i - x||^2/2\sigma_i^2)}$$

for $i = i$:

$$p_{i|i} = 0$$

where σ_i is the variance of the Gaussian centered on data point x_i [47]. If the conditional probability $p_{j|i}$ is relatively high it represents nearby data points and for distant data points, the probability is almost infinitesimal [46]. Using the conditional distributions, $p_{j|i}$, the joint probability distribution is calculated using:

$$p_{ij} = \frac{p_{j|i} + p_{i|j}}{2N}$$

where $p_{ji} = p_{ij}$, $p_{ii} = 0$ and $\sum_{i,j} p_{ij} = 1$.

The second stage is to create a data set of points in a low-dimensional space. A random data set of N points and K features or target dimensions is created. A joint probability distribution using the t-distribution is created as:

$$q_{ij} = \frac{(1 + ||y_i - y_j||^2)^{-1}}{\sum_k \sum_{l \neq k}(1 + ||y_k - y_l||^2)^{-1}}$$

where q is the probabilities and y is the data points [47].

When using the t-distribution rather than the Gaussian distribution the crowding of points in lower dimensional space is prevented since there is a heavy tail property of the t-distribution. This property implies that moderate distances in the high dimensional space become extreme in the low dimensional space and thus dissimilar objects are modeled separately in the scatter plot, implying that crowding is prevented [47].

The third stage is minimizing the Kullback-Leiber divergence of the distribution P from the distribution Q using gradient descent according to:

$$KL(P||Q) = \sum_{i \neq j} p_{ij} log \frac{p_{ij}}{q_{ij}}$$

where P and Q are the joint probability distribution from the high-dimensional space to the low-dimensional space, respectively [47].

The Kullback-Leiber divergence measures the difference between two distributions, and a smaller KL divergence implies that the distributions are similar, and where zero means identical. Accordingly, the minimization of the KL divergence resonates with the similarities between the high-dimensional objects.

5.4 Peak Segmentation

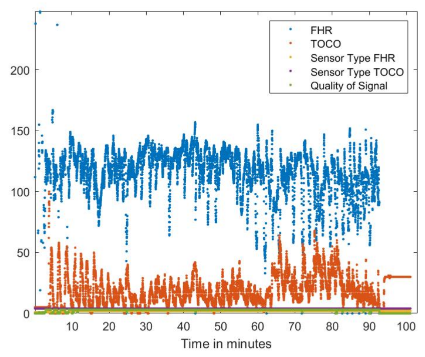
Using the peak detection algorithm on the 'good segments', potential peaks were identified. In figures 5.1 and 5.2, the original CTG measurement, and the TOCO curve before and after peak detection are shown for id '1' and '26', respectively.

To classify a peak as an uterine contraction, some conditions needed to be met. This was to increase the likelihood of correctly assessing the peaks or artifacts depending on clinical conditions. The conditions are presented in table 5.1.

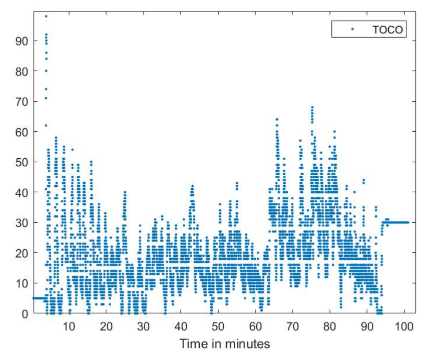
Table 5.1: Conditions for Assessment of Uterine Contraction

Condition	Value
Duration	> 50 seconds
Difference baseline to peak	> 10 points
Difference baseline values	$< abs(10)$ points

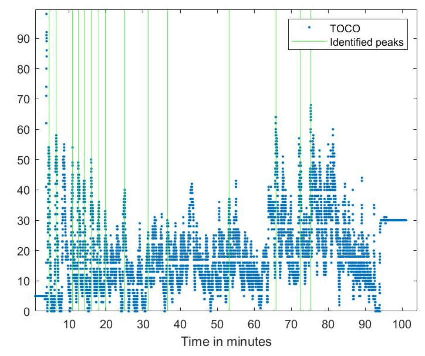
The first condition was the duration of the peak. To be assessed as a contraction, the duration should be around 60-180 seconds. However, to not sort out contractions being slightly less than 60 seconds, the threshold value was set to 50 seconds. There was no need to set an upper limit, as this did not change the result of the peak detection algorithm. In figure 5.3, a visualization of peaks violating the first condition is displayed.



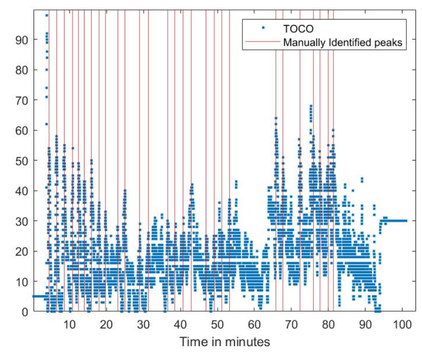
(a) CTG curve. X-axis is time in minutes. Y-axis is beats per minute for FHR signal



(b) TOCO curve. X-axis is time in minutes.



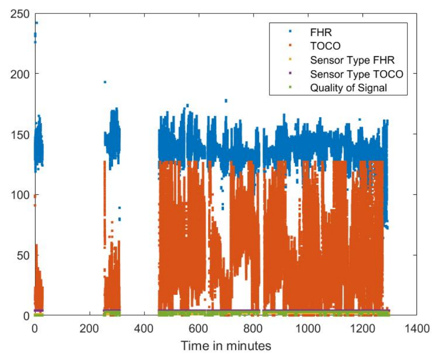
(c) TOCO curve with green lines for detected peaks using Gaussian filter. In total 15 identified accurate peaks.



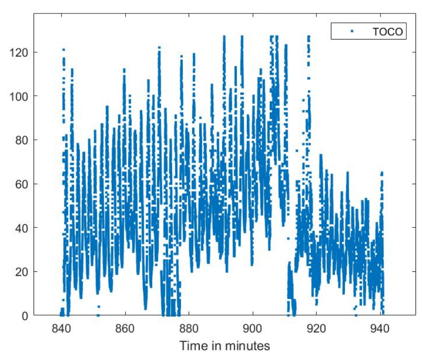
(d) TOCO curve with green lines for manually detected peaks. In total 28 identified accurate peaks.

Figure 5.1: Identification of peaks on mat file with id=1

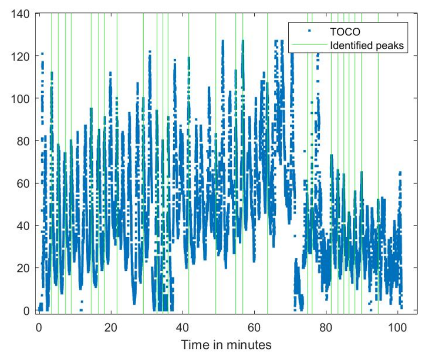
The second condition relates to the increase in pressure from the baseline. A contraction must, as expected, have an increase in pressure value. The threshold value was set to 10 points between the baseline and the highest value to sort out the peaks that were only noise or artifacts. The last condition for contraction assessment was regarding the retrieval of the baseline. After a contraction, the pressure should return to its previous baseline value. There were cases where the peak detection algorithm detects peaks that were artifacts, for example tightening of the external TOCO transducer. When tightened, the pressure increases, which would indicate a peak, but since the baseline does not return to its prior value, it can be sorted out as an artifact. Moreover, when concatenating the TOCO curve as a continuous signal, there was a risk that the peak detection algorithm could identify peaks between two different segments in time. When concatenated, the difference in baselines could be interpreted as a local maximum, and thus a peak. But since it did not return to the prior baseline value, it should be sorted out as an artifact in the data. However, the TOCO curve does not always return to the exact same pressure value as before the contraction, hence the condition was not set as strict. Instead, the difference between the start and stop values of the peak, and thus the baseline values, should be less than 10 points. A visualization of peaks violating the conditions is displayed in figure 5.4. All conditions had to be met to assess the peak as an uterine contraction. In figure 5.5, two different types of peaks complying with the conditions are visualized.



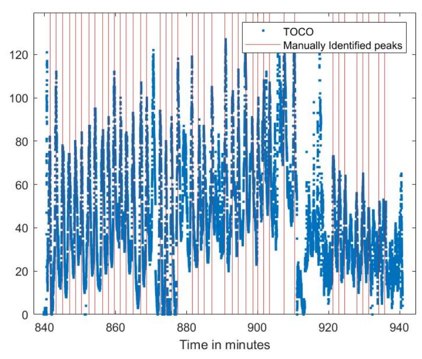
(a) CTG curve. X-axis is time in minutes. Y-axis is beats per minute for FHR signal



(b) TOCO curve. X-axis is time in minutes.

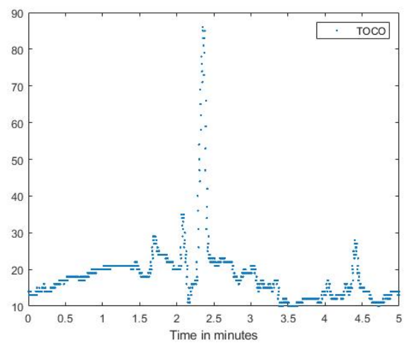


(c) TOCO curve with green lines for detected peaks using Gaussian filter. In total 26 identified accurate peaks.

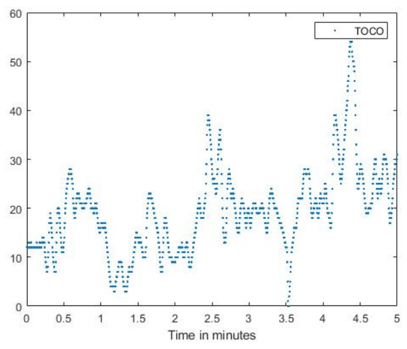


(d) TOCO curve with green lines for manually detected peaks. In total 42 identified accurate peaks.

Figure 5.2: Identification of peaks on the fifth segment for file with id=26

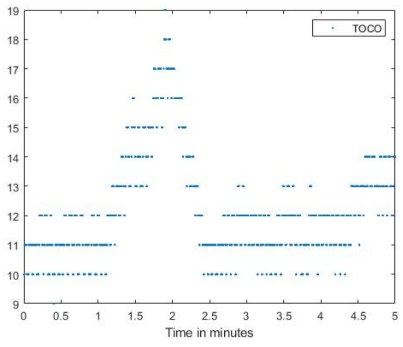


(a) Condition 1 violated: Example of a local maximum violating the condition of >50seconds.

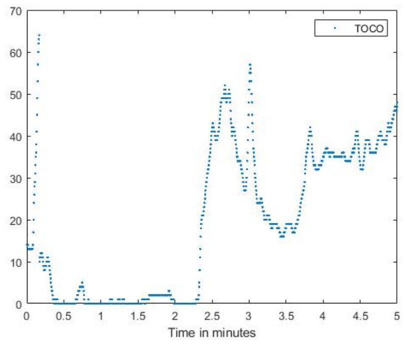


(b) Condition 1 & 2 violated: Example of noisy TOCO signal violating the conditions of duration > 50 seconds and peak > 10 points.

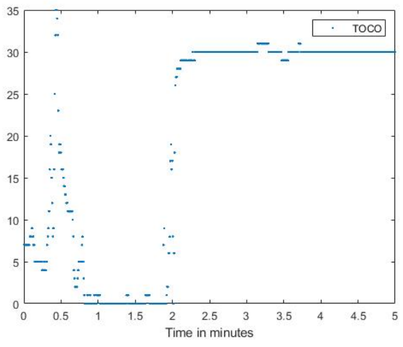
Figure 5.3: Examples of peaks violating the clinical conditions.



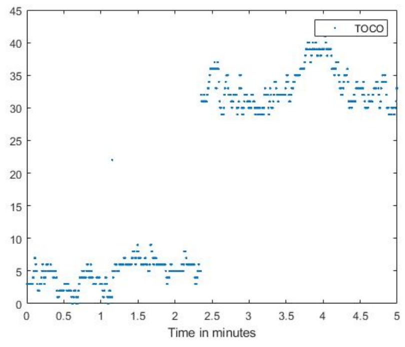
(a) Condition 2 violated: Example of a local maximum violating the condition of peak >10 points.



(b) Condition 3 violated: Example of a local maximum violating the condition of peak not returning to baseline.

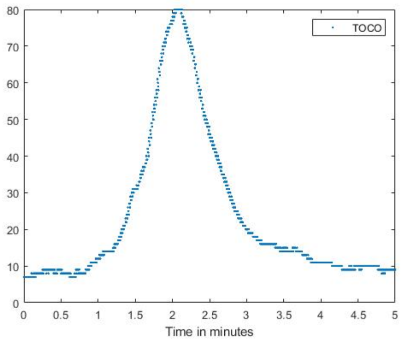


(c) Condition 3 violated: Example of artifacts, where a local maximum is incorrectly indicated.

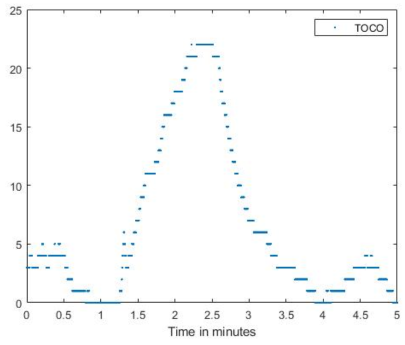


(d) Condition 3 violated: Example of the concatenating TOCO curve, where a local maximum is incorrectly indicated in-between segments.

Figure 5.4: Examples of peaks violating the clinical conditions.



(a) Example of a clear local maximum meeting all requirements of being a uterine contraction. Increase of 70 pressure points.



(b) Example of a clear local maximum meeting all requirements of being a uterine contraction. Increase of 20 pressure points.

Figure 5.5: Examples of peaks meeting the clinical conditions.

In order to extract short segments containing visually clear peaks, the threshold for assessing a peak as accurate was set restrictively. This resulted in a large number of inaccurate peaks and only a small group of accurate ones, where the latter was saved for further use. The segments that should not contain a peak were extracted in two different ways. The first way was by setting the threshold for accurate peaks very liberally. This caused a large number of peaks to be labeled as accurate, and only the poorest ones were labeled as 'non-peaks'. These were saved for further use. The second way was by concatenating all segments between the identified peaks, accurate, and inaccurate, from a 'good segment'. The new segments were cut into shorter segments on which the peak detection algorithm was used. All segments that contained a peak according to the peak detection algorithm were excluded, and the ones that did not contain a peak were saved. All used segments were five minutes long, which is equivalent to 1200 data points.

The above method resulted in a large and fairly good set of peaks and non-peaks. However, to be able to fully evaluate the machine learning algorithms, a better data set was needed, where it was guaranteed that every peak and non-peak were labeled correctly. This set was created manually from the first set, and consisted of 1000 peaks and 1000 non-peak windows. This data set was visualized using t-SNE, to test if the data set of peaks and non-peaks had a significantly different structure. The whole data set was plotted in a scatter plot with the two-dimensional interpretation of the high-dimensional space, displayed in figure 5.6.

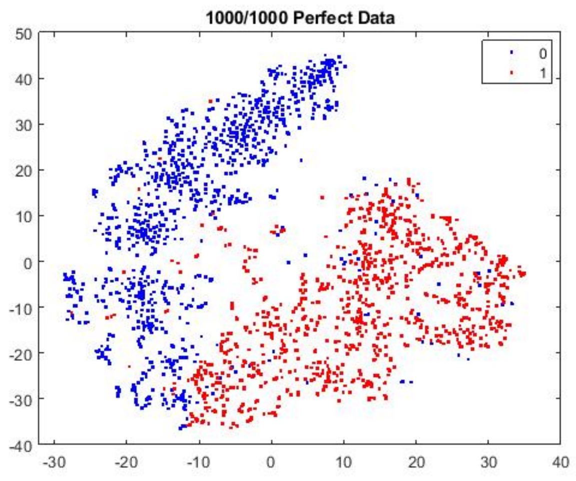


Figure 5.6: Two-dimensional interpretation of 1000 peaks and 1000 non-peaks, binary representation is 1 and 0 respectively.

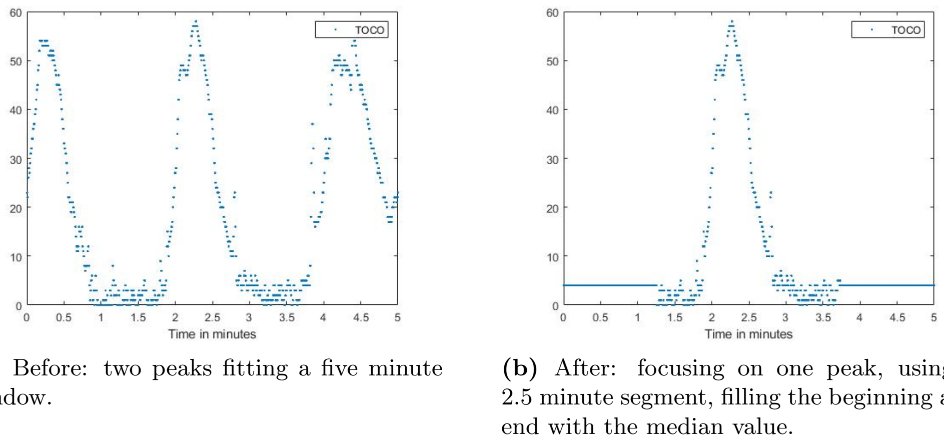
Inspecting the scatter plot closely, it is clear that there is a distinct difference between peaks and non-peaks when converting the 1200x1 dimensions to 2x1 using t-SNE. Consequently, the two distinct clusters of peaks and non-peaks showcase that the TOCO curve for peaks and non-peaks have significantly different structures, and it should thus be possible to make distinctions using machine learning.

5.5 Method

Six different machine learning algorithms were used to identify if a window contained a peak or not. All algorithms were used with 10-fold cross-validation to ensure that the result was not haphazardly dependent on the data set. The algorithms were first used on single five-minute windows, and then a sliding window algorithm was implemented to find all peaks in an actual CTG file.

The sliding window was initially set to evaluate every possible 5-minute segment in each file. However, setting the slide to only one step, i.e., with an almost full overlap, resulted in a slow and unsatisfying algorithm, where the same peak was identified numerous times. Since a peak often lasts between one and two minutes, i.e., between 240 to 480 data points, one peak could easily generate 500 positive outcomes. This problem was handled by only saving unique indices for the maximum value of each peak. However, the algorithm was not time efficient and it produced large amounts of unnecessary data. Different settings of the slide were tested, and a slide of 20 percent of the window length was used since it did not impair the results, yet substantially accelerated the algorithm.

Another problem that occurred was that the initial window length was found to be too long. A window of five minutes would often fit more than one peak, and since the machine learning algorithms only indicate whether the window contains a peak or not, this led to some insecurity regarding which peaks were genuinely identified. This was solved by using only 2.5 minutes of TOCO signal and filling the beginning and end of each window with the median value of the used TOCO data, visualized in figure 5.7. Using 2.5-minutes long segments builds on the theory of contractions lasting between 60 and 180 seconds.



(a) Before: two peaks fitting a five minute window.

(b) After: focusing on one peak, using a 2.5 minute segment, filling the beginning and end with the median value.

Figure 5.7: Before and after using a 2.5 minute window with filling of median value, instead of the whole 5 minute segment. This to mitigate the risk of identifying several peaks in the same window.

The noise in the data was not a problem while only working with the individual peaks. However, the use of the sliding window algorithm led to issues with identified peaks that only were artifacts in the data. This problem was solved by interpolation of the data through the built-in *filloutliers* method in Matlab, using the moving median for the identification of outliers.

For the support vector machine, k-nearest neighbors, and the decision tree algorithms, the default settings in Matlab were used. The convolutional neural network was tested with different training options and layers. Three different layer settings with individually specified training options were chosen. The models were gradually increased in the number of layers. The first model had one convolutional layer and a total of seven layers, the medium model had three convolutional layers and a total 13 layers, and the large model had four convolutional layers and a total 23 layers including batch normalization layers and a dropout layer. The large model was inspired by the structure of the best convolutional neural network used to evaluate peaks in chromatographic data in a report by Risum and Bro [16]. Specifics regarding the layers and training options for the CNN models are shown in Appendix B.

5.6 Results

In this section, the results of the different machine learning classifiers are presented. All results are based on the classification of a test set that has not been used during the training of the models. This is done to ensure non-biased results. The test set consists of 20 percent of the total data set of perfect peak data, more specifically 200 peaks and 200 non-peaks saved in 5-minute segments. A positive prediction implies classifying the 5-minute segment as a peak, whereas a negative prediction equals classifying the 5-minute segment as not containing a peak. The classification is represented as a binary outcome, where 1 is the equivalent of a window containing a peak and 0 is the equivalent of a window not containing a peak.

Statistical measures for all models are presented in table 5.2 with mean and standard deviations based on the 10-fold cross-validation. Moreover, confusion matrices are displayed in figure 5.8 for the models in question. Finally, the best model for each machine learning algorithm is tested on actual CTG files, where each model identifies peaks in the TOCO curve using a moving window with a 30-second slide.

As can be seen in table 5.2 the k-NN has the best results for all statistical measures. The model has a very high sensitivity with a low standard deviation, 98.70% (1.44), implying a high probability of predicting actual peaks as peaks. Moreover, the model has a high mean F1 score, 94.65%, with a low standard deviation of 0.32%, meaning that the model has good accuracy and the ability to both predict peaks and non-peaks. Not far behind is the large CNN model with 23 layers. It has a F1 score of 93.09% with a standard deviation of 0.86% on the test data set. The sensitivity is 95.00 %, with a very low standard deviation of 0.75%. However, the other models, besides SVM, are not much inferior in neither F1 score nor the other statistical measures. Hence, k-NN, DT, and CNN are all models with a good ability to detect peaks in 5-minute segments.

Table 5.2: Results for the six machine learning models. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.8685 ±0.0142	0.6920 ±0.0293	0.7803 ±0.0180	0.7982 ±0.0151
k-NN	0.9870 ±0.0042	0.9015 ±0.0041	0.9443 ±0.0033	0.9465 ±0.0032
DT	0.9395 ±0.0144	0.8925 ±0.0140	0.9160 ±0.0097	0.9179 ±0.0095
CNN_s[1]	0.9050 ±0.0321	0.8455 ±0.0479	0.8753 ±0.0204	0.8790 ±0.0183
CNN_m[2]	0.9270 ±0.0221	0.8905 ±0.0387	0.9088 ±0.0119	0.9105 ±0.0098
CNN_l[3]	0.9500 ±0.0075	0.9090 ±0.0145	0.9295 ±0.0090	0.9309 ±0.0086

When using the best classifier for each machine learning algorithm on actual CTG files, there are some differences. These are visualized in figure 5.9 and 5.10. Although, more or less equal results in accuracy and F1 score, both regarding the mean value and standard deviation, there are differences in the number of peaks identified for each file. For example, the best k-NN model identifies 19 peaks in mat-file 1, see figure 5.9. This is the second least number of identified peaks, while the best SVM model identifies 35 peaks. The other four models identify 22 to 24 peaks. When manually assessing the TOCO curve for mat-file 1, 28 peaks are identified, which can be seen in figure 5.1. For mat-file 26, segment 5, the k-NN

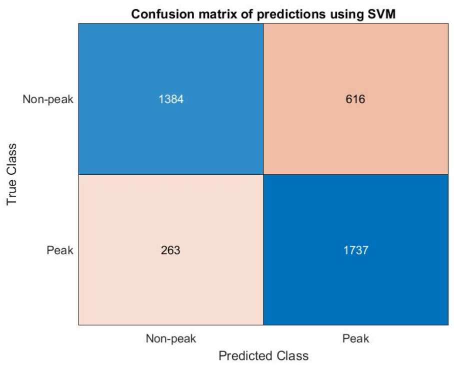
[1]Small, specifications in Apendix B
[2]Medium, specifications in Apendix B
[3]Large, specifications in Apendix B

identifies fewer peaks than the other models, with 44 contractions compared to 46 and 48 contractions, see figure 5.10. The manual assessment detects 42 peaks in the same segment, see figure 5.2.

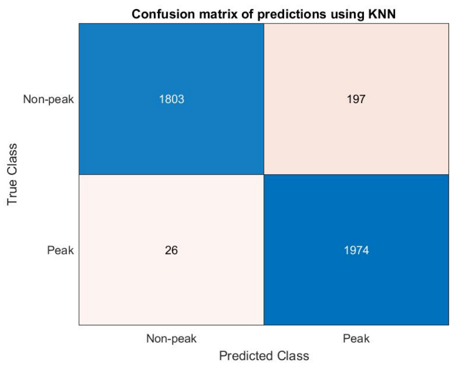
5.7 Discussion

The previous section concluded that the k-NN model had the best statistical measures. The model had an average F1 score of 94.65% with a standard deviation of 0.32%. Moreover, it had a high sensitivity of 98.70%, implying that the model is good at predicting positive outcomes, i.e., it is good at identifying uterine contractions in the TOCO curve. However, the CNN_l, CNN_m, and DT models also performed well. All three models had a F1 score above 90% with a standard deviation around 0.90%. Even though the models generated similar results regarding the F1 score and accuracy, there were differences in performance when using the best models on actual CTG files. The k-NN model may have higher accuracy and F1 scores than the others, however, it seems to be more restrictive in predicting peaks in CTG files. Compared to the other best models, the best k-NN model and CNN_s model classified the least amount of peaks in CTG file 1. The results were different for file 26, which had more profound peaks with a higher frequency. For file 26, segment 5, the models were more or less equal in the resulting number of peaks. Nevertheless, there was a difference in where the peaks were detected and indicated. The best k-NN model and CNN_l model detected less peaks than the manual detection of 28 peaks in mat file 1. The k-NN model indicated nine peaks fewer and the CNN model detected ten peaks less. This implies that even if the statistical measures were high, the machine learning models still failed to detect some peaks that clinically would be interpreted as uterine contractions. Consequently, there is a risk of missing peaks and wrongly estimating the frequency of uterine contractions. Yet, for mat file 26, segment 5, the k-NN model had the closest estimate of the number of peaks, 44, compared to the manually detected number of peaks, 42. It was still the most restrictive model, but all other models were a bit too liberal and detected more peaks than the manually conducted analysis detected. Nevertheless, it is not certain that the k-NN model is the best model for classifying peaks. Even though it had the best statistical measurements in several categories, it did tend towards being restrictive in the positive classification. In other words, both statistical measures and performance on actual CTG data must be considered when choosing which model to use during the rest of the project.

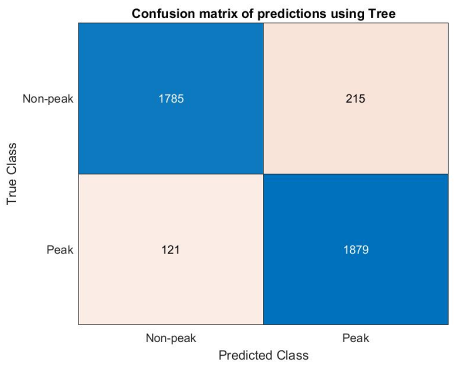
Furthermore, there is more potential in the peak detection algorithm. Instead of solely indicating where the uterine contraction peaks in the TOCO curve, there is potential in finding the duration. As previously mentioned, uterine contractions are evaluated based on frequency, intensity, and duration. The frequency can be calculated using the indices for the local maximas, which is done in the following chapters. The intensity, however, can not be calculated given the available data set, since all TOCO data is measured via an external transducer. This makes pressure measurements dependent on external factors. To be able to calculate the authentic pressure difference, the uterine contractions must be monitored via an intrauterine pressure catheter. The duration of the uterine contraction could be estimated by calculating the difference between the local minimas surrounding the peaks. These values are found in the peak triplet mentioned in section 5.2. Accordingly, there is more potential in the peak detection algorithm that could give valuable information in combination with the FHR curve when predicting fetal distress.



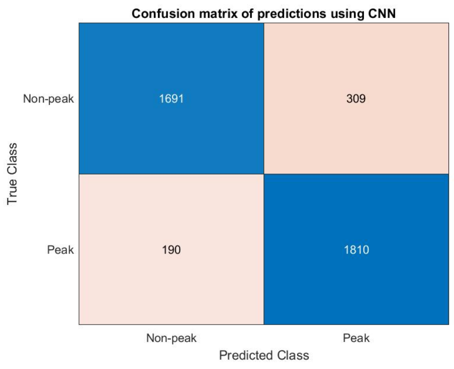
(a) SVM



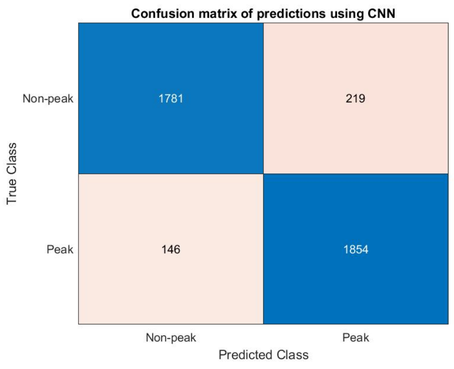
(b) k-NN



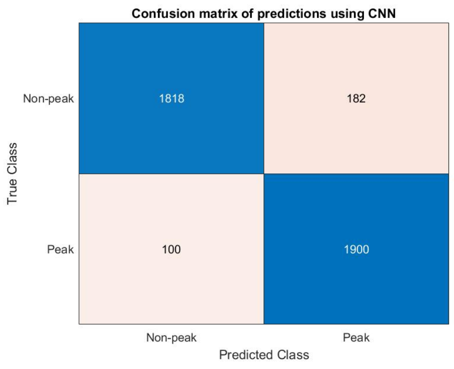
(c) DT



(d) CNN_s

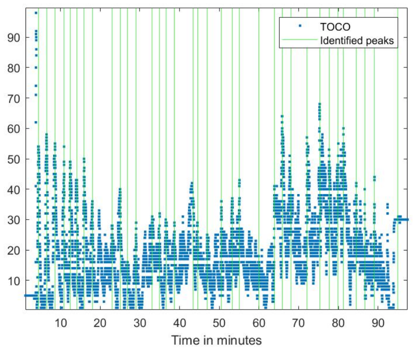


(e) CNN_m

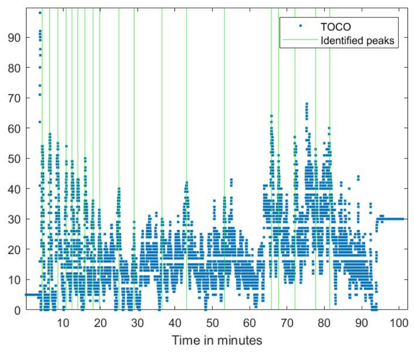


(f) CNN_l

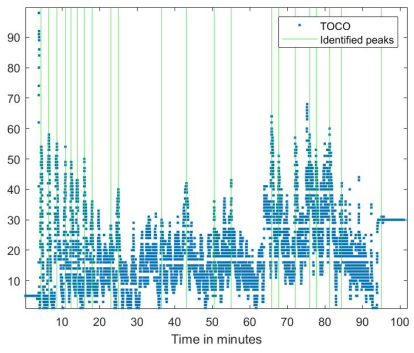
Figure 5.8: Confusion matrices of predictions using six different machine learning models. The predictions were made on perfect test data, total 200 peaks and 200 non-peaks.



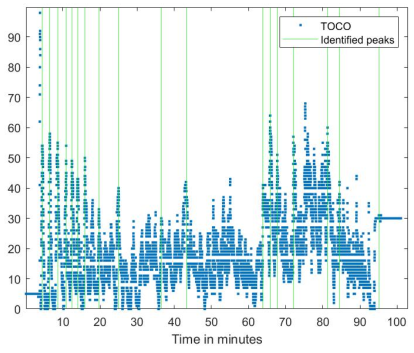
(a) Peak detection using the best model for SVM. The model detects 35 uterine contractions.



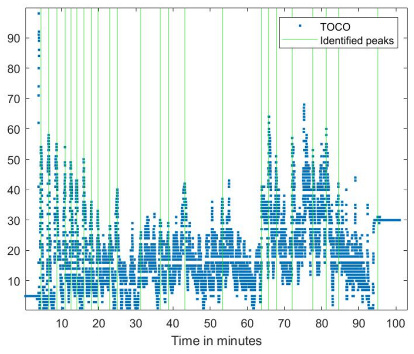
(b) Peak detection using the best model for k-NN. The model detects 19 uterine contractions.



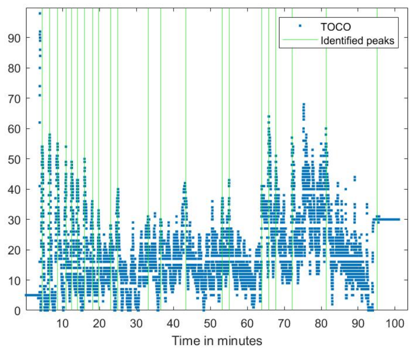
(c) Peak detection using the best model for DT. The model detects 22 uterine contractions.



(d) Peak detection using the best model for CNN_s. The model detects 18 uterine contractions.

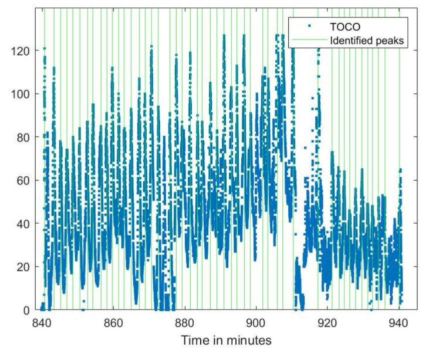


(e) Peak detection using the best model for CNN_m. The model detects 24 uterine contractions.

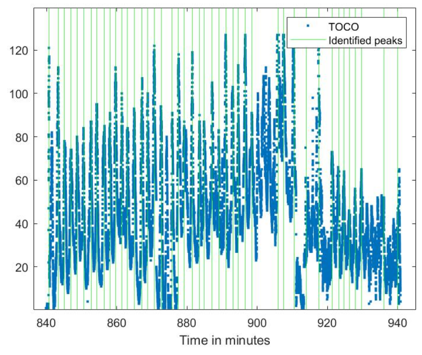


(f) Peak detection using the best model for CNN_l. The model detects 22 uterine contractions.

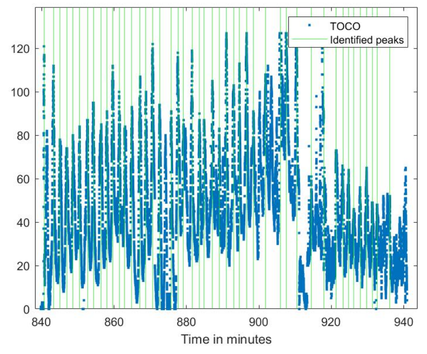
Figure 5.9: Identification of uterine contractions, for mat file with id = 1, using the best model for each machine learning algorithm.



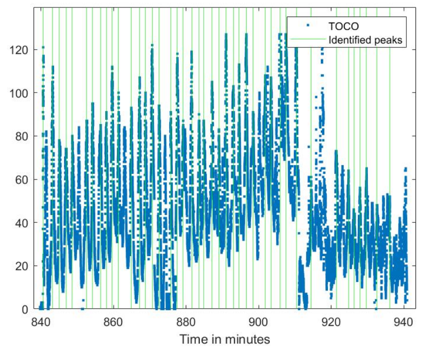
(a) Peak detection using the best model for SVM. The model detects 48 uterine contractions.



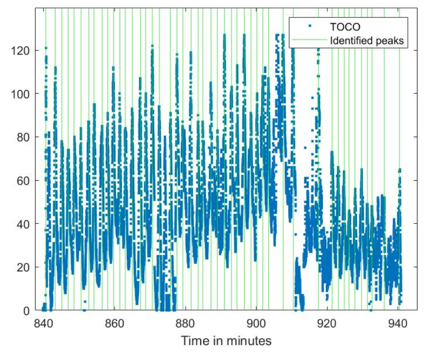
(b) Peak detection using the best model for k-NN. The model detects 44 uterine contractions.



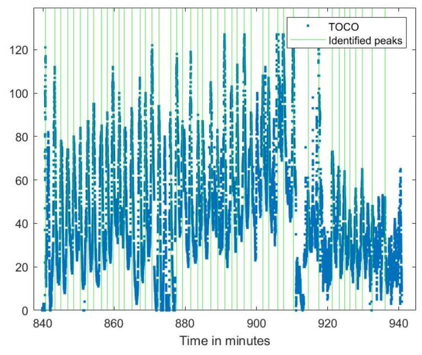
(c) Peak detection using the best model for DT. The model detects 46 uterine contractions.



(d) Peak detection using the best model for CNN_s. The model detects 46 uterine contractions.



(e) Peak detection using the best model for CNN_m. The model detects 47 uterine contractions.



(f) Peak detection using the best model for CNN_l. The model detects 46 uterine contractions.

Figure 5.10: Identification of uterine contractions, for mat file with id = 26, segment 5, using the best model for each machine learning algorithm.

6. Find the Stage of Labor

In the following chapter, two different approaches for segmenting stages of labor are introduced. The first algorithm uses pattern recognition and machine learning to find distinctions between late and non-late stages, and the second algorithm uses the frequency of uterine contractions to segment out stages. For both cases, the segmentation, method, and results are presented. It is followed by a discussion of the algorithms.

6.1 Justify the Identification of Labor Stages

A normal labor goes through three different stages, all with their own characteristics. Given the differences in nature, there could be a possibility of finding distinctions in the CTG curve that could indicate how long the labor would proceed or which stage the labor has entered. These indications could be favorable in a later classification of Apgar score, since different features could be important for the fetal status and the outcome of labor depending on which stage the labor has entered.

The given signal data does not contain information regarding when the monitoring of the mother started, relative to when the labor started, nor information about the cervical dilation. Further, there is no information regarding the time of birth, implying that it is not certain that the last segment in a CTG file in fact is the last minutes before birth. It is thus hard to make distinctions from the given data on how long the labor has proceeded or when the labor is terminated. Accordingly, an algorithm to find and classify the stages of labor in a CTG file could be value-adding to further comparisons and predictions of the outcome of labor.

6.2 Classify Segments Using Machine Learning

This part aims to design an algorithm that can identify if it is more than 30 minutes left of the delivery or not. Due to the files missing an indicator of when the child is born, it is not possible to classify if it is *less* than 30 minutes left of the labor. However, all segments earlier than the last 30 minutes of a CTG file are in fact not in the last 30 minutes of birth, and thus these can be classified as non-late.

The aim to classify if it is more than 30 minutes left of the delivery is based on the assumption that the late segment should be within the second stage of labor. The duration of the expulsion phase can vary between one and three hours, however, the active pushing should not exceed 45-60 minutes. Accordingly, the choice of 30-minute segments builds on the assumption that they have a higher possibility of being in the active phase, and thus have a higher chance of being distinctly different from the earlier segments in the CTG data. Since there is no indicator of when the labor is terminated the last segment could not be too long, as it would risk being in several stages. Moreover, longer segments would also reduce the number of segments in the data set, as the CTG often is measured continuously for 20 to 30 minutes.

6.2.1 Segmentation

The non-late segments were extracted from the previously mentioned 52 344 segments and consisted of 30 minutes of continuous data. If the extracted segment was longer than 30 minutes, the starting point of the new non-late segment was chosen randomly. If the extracted segment was shorter than 30 minutes, it was not included in the data set. Moreover, the non-late segments were not extracted from the last segment for each CTG file, as these were segmented separately.

The late segments were extracted from the 52 344 segments and consisted of 30 minutes of continuous data from the last segment of each CTG file. For example, for file 26, six segments were extracted that contained more than 20 minutes of data. In this case, the sixth segment had the potential of being classified as a late segment if fulfilling two conditions. The first condition was that the segment should have 30 minutes of continuous data. The second condition was that the last data point of the potential late segment had to be within the last ten minutes of the original CTG file. Otherwise, the segment was not considered late enough. This displacement, of ten minutes, was allowed since a large part of the files contained a few late data points which, if no difference was allowed, would have excluded many of the late segments. However, the maximum limit was needed to ensure that the segments were not too early in the process.

The new segments contained both TOCO and FHR values, and the peaks in each segment were identified with the best k-NN model from the previous section. The clinical data, corresponding to the main CTG file, were also saved for each segment. The segments were given a binary label, depending on whether they were late or not. The bottleneck, in this case, was the late segments. In total, 6463 late segments were extracted compared to 25 004 non-late segments. Two different training sets, as well as one test set, were assembled from the equal distribution of late and non-late segments. All of these sets were balanced, with 50 percent late segments and 50 percent non-late segments. The large training set contained 10 340 files and the test set included 2585 files. The smaller training set was a subset of the larger, and it consisted of 1000 files. This set was used for testing different combinations of features, and for training larger, more time-consuming CNN models. Research has shown that the optimal structure for a smaller data set can be used as guidance for the starting point to find the optimal structural hyperparameters on which the entire data set is trained. This reduces the number of samples processed and minimizes the time needed to find the optimal training structure [48]. The used test data was the same regardless of which training set was used.

6.2.2 Method

Six different machine learning algorithms were used to classify non-late and late segments. The six machine learning algorithms were decision tree, k-nearest neighbor, support vector machine, and three different setups of convolutional neural networks, specified in Appendix B.

The input data consisted of both vectors and single constants from the signal and clinical data. The output was a binary classification where 0 equaled a non-late segment, and 1 corresponded to a late segment. A negative classification implied a segment being a non-late segment, meaning that it is at least than 30 minutes until birth. A positive classification implied a late segment. However, being classified as a late segment does not imply, by certainty, that it is within the last half hour of labor since it is not sure that the CTG is measured until birth. Nevertheless, it is an indication of different stages of birth and could thus be a value-adding classification when later progressing to classifying Apgar.

The smaller data set, of 1000 samples, was used to evaluate different combinations of features more time-efficiently. Moreover, the feature selection was only based on DT, k-NN, SVM, and CNN_s to reduce time expense further. After evaluating the F1 score for different combinations of features, the best combinations of features were tested for the larger training set for the four above-mentioned models. The best combination of features was also tested for the larger CNN models using the smaller training set. Finally, the models with the best feature combinations were used on the test set, which was never used in the training of the models. All algorithms were used with 10-fold cross-validation, to ensure that the result was not haphazardly dependent on the data set.

Feature Selection

The first features introduced, to classify the late and non-late segments, were from the CTG data. The base case was classifying the stage of labor just using the FHR and TOCO as input. The other cases introduced features from the clinical data, known prior to birth, as well as derived features from the CTG data as input. Below are the explanations of the derived features from the CTG data.

Uterine Contractions

Uterine contractions, and their frequency, were used as a feature since the frequency of contractions could potentially differ between stages of labor. Closer to birth, in the second stage, the frequency intensifies compared to the first stage. The uterine contractions were detected using the best k-NN model from the previous section. The output from the k-NN model was a binary vector, with the same length as the TOCO signal, where 1 indicated a peak in the TOCO curve whereas 0 did not. The peak vector was used as an input for the classification of late and non-late segments in two different ways, as a whole vector and as the frequency per hour. The entire binary vector of ones and zeros, $v_i,\ i = 1 : 7200$, was used since it would indicate more specifically how the frequency differed over time. The vector showcases where the peaks were located in the TOCO curve, and could thus give valuable information regarding the uterine contractions in combination with the FHR and TOCO curves. On the other hand, the peaks are present in the TOCO curve, and thus the indices for where the contraction peaks could be redundant information. Instead, the vector was also used to calculate the frequency per hour in the 30-minute segment. The frequency was calculated as follows:

Assume v is the binary vector of peaks from the k-NN model then:

$$Frequency = \sum_{i=1}^{n} v_i * \frac{14400}{||v||}$$

where $n = ||v||$ is the length of the vector, which equals 7200 data points for a 30-minute segment. 14 400 is the number of data points for one hour.

Normalizing the TOCO Curve

Another feature used as input was the normalized TOCO curve. Since the values in the TOCO curve do not represent the actual pressure, the values in the vector could give wrongful information to the machine learning algorithms. Depending on the tightening of the external TOCO transducer, the same internal contraction pressure could give different results in the TOCO measurement. Hence, the absolute pressure value is not strictly comparable between different samples. To normalize the TOCO curve, the used method was to calculate the standard deviation of each segment and divide each TOCO value by this number. The reason for not normalizing the TOCO values between zero and one was due to not wanting to lose too much of the characteristics of the data. In addition to the transformed TOCO curve, the standard deviation was used as a separate input as this could contain valuable information.

Several Vectors as Input

When using several vectors as input, for example, both the TOCO and FHR vectors, it generated some problems. Ideally, the machine learning algorithms would take in both vectors parallel to each other, and evaluate them simultaneously since they are dependent on each other. However, the only machine learning algorithm that folds over the vectors is the CNN algorithm. Therefore, this algorithm was used with the vectors parallel to each other, and the rest of the machine learning algorithms were used with a single row vector as input, putting the vectors in sequence. The input data for the CNN consisted of a matrix where the first column contained the TOCO data and the chosen features as constants at the end of the same vector. The second column had the same length and contained the FHR data, but with zeros at the end instead of the feature constants. This would ensure that the CNN algorithm would evaluate the vectors simultaneously, and thus find information regarding how the FHR and TOCO depend on each other. Further, in cases where the peak vector also was included, the matrix consisted of three columns. The peak vector was located in the second column in the input matrix, after the TOCO vector, with the same number of zeros at the end as the number of constants. It can be represented as follows:

$$\begin{bmatrix} T_1 & P_1 & F_1 \\ T_2 & P_2 & F_2 \\ \vdots & \vdots & \vdots \\ T_n & P_n & F_n \\ C_1 & 0 & 0 \\ C_2 & 0 & 0 \\ \vdots & \vdots & \vdots \\ C_N & 0 & 0 \end{bmatrix}$$

where T is the TOCO-vector, P is the peak-vector and F is the FHR-vector, n is the length of the vectors, equal to 7200 data points, and N is the number of constants.

6.2.3 Results

In the following section, the results from the classification of the non-late and late segments are presented. The first part contains the mean results from the small data set of 500 late and 500 non-late segments, with 10-fold cross-validation. A positive prediction implies classifying the 30-minute segment as a late segment, whereas a negative prediction is classifying the 30-minute segment as not being in the last 30-minutes of labor. The classification is represented as a binary outcome, where 1 is equivalent to being a late segment and 0 is equivalent to not being in the last 30 minutes of labor.

The second part presents the results from the models with the best combination of features. These results are based on the classification of a test set that has not been used during the training of the models. In these cases, the statistical measures, including the mean and standard deviation based on 10-fold cross-validation, as well as the confusion matrices are presented.

Feature Combinations

Table 6.1 contains the result from the base feature combination. It only takes the TOCO and FHR signal as input. When comparing the mean F1 scores, the best machine learning model is the DT, with 79.44% (3.18). However, the k-NN model has higher specificity, implying that it is better at classifying segments that are not in the last 30-minutes of labor. When normalizing the TOCO signal with the standard deviation, and adding the standard deviation as a feature, the F1 score for the k-NN model increases from 69.21% to 72.09%, however increasing the standard deviation, see table 6.2. Yet, the DT model still has the highest mean F1 score of 78.32%. The small CNN model does not give any satisfying results when normalizing the TOCO curve. It seems that when this is done, some of the information is lost, and the model cannot find any patterns connected to the different stages of labor.

Table 6.1: The results from evaluation of data set 500 late and 500 non-late segments. Features: TOCO & FHR

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6840 ±0.0717	0.7060 ±0.0806	0.6950 ±0.0677	0.6916 ±0.0681
k-NN	0.6260 ±0.0633	0.8200 ±0.0442	0.7230 ±0.0353	0.6921 ±0.0460
DT	0.7922 ±0.0513	0.7760 ±0.0572	0.7910 ±0.0348	0.7944 ±0.0318
CNN_s	0.5200 ±0.0742	0.6340 ±0.0712	0.5630 ±0.0668	0.5494 ±0.0539

Table 6.2: The results from evaluation of data set 500 late and 500 non-late segments. Features: normalized TOCO, Standard Deviation & FHR

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6760 ±0.0659	0.6500 ±0.0627	0.6630 ±0.0467	0.6667 ±0.0483
k-NN	0.7680 ±0.0779	0.6380 ±0.0802	0.7030 ±0.0670	0.7209 ±0.0653
DT	0.7940 ±0.0472	0.7660 ±0.0680	0.7800 ±0.0424	0.7832 ±0.0388
CNN_s	NaN	NaN	NaN	NaN

In table 6.3, the peak frequency per hour is added as a feature in addition to the TOCO and FHR signal. For this combination of features, the DT model has the highest mean F1 score of 78.88%. The k-NN model has the highest mean specificity, with 83.40%, implying that the model is good at classifying non-late segments. On the contrary, the DT model has the highest mean sensitivity, 81.40%, meaning that it is the best at classifying late segments. When adding the peaks as a vector, see table 6.4, the DT model still has the highest mean F1 score, with 78.35%. Moreover, this feature combination gives the highest mean F1 score, 55.14%, of all combinations regarding the CNN_s model. To be noted is the high standard deviation, 14.49%. This is due to one k-fold with very unsatisfying results. If excluding the seventh fold, the F1 score increases to 59.25% with a 6.82% standard deviation. It is an indication that the CNN algorithm benefits from, and can find patterns when, the three vectors are folded over simultaneously, and thus including the peak vector as one feature could be beneficial for the CNN models.

Table 6.3: The results from evaluation of data set 500 late and 500 non-late segments. Features: TOCO, Peak Frequency (per hour) & FHR

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6800 ±0.0411	0.7180 ±0.0416	0.6990 ±0.0260	0.6929 ±0.0280
k-NN	0.6460 ±0.0822	0.8340 ±0.0422	0.7400 ±0.0442	0.7110 ±0.0608
DT	0.8140 ±0.0566	0.7520 ±0.0368	0.7830 ±0.0254	0.7888 ±0.0301
CNN_s	0.5140 ±0.1132	0.6380 0.0851	0.5760 ±0.0600	0.5434 ±0.0888

Table 6.4: The results from evaluation of data set 500 late and 500 non-late segments. Features: TOCO, Peak Vector & FHR

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.7000 ±0.0772	0.7160 ±0.0804	0.7080 ±0.0630	0.7051 ±0.0655
k-NN	0.6540 ±0.0589	0.8240 ±0.0350	0.7390 ±0.0292	0.7137 ±0.0398
DT	0.7960 ±0.0729	0.7660 ±0.0654	0.7810 ±0.0387	0.7835 ±0.0414
CNN_s	0.5580 ±0.1851	0.6060 ±0.1526	0.5820 ±0.0539	0.5514 ±0.1449

When normalizing the TOCO curve, see table 6.5, the small CNN model still has trouble finding patterns separating the two classes. The F1 scores are worse when normalizing the TOCO curve for the DT and SVM models, all else being equal. The mean F1 scores are reduced from 78.88% and 69.29% to 77.20% and 67.55%, respectively. However, the F1 score increases for the k-NN model from 71.10% to 72.72%.

Table 6.5: The results from evaluation of data set 500 late and 500 non-late segments. Features: normalized TOCO, Standard Deviation, Peak Frequency (per hour) & FHR

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6780 ±0.0977	0.6780 ±0.0636	0.6780 ±0.0514	0.6755 ±0.0652
k-NN	0.7820 ±0.0569	0.6320 ±0.0732	0.7070 ±0.0353	0.7272 ±0.0322
DT	0.7800 ±0.0869	0.7640 ±0.0624	0.7720 ±0.0355	0.7720 ±0.0449
CNN_s	NaN	NaN	NaN	NaN

Table 6.6 presents the results when including all clinical data known prior to birth. The best model, regarding the mean F1 score, is the DT with 79.22% and 3.98% standard deviation. In table 6.7, the gestational age is removed, and further in table 6.8 preeclampsia and diabetes are also removed. The highest F1 score of all combinations can be found for the DT model when having the features: *TOCO, Peak Frequency (per hour), FHR, Parity, Smoking, and Placental Abruption.* It has a mean score of 80.06% with a standard deviation of 4.49%. However, it is only marginally better than the smallest base model with the features: *TOCO and FHR* (mean F1 score of 79.44 %), and thus this should be premiered according to the simplicity rule when selecting features.

Table 6.6: The results from evaluation of the data set with 500 late and 500 non-late segments. Features: TOCO, Peak Frequency (per hour), FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes & Gestaional Age

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6860 ±0.0626	0.7300 ±0.0492	0.7080 ±0.0416	0.7007 ±0.0477
k-NN	0.6420 ±0.0569	0.8380 ±0.0636	0.7400 ±0.0316	0.7110 ±0.0382
DT	0.7920 ±0.0627	0.7940 ±0.0558	0.7930 ±0.0362	0.7922 ±0.0398
CNN_s	0.5000 ±0.0933	0.6380 ±0.0902	0.5690 ±0.0300	0.5334 ±0.0559

Table 6.7: The results from evaluation of the data set with 500 late and 500 non-late segments. Features: TOCO, Peak Frequency (per hour), FHR, Parity, Smoking, Placental Abruption, Preeclampsia & Diabetes

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6740 ±0.0660	0.6960 ±0.0556	0.6850 ±0.0532	0.6810 ±0.0560
k-NN	0.6440 ±0.0826	0.8280 ±0.0575	0.7360 ±0.0624	0.7078 ±0.0750
DT	0.8000 ±0.0422	0.7280 ±0.0930	0.7640 ±0.0558	0.7732 ±0.0462
CNN_s	0.5360 ±0.0853	0.5860 ±0.0924	0.5610 ±0.0477	0.5475 ±0.0588

Table 6.8: The results from evaluation of the data set with 500 late and 500 non-late segments. Features: TOCO, Peak Frequency (per hour), FHR, Parity, Smoking & Placental Abruption

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.7000 ±0.0843	0.7020 ±0.0457	0.7010 ±0.0423	0.6988 ±0.0546
k-NN	0.6440 ±0.0771	0.8300 ±0.0785	0.7370 ±0.0633	0.7094 ±0.0727
DT	0.7980 ±0.0466	0.8040 ±0.0602	0.8010 ±0.0458	0.8006 ±0.0449
CNN_s	0.4920 0.0725	0.6540 ±0.0667	0.5730 ±0.0386	0.5335 ±0.0549

Best Feature Combinations

Three different feature combinations are tested for the large training set, as well as the medium and large CNN models. The first feature combination is derived from the smallest model with a high F1 score, the second combination has good potential for the larger CNN models, and the third combination is obtained from the highest F1 score of all combinations:

- TOCO & FHR
- TOCO, Peak Vector & FHR
- TOCO, Peak Frequency (per hour), FHR, Parity, Smoking & Placental Abruption

The results for the first feature combination are presented in table 6.9 and figure 6.1. As can be seen, the CNN_l model has the best results in all statistical measures. The specificity, i.e., the ability to classify non-late segments, is 80.05 % with a 1.70% standard deviation. Further, the F1 score is 78.52% with a standard deviation of 0.76%. The other models, SVM, k-NN, and DT perform worse when increasing the sample size from 1000 to 10 341 samples. For example, the F1 score for the DT model reduces from 79.44% to 77.02%. However, the standard deviation decreases from 4.60% to 0.5%, implying a more stable classification algorithm when increasing the sample size. It is not as dependent on the distribution of samples in each k-fold when there are more samples included in the whole data set. The small CNN model does not give any satisfying results when the sample size is increased. Instead, the model is not able to make predictions at all with this setup of layers [1]. Since the smaller data set is a subset of the larger data set, it is not certain that the smaller one captures the structure of the larger one. Accordingly, the challenge is to find the structure of the optimal network appropriate for a specific sample size. Generalizing building rules may not be applicable for this data set since this would assume that the subset has the same structure as the entire data set. This could explain the varying results of the small CNN model when increasing the training set from a subset to the entire data set.

Table 6.9: Results for the six machine learning models for feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6584 ±0.0133	0.6780 ±0.00134	0.6682 ±0.0070	0.6648 ±0.0079
k-NN	0.6455 ±0.0065	0.7614 ±0.0053	0.7035 ±0.0039	0.6851 ±0.0046
DT	0.7783 ±0.0088	0.7575 ±0.0106	0.7679 ±0.0051	0.7702 ±0.0050
CNN_s[2]	NaN	NaN	NaN	NaN
CNN_m[3]	0.7007 ±0.0250	0.7676 ±0.0337	0.7342 ±0.0101	0.7248 ±0.0098
CNN_l[4]	0.7755 ±0.0125	0.8005 ±0.0170	0.7880 ±0.0080	0.7852 ±0.0076

When introducing the peak vector as a feature, the F1 score remains almost the same, as can be seen in table 6.10. The best score, 77.11%, is from the DT model. For the SVM, k-NN, and medium CNN models, the F1 scores increase marginally. Moreover, when including the peak vector as an input, the small CNN model is able to make predictions. Nevertheless, the small model is substantially worse than both the medium and large CNN models, 56.08% (4.14%) compared to 73.33% (1.53%) and 76.49% (0.96%), implying that seven layers are not enough to capture the patterns in the input data and classifying non-late and late segments. Figure 6.2 presents the performance of the classification algorithms for the given feature combination.

Table 6.10: Results for the six machine learning models for feature combination *TOCO, Peak Vector & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6567 ±0.0112	0.6848 ±0.0192	0.6708 ±0.0087	0.6660 ±0.0074
k-NN	0.6457 ±0.0057	0.7619 ±0.0147	0.7038 ±0.0056	0.6854 ±0.0036
DT	0.7725 ±0.0039	0.7691 ±0.0080	0.7708 ±0.0033	0.7711 ±0.0026
CNN_s	0.5398 ±0.0701	0.6223 ±0.0856	0.5810 ±0.0264	0.5608 ±0.0414
CNN_m	0.7060 ±0.0468	0.7817 ±0.0509	0.7439 ±0.0104	0.7333 ±0.0153
CNN_l	0.7560 ±0.0149	0.7794 ±0.0197	0.7677 ±0.0099	0.7649 ±0.0096

[1]Different subsets of the entire data sets are tested for the small CNN, however, it is only smaller sets of approximately 2000 samples that the model can make predictions on.

[2]Small, specifications in Apendix B

[3]Medium, specifications in Apendix B

[4]Large, specifications in Apendix B

In table 6.11 and figure 6.3, the results of the feature combination: *TOCO, Peak Frequency (per hour), FHR, Parity, Smoking & Placental Abruption* are presented. The best model according to the F1 score is the CNN_l model with a mean of 77.16% and a standard deviation of 0.69%. The DT model has the same mean F1 score, however, the standard deviation is marginally higher (0.77%). When comparing the two models, the large CNN model has higher specificity, i.e., the ability to predict non-late segments, 79.64% compared to 76.42%. On the other hand, the DT model has a higher sensitivity, 77.63% compared to 75.61%. The small CNN model is not able to make predictions on this data set with the given feature setup. All results show 'Not a number' [5]. Accordingly, results from the small CNN model are not presented for this feature combination.

Table 6.11: Results for the six machine learning models for feature combination *TOCO, Peak Frequency (per hour), FHR, Parity, Smoking & Placental Abruption.* The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6574 ±0.0090	0.6638 ±0.0141	0.6606 ±0.0086	0.6595 ±0.0079
k-NN	0.6462 ±0.0068	0.7613 ±0.0105	0.7038 ±0.0040	0.6856 ±0.0038
DT	0.7763 ±0.0051	0.7642 ±0.0135	0.7703 ±0.0087	0.7716 ±0.0077
CNN_s	NaN	NaN	NaN	NaN
CNN_m	0.7122 ±0.0683	0.7545 ±0.0938	0.7333 ±0.0191	0.7268 ±0.0175
CNN_l	0.7561 ±0.0103	0.7964 ±0.0187	0.7763 ±0.0080	0.7716 ±0.0069

When testing the two best models from the best feature combination on actual CTG files, the results are varying. The best feature combination is TOCO and FHR, and the two best models are DT and CNN_l. For mat file 1, the two models classify four different 30-minute windows as late segments, implying that these segments are similar to the segments in the last 30-minutes of the CTG files. When merging segments with overlapping 30-minutes windows, both models identify the last 60 minutes in the file as 'late'. This is visualized in figure 6.4.

For mat file 3, the duration of the CTG measurement is longer, see figure 6.5. In total, the file is over 2200 minutes ($\sim$ 37 hours). Both models detect late segments within the last three hours of measurement. However, these segments are not consistent and there are periods in-between classified 'late' segments where the model does not identify segments as late. The CNN model ends up with three segments after merging is conducted, while the DT model ends up with two segments. Nevertheless, both models classify the end of the CTG file as a late segment.

When using the models on mat file 26 there is a larger difference in the number of identified late segments. This is visualized in figure 6.6. While the CNN model classifies three 30-minute windows as late, the DT model identifies 20 windows as late. However, both models detect late segments in the last seven hours of labor. When merging the windows to exclude overlapping segments, the CNN model ends up with two segments. The first segment is approximately 6.5 hours from termination of the CTG measurement, while the second segment is within the last one to two hours of the measurement. The DT model identifies five segments, where the last four segments have small gaps in-between, located in

[5]Different subsets of the entire data sets are tested for the small CNN, however, it is only the smaller sets of approximately 2000 samples that the model can make predictions on.

the last 5 hours of the measurement. Similar to the CNN model, the DT model classifies a 30-minute window as late while being 6.5 hours from the termination of monitoring.

6.3 Segmentation using Contraction Frequency

This part aims to extract segments that are in the same stage of labor by using the frequency of uterine contractions. Since the data do not contain information regarding cervical dilation or the time of birth, it is difficult to identify the different stages of labor. The expectation is that by identifying segments where a certain frequency is achieved, these segments will represent the same stage of labor.

6.3.1 Method

The minimum threshold was set to three peaks in a 10-minute period building on the assumption that these segments would capture the beginning of the active phase of labor. To extract these segments, all files were iterated over, and the previously identified 'good segments' were examined. Each CTG file that contains one, or more, useful segments, according to the earlier established requirements was evaluated. The useful segments were analyzed in ascending order, from the first to the last segment in each CTG file. The segments were iterated over to find the first 10-minute period that contained three peaks. When this subsegment was found, it was combined with the succeeding 20 minutes of data, and the whole 30-minute segment was saved. Every 10-minute segment that was examined was followed by a minimum of 20 minutes of continuous data, ensuring that the saved segments did not contain interruptions.

Two different methods were examined. The first one comprised a requirement on the whole 30-minute segment to contain a minimum of nine peaks, i.e., an average of at least three peaks per 10-minute period. If this requirement was not met, the algorithm kept iterating over the file until a segment that met the criteria was found, otherwise, no segment was saved for the CTG file in question. The second method did not have a requirement based on the total frequency in the 30-minute segments. In this case, the selection was solely based on the frequency of uterine contractions during the first 10 minutes of the segment. Hence, the only requirement for the subsequent 20 minutes was that the data met the basic criteria, i.e., that it was included in the 'good segments'. After comparing the outcome of the two methods, the second alternative was chosen for segmentation. The reason for this was that the second method resulted in a larger data set, and that the previously chosen threshold was set to apply for 10 minutes. Thus, it seemed reasonable to do segmentation based exclusively on this criteria.

To identify the peaks, and thus get the frequency, five different peak detection models were used. For the first 1000 CTG files, a comparison was made between the different peak identification models to see how many segments with the required frequency each model found. The model that found the most frequency segments among the first 1000 CTG files and the best model, based on a statistical assessment, were used to extract frequency segments from the full data set. The extracted segments consisted of 30 minutes of data, i.e., 7200 data points, from the first time that the selected frequency was achieved. The reason for using two different models to create data sets was mainly to evaluate whether the algorithms found different segments in time, and if the number of segments differed.

6.3.2 Results

In the following section, the results from the frequency segmentation are presented. First, the number of found segments for the first 1000 CTG files for all five models is presented. Secondly, two of the models are used on the full data set of 'good segments'.

In table 6.12, the results from identifying peaks using the different machine learning models are shown. As seen from the table, using the SVM and CNN_m models results in the highest number of segments extracted from 1000 CTG files, 551 and 495 segments, respectively. Using the three other peak detection models results in the same amount of segments, 460.

Table 6.12: The number of found segments with the specified frequency among the first 1000 CTG files, using different machine learning methods

Model	Number of Segments
SVM	551
k-NN	460
DT	460
CNN_s	460
CNN_m	495

The SVM and k-NN models are used on the whole data set, resulting in 62 991 and 52 671 found segments. The SVM model is chosen since it generates the largest number of segments. There are two reasons for choosing the k-NN model. Firstly, it has the highest statistical measures and is considered the best model for peak identification. Secondly, it differs substantially in the number of found segments compared to the SVM model. The reason for assembling two data sets, instead of one, is to enable the ability to compare the outcomes of later analysis. Therefore, it seems reasonable to choose two models that differ in the number of found segments in the first 1000 CTG files.

When evaluating the starting times for these segments, the difference is considerable, see figure 6.7. Some segments start on the first measurement of the whole CTG file, while other segments start 70 hours after the first measurement. Moreover, the time between the start of each frequency segment and the end of the CTG file is also substantial. The time between the first occurrence of three contractions in a 10-minute period, and the end of the CTG file can vary between 3 minutes and 70 hours. However, the majority of KNN segments($\sim$ 52%) has less than 5 hours between the start of the frequency segment and the termination of monitoring the labor. For the SVM segments approximately 50 percent has less than 5.5 hours between start and termination of CTG monitoring. Although the time in itself does not indicate much regarding the labor stage, the large difference is somewhat concerning.

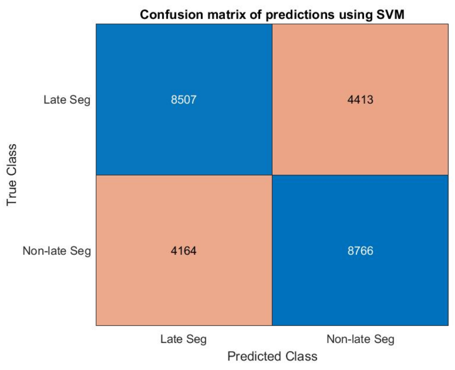
6.4 Discussion

Making distinctions between different stages solely based on the given signal and clinical data is difficult. The results in the previous section showed that the best machine learning algorithm had an 80.05% chance of predicting non-late segments accurately. The best model was the large CNN model with TOCO and FHR as features. It had a F1 score of 78.53% with a 0.76% standard deviation. Since it is not certain that the last 30 minutes of CTG data were actually in the last 30 minutes of labor, it was hard to assess the results of predicting late segments. Nevertheless, the large CNN model had one of the highest sensitivities, 77.55% (1.25%). Only the DT models with TOCO and FHR as features (77.83%, 0.88%) and TOCO, peak frequency, parity, smoking, and placental abruption as features (77.63%, 0.51%) were higher. Moreover, in all feature combinations, the best models were either DT or the larger CNN. The algorithms had a structure that was able to detect patterns in the data to a greater extent.

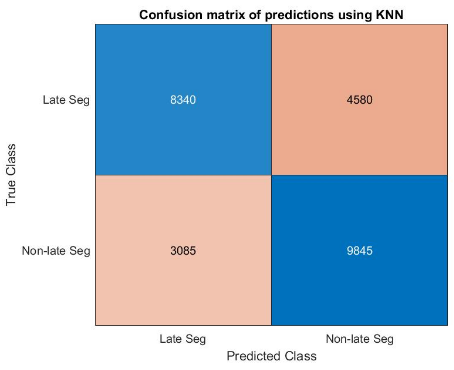
The results also indicated that the clinical parameters did not have a great impact on the statistical performance. It seemed like it was primarily the TOCO and FHR signals that determined how the machine learning algorithms classified segments. The importance of TOCO can be supported by theory, since the duration and frequency of uterine contractions both evaluation tools for determining the stage of labor, alongside cervical dilation. However, since both TOCO and FHR are measured simultaneously during CTG, it was determined to evaluate them together as a base case. Accordingly, the best feature combination for the continuous work of classifying segments should be using TOCO and FHR as input, as this combination had the most satisfying results while being the smallest.

When testing the DT and CNN_l models on actual CTG files, the results varied. For all CTG files, the models seemed to be restrictive in classifying 30-minute segments as late. With an 80% overlap for the sliding window, the models should indicate a late segment five times before the CTG measurement is terminated. However, this was not the case. Instead, the models did not indicate segments consecutively. Nevertheless, when combining segments with overlap, the results were somewhat improved. For mat file 1, the number of detected late segments reduced from four and five to one combined one-hour segment for both models. In theory, the second stage of active labor can proceed for 60 minutes, and thus a one-hour late segment is supported by theory. For mat files 3 and 26, the models classified more segments as late, while not being in the last 30 minutes of labor. To exemplify, the DT model detected 20 late segments, where the first segment started 6.5 hours before the termination of the CTG measurement. Accordingly, the models can not be trusted for real-time monitoring and evaluation regarding the stage of labor. They could indicate a late segment where there was no chance that the labor had entered the second stage. The poor results might be explained by the used data set. A non-late segment is by certainty not within the last 30 minutes of labor, but a late segment has no guarantee to be within the last half hour. Therefore, a manual assessment of the CTG files is needed, to determine if the labor has entered the second phase.

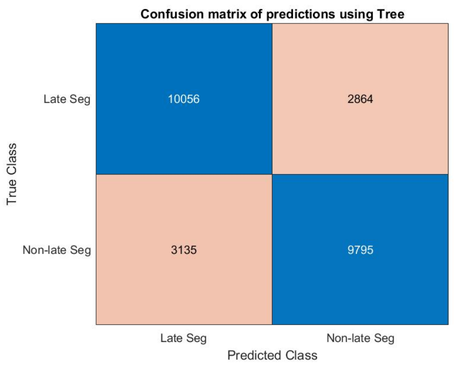
An alternative method to find different stages, besides machine learning, is through assessing frequency. The results showed that using the SVM model for peak detection, extracted the most segments fulfilling the requirement of three contractions during a 10-minute period. However, the SVM model was not the peak detection model with the highest F1 score or sensitivity. When using the best k-NN model with over 98% sensitivity, 460 segments were found compared to 551 segments for the SVM model. As previously mentioned, the k-NN model was a bit more restrictive, but since the accuracy was higher, this model should not be disregarded. Moreover, comparing segments of 30-minutes of continuous data, with the same frequency, should entail a higher chance of comparing similar segments in time. However, the results indicated that this was not the case. Some of the frequency-based segments started within the first minutes of CTG monitoring with between 30 minutes and 60 hours left of monitoring, while other segments started after 65 hours with 35 minutes to 2.5 hours left of monitoring. It is an indication that the segments are not fully comparable. Hence, making distinctions solely based on the given data set is difficult. Parameters regarding cervical dilation or time of birth would be favorable to find and compare segments from the same stages of birth.



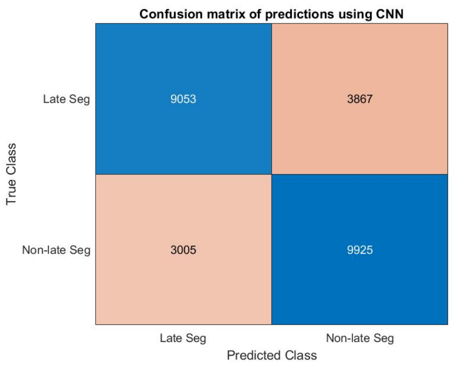
(a) SVM



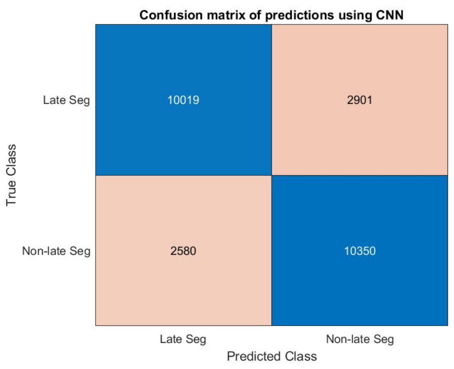
(b) k-NN



(c) DT

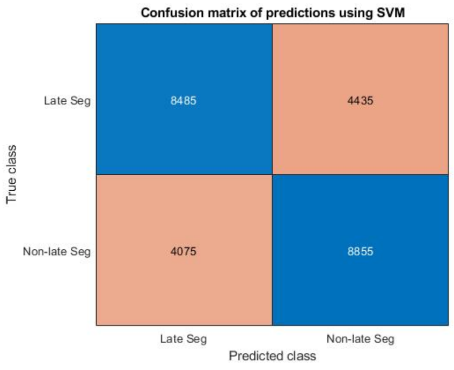


(d) CNN_m

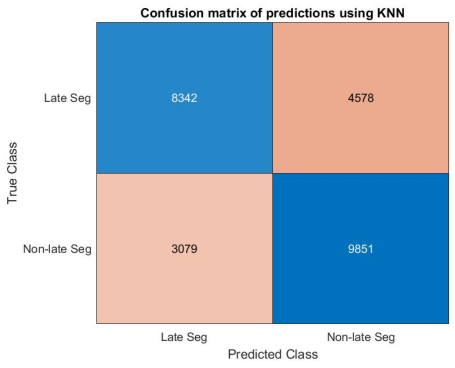


(e) CNN_l

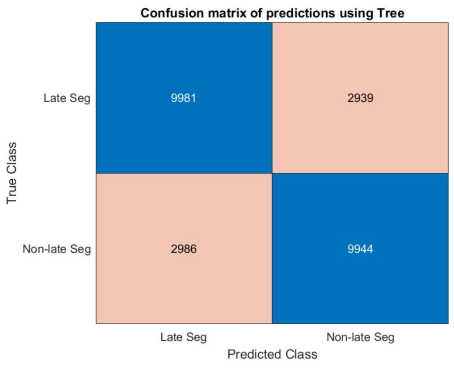
Figure 6.1: Confusion matrices of predictions using five different machine learning models for feature combination *TOCO & FHR*. The predictions were made on test data, in total 2585 samples equally distributed between late and non-late segments.



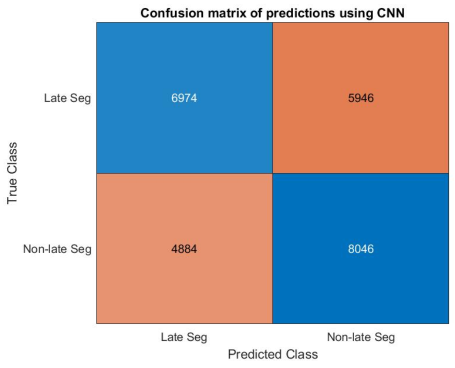
(a) SVM



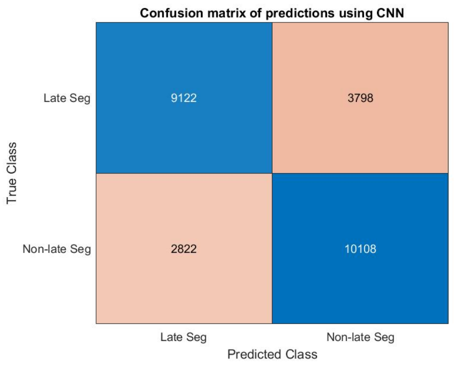
(b) k-NN



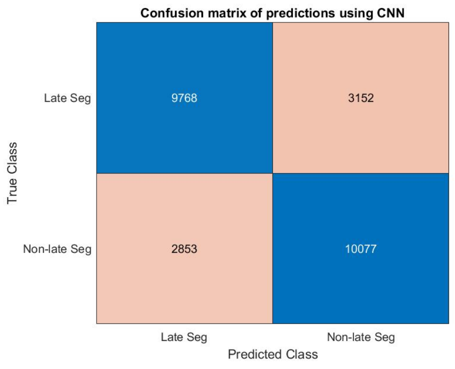
(c) DT



(d) CNN_s

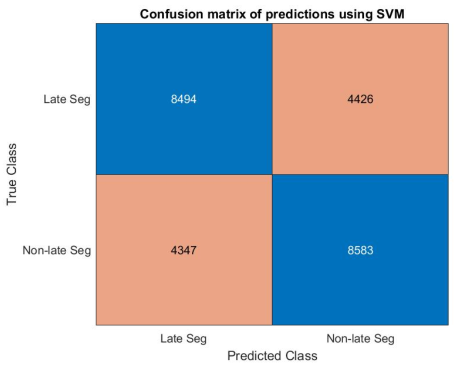


(e) CNN_m

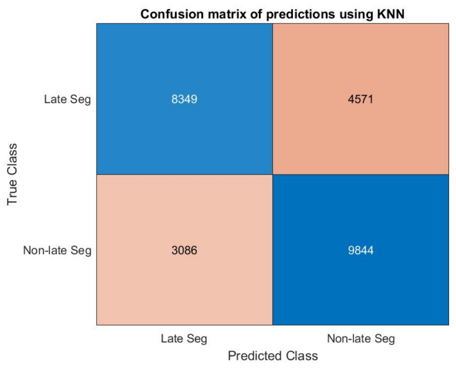


(f) CNN_l

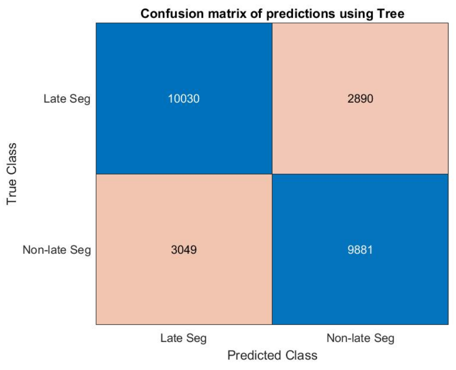
Figure 6.2: Confusion matrices of predictions using six different machine learning models for feature combination *TOCO, Peak Vector & FHR*. The predictions were made on test data, in total 2585 samples equally distributed between late and non-late segments.



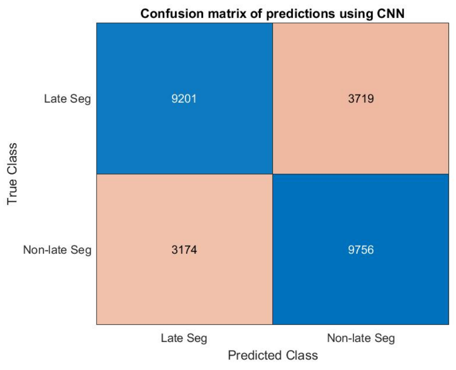
(a) SVM



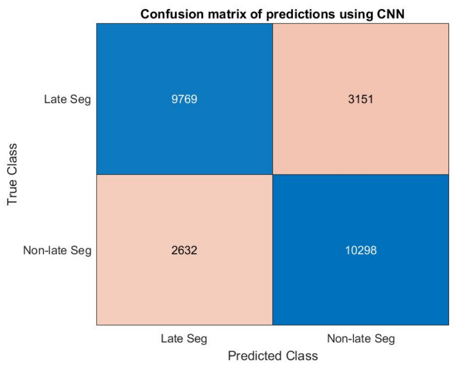
(b) k-NN



(c) DT

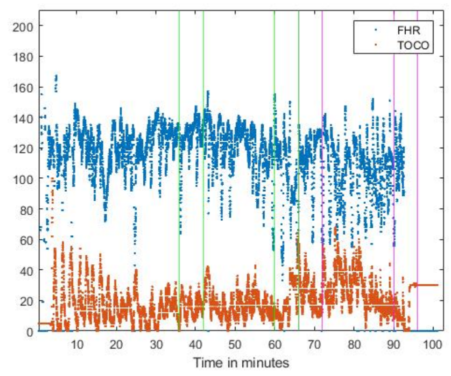


(d) CNN_m

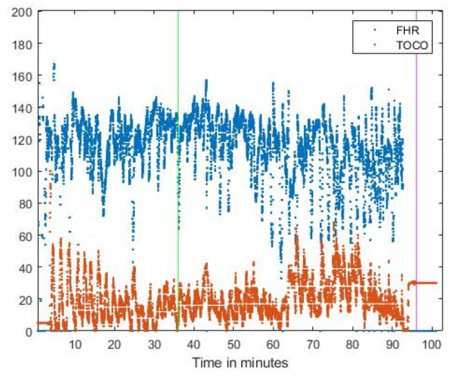


(e) CNN_l

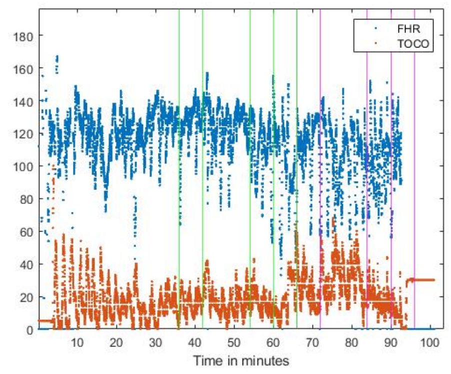
Figure 6.3: Confusion matrices of predictions using five different machine learning models for feature combination *TOCO, Peak Frequency (per hour), FHR, Parity, Smoking & Placental Abruption*. The predictions were made on test data, in total 2585 samples equally distributed between late and non-late segments.



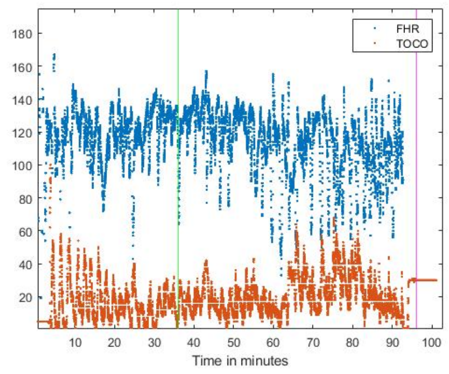
(a) Detection of late segments using the large CNN-model with features: *TOCO & FHR*.



(b) Detection of late segments, without overlap, using the large CNN-model with features: *TOCO & FHR*.

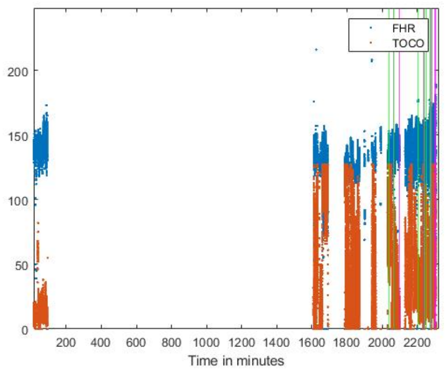


(c) Detection of late segments using the DT-model with features: *TOCO & FHR*.

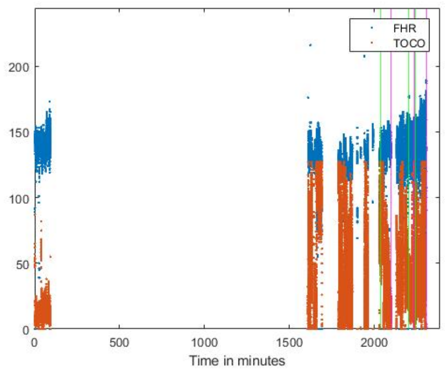


(d) Detection of late segments, without overlap, using the DT-model with features: *TOCO & FHR*.

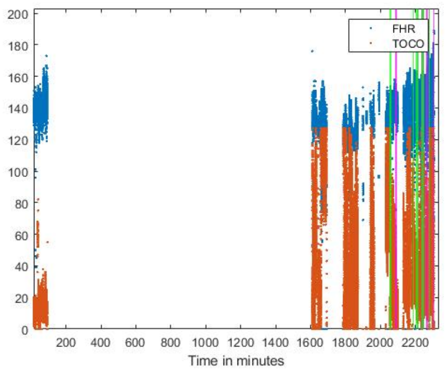
Figure 6.4: Identification of late segments, for mat file with id = 1, using the best CNN_l and DT models. Left-hand side shows all identified late segments. Right-hand shows the merged segments without any overlaps.



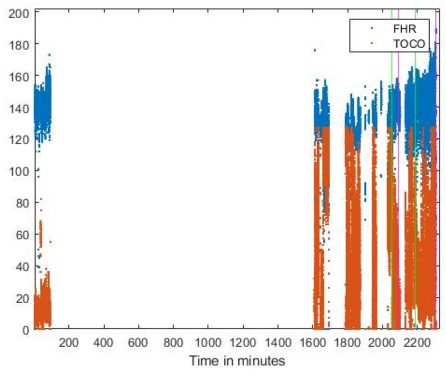
(a) Detection of late segments using the large CNN-model with features: *TOCO & FHR*.



(b) Detection of late segments, without overlap, using the large CNN-model with features: *TOCO & FHR*.

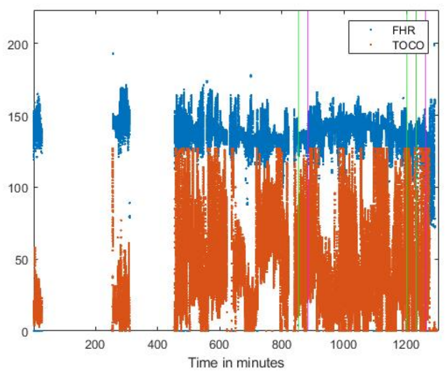


(c) Detection of late segments using the DT-model with features: *TOCO & FHR*.

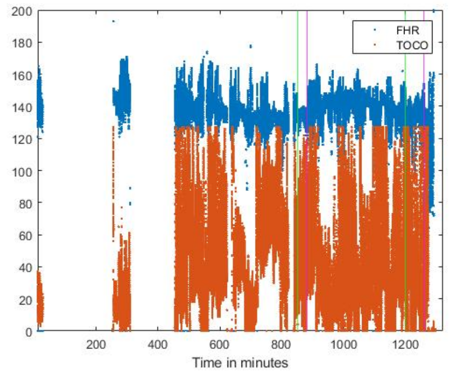


(d) Detection of late segments, without overlap, using the DT-model with features: *TOCO & FHR*.

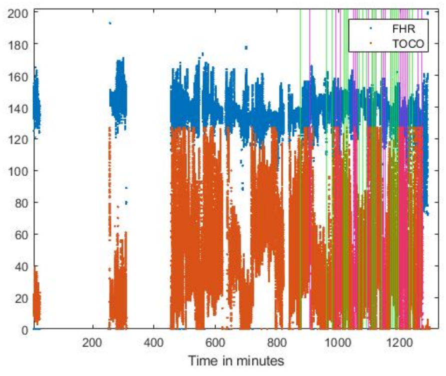
Figure 6.5: Identification of late segments, for mat file with id = 3, using the best CNN_l and DT models. The left-hand side show all identified late segments. The right-hand side show the merged segments without any overlaps.



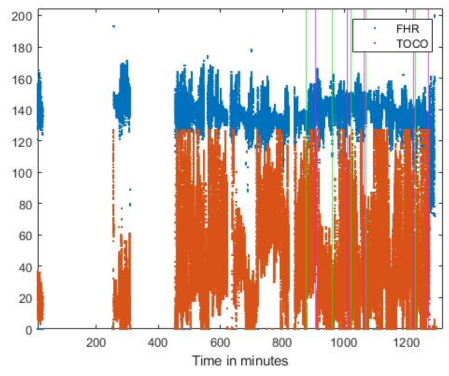
(a) Detection of late segments using the large CNN-model with features: *TOCO & FHR*.



(b) Detection of late segments, without overlap, using the large CNN-model with features: *TOCO & FHR*.

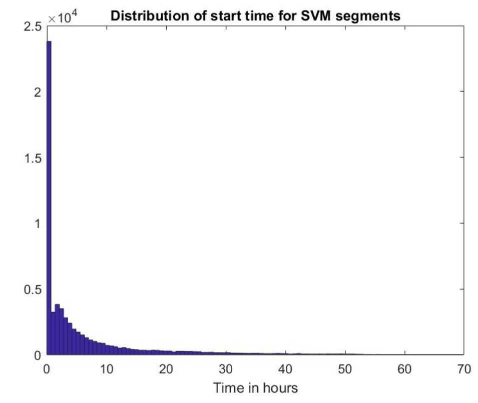


(c) Detection of late segments using the DT-model with features: *TOCO & FHR*.

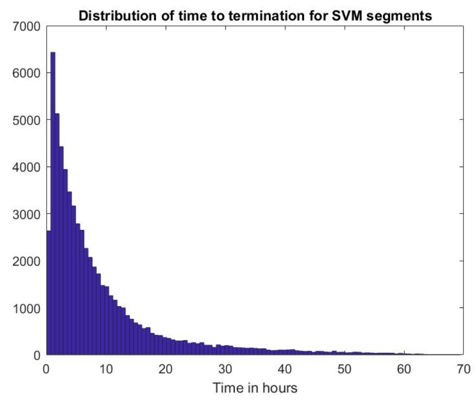


(d) Detection of late segments, without overlap, using the DT-model with features: *TOCO & FHR*.

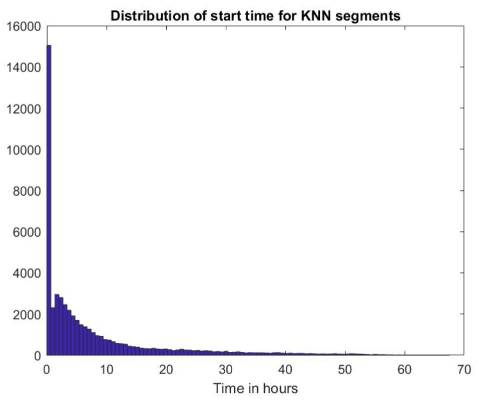
Figure 6.6: Identification of late segments, for mat file with id = 26, using the best CNN_l and DT models. The left-hand side show all identified late segments. The right-hand side show the merged segments without any overlaps.



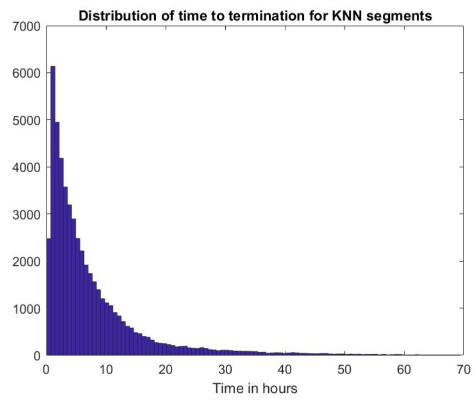
(a) Distribution of starting times for the SVM segments.



(b) Distribution of time to termination for the SVM segments.



(c) Distribution of starting times for the k-NN segments.



(d) Distribution of time to termination for the k-NN segments.

Figure 6.7: Visualization of the distribution of the starting times, and the time to termination for the frequency segments.

7. Classify the Apgar Score Given Labor Stage

In this chapter, the relationship between the Apgar score and uterine contractions in combination with the fetal heart rate is investigated. The aim is to investigate if there is a relationship between the CTG, clinical data, and the Apgar score, and if the machine learning algorithms can recognize any patterns. Furthermore, the relationship between CTG data and Apgar score is investigated for different stages of labor. Four different data sets are used for the classification of high and low Apgar scores: late segments, non-late segments, frequency-based segments, and mixed segments. The chapter contains a short explanation regarding the Apgar score as well as a joint method for all of the data sets. The segmentation and results of the different data sets are presented individually. Finally, the chapter contains a comparison between the results and a general discussion of the association between Apgar score and CTG data.

7.1 Apgar Score

After termination of labor, when the child is delivered, the physical condition of the newborn is assessed according to five components: heart rate, respiratory effort, muscle tone, reflex irritability, and color. Each component is given a value from zero to two. The assessment lays the foundation for the combined Apgar score, ranging from zero to ten. A higher score indicates good physical conditions, while a low score indicates poor overall status, and thus an increased risk of complications. Studies have shown that lower Apgar scores are associated with an increased relative risk of neonatal death, non-dependent on gestational age [5]. There are also demonstrated associations between cerebral palsy and low Apgar scores, where the adjusted risk ratio for cerebral palsy increases steadily with decreasing Apgar scores [49]. In Sweden, the Apgar score is evaluated after one, five, and ten minutes [50]. The Apgar score at five minutes can be evaluated according to the following definition: 10-7 reassuring, 6-4 moderately abnormal, and 3-0 low [51]. The evaluation of the infant determines the need for interventions in the delivery room. Accordingly, there is value in predicting the Apgar score of the fetus during the labor, and thus having the ability to intervene proactively.

7.2 Method

The same six machine learning algorithms that were used in the previous chapters were used for all data sets in this chapter. The input data consisted of both vectors and single constants from the CTG and clinical data. The output was a binary classification where 0 equaled a high Apgar score, and 1 corresponded to a low Apgar score. A negative prediction (0) implied classifying the sample as a reassuring, i.e., with an Apgar score equal to or higher than seven. A positive prediction (1) equaled classifying a sample as a abnormal or low, i.e., with an Apgar score below seven. Two different combinations of features were tested. The base case contained the TOCO and FHR signals, while the other combination consisted of the TOCO, FHR, peak frequency, and all clinical parameters known before the delivery, i.e., parity, smoking, placental abruption, preeclampsia, diabetes, and gestational age.

7.3 Late Segments from Machine Learning Algorithms

7.3.1 Segmentation

The late segments from chapter 6.2, *Classify Segments Using Machine Learning*, is only a subset of all of the usable late segments in the whole data set. In this part, there was a need for more late segments in order to assemble a data set that contained enough samples with low Apgar scores. Since 1.65 percent of the newborns, in the data set, has a low Apgar score at five minutes, the conclusion was that all possible segments which met the requirements were needed to yield more robust results. The requirements for being a late segment were set to the same as in the previous chapter.

From the whole data set of 98 328 samples, 27 903 late segments were extracted, where 435 of them had a low Apgar score. The data were divided into a test and a training set, where the test set consisted of 20 percent of the files and the training set the remaining 80 percent. By using the Matlab-function *CVpartition* it was ensured that each data set had a distribution of good and bad outcomes according to the true distribution. The test set had the true distribution of good and bad outcomes, whereas the training set was downsized until it was equally balanced between high and low Apgar scores. The distributions for the sets are found in table 7.1.

Table 7.1: The distribution between high and low Apgar score, as well as the total number of samples in the training and test set.

	Low Apgar	High Apgar	Total
Training Set	348	348	696
Test Set	87	5494	5581

7.3.2 Results

Table 7.2 and figure 7.1 present the results of the base case. The k-NN model presents the highest sensitivity of 64.48% together with a low standard deviation of less than 3%. The large CNN model has the highest specificity, 60.03%, accuracy, 59.95%, and F1 score, 4.10%, with a low standard deviation, below 2%, for all performance measures. The SVM has lower results than the others for all measures. In addition, the model presents substantially higher standard deviation for all performance measures, except for the F1 score. The F1 scores are generally very low. This is partly explained by the large imbalance in the test set, where only 1.55 percent of the data consists of samples with low Apgar scores.

Table 7.2: Results for the six machine learning models for the late segments with the feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.5080 ±0.1273	0.4761 ±0.1181	0.4766 ±0.1144	0.0292 ±0.0031
k-NN	0.6448 ±0.0294	0.5224 ±0.0227	0.5243 ±0.0222	0.0406 ±0.0022
DT	0.6023 ±0.0731	0.5280 ±0.0339	0.5292 ±0.0327	0.0384 ±0.0039
CNN_s	0.6057 ±0.0778	0.4842 ±0.0406	0.4861 ±0.0390	0.0354 ±0.0025
CNN_m	0.5575 ±0.0520	0.5586 ±0.0334	0.5586 ±0.0325	0.0379 ±0.0030
CNN_l	0.5483 ±0.0236	0.6003 ±0.0179	0.5995 ±0.0174	0.0410 ±0.0017

Table 7.3 and figure 7.2 present the results of the larger feature combination. The SVM model has a sensitivity of 42.76% while all the others exceed 50%. The highest sensitivity, 69.89%, is achieved by the DT model. This model also presents both specificity and accuracy above 60%. The F1 score for the DT model is 5.26%, which is the highest among all models. The CNN models perform rather similar to each other, all with a sensitivity of around 55%. The large CNN model yields a specificity and accuracy of 60.94% and 60.84%, while the small and medium CNN models perform approximately 55% for both measures. For the F1 score, the small and medium CNN models yield 3.73% and the large has a slightly higher score, 4.15%. Concerning the standard deviations, the SVM generally presents the highest figures, while the other models have more similar values.

Table 7.3: Results for the six machine learning models for the late segments with the feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes, Peak Frequency (per hour) & Gestational Age*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.4276 $\pm$0.0940	0.5413 $\pm$0.0712	0.5395 $\pm$0.0686	0.0280 $\pm$0.0021
k-NN	0.6437 $\pm$0.0330	0.5265 $\pm$0.0243	0.5283 $\pm$0.0236	0.0408 $\pm$0.0019
DT	0.6989 $\pm$0.0519	0.6052 $\pm$0.0173	0.6066 $\pm$0.0170	0.0526 $\pm$0.0043
CNN_s	0.5667 $\pm$0.0659	0.5410 $\pm$0.0662	0.5414 $\pm$0.0644	0.0373 $\pm$0.0031
CNN_m	0.5529 $\pm$0.0510	0.5559 $\pm$0.0273	0.5558 $\pm$0.0262	0.0373 $\pm$0.0022
CNN_l	0.5437 $\pm$0.0339	0.6094 $\pm$0.0139	0.6084 $\pm$0.0136	0.0415 $\pm$0.0025

The differences between the base case and the larger feature combination are small. All of the models, except the SVM, increase the sensitivity when extending the input to the larger combination. The sensitivity of the SVM model decreases from 50.80% to 42.76%. Regarding the specificity and accuracy, all models except the medium CNN, have lower results for the base case. The F1 score is higher for four of the models, the k-NN, DT, CNN_s and CNN_l when the large feature combination is used. Especially the result of the DT increases from 3.86% to 5.26%. The majority of the results do improve for the larger feature combination. However, they are just marginally better, and still very unsatisfactory.

7.4 Non-Late Segments from Machine Learning Algorithms

7.4.1 Segmentation

The same non-late segments that were used in chapter 6.2, *Classify Segments Using Machine Learning*, were used for this part. The reason for not extracting more segments, as were made for the late segments in the previous section, was that the number of non-late segments was already approximately equal to the extended number of late segments. As in previous cases, the data were split into test and training sets, where samples with low Apgar scores were dispersed between the sets according to the true distribution. The training set was downsized until it was equally distributed between samples with high and low Agar scores. The distribution for the sets are found in table 7.4.

Table 7.4: The distribution between high and low Apgar score, as well as the total number of samples in the training and test set.

	Low Apgar	High Apgar	Total
Training Set	368	368	736
Test Set	92	4909	5001

7.4.2 Results

Table 7.5 and figure 7.3 displays the results for the base case, with TOCO and FHR as input, for the non-late segments. The results from the large feature combination are presented in table 7.6 and figure 7.4.

As seen from table 7.5 the results of the SVM model deviate from the rest of the models. The sensitivity, at 43.04%, is the lowest result among the models, while the specificity is the highest, at 56.99%. The corresponding standard deviations are strikingly high, at 30.91% and 30.03%, respectively. The DT model has the highest F1 score of 3.67%. However, the difference between the models in this setup is very small regarding the F1 score, the lowest result is 3.41% and it is generated by the SVM model.

Table 7.5: Results for the six machine learning models for the non-late segments with the feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.4304 ±0.3091	0.5699 ±0.3003	0.5673 ±0.2892	0.0341 ±0.0060
k-NN	0.5696 ±0.0295	0.4301 ±0.0230	0.4327 ±0.0224	0.0356 ±0.0018
DT	0.4924 ±0.0602	0.5246 ±0.0194	0.5240 ±0.0190	0.0367 ±0.0046
CNN_s	0.5163 ±0.0643	0.4763 ±0.0588	0.4771 ±0.0569	0.0351 ±0.0030
CNN_m	0.4739 ±0.0686	0.5094 ±0.0741	0.5087 ±0.0715	0.0343 ±0.0017
CNN_l	0.4913 ±0.0520	0.5045 ±0.0297	0.5042 ±0.0287	0.0352 ±0.0032

All models except the medium and large CNN improve their F1 scores in this setup compared to the base case, see table 7.6. The DT has the highest F1 score, 5.13%, with a standard deviation of 0.4%. The SVM changes from having lower sensitivity and higher specificity to the opposite for this feature combination. However, the high standard deviations, above 30%, indicate that the model is overfitted and has lost its capability to make generalizations. The k-NN and DT models improve both the sensitivity and the specificity for the larger feature combination. However, the k-NN is only improving marginally. The sensitivity increases from 56.96% to 58.59%, while the corresponding standard deviation reduces from 2.95% to 2.58%. The specificity is almost constant, 43.01% for the base case and 43.21% for the large case, while the standard deviation increases, 3.38% compared to the previous 2.30%. The improvement in the DT model is more substantial. The sensitivity increases from 49.24% for the base case to 62.17%, while the associated standard deviation reduces from 6.02% to 3.65%. The specificity for this feature combination is 57.48% compared to the previous result of 52.46%. However, the standard deviation does increase simultaneously, from 1.94% for the base case to 3.20% for the large case.

Table 7.6: Results for the six machine learning models for the non-late segments with the feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes, Peak Frequency (per hour) & Gestational Age*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6685 ±0.3320	0.3456 ±0.3123	0.3515 ±0.3005	0.0356 ±0.0040
k-NN	0.5859 ±0.0258	0.4321 ±0.0338	0.4349 ±0.0329	0.0368 ±0.0021
DT	0.6217 ±0.0365	0.5748 ±0.0320	0.5757 ±0.0312	0.0513 ±0.0040
CNN_s	0.6152 ±0.0925	0.4136 ±0.0755	0.4173 ±0.0726	0.0373 ±0.0025
CNN_m	0.4380 ±0.0464	0.5229 ±0.0283	0.5213 ±0.0272	0.0325 ±0.0025
CNN_l	0.4696 ±0.0711	0.5208 ±0.0402	0.5199 ±0.0384	0.0346 ±0.0034

7.5 Frequency-based Segments

7.5.1 Segmentation

The frequency segments from chapter 6.3, *Finding Segments with Frequency*, were divided into a training and a test set. The test set contained 20 percent of the segments and was saved according to the original distribution, while the training set was downsized until balanced. The sizes of each of the sets are presented in table 7.7.

Table 7.7: The number of samples in the training and test sets, using the SVM and k-NN for segment extraction and the distribution between low and high Apgar scores.

	SVM			k-NN		
	Low Apgar	High Apgar	Total	Low Apgar	High Apgar	Total
Training set	755	755	1510	575	575	1150
Test set	182	12415	12597	158	10376	10534

7.5.2 Results

The base case, where TOCO and FHR are used as input, is tested for both data sets. This combination is tested for the k-NN, SVM, DT, and the small, medium and large CNN. The results for the k-NN segments are presented in table 7.8 and in the confusion matrices in figure 7.5, and the corresponding results for the SVM segments are found in table 7.9 and figure 7.6.

Table 7.8 shows that the SVM model generates the highest sensitivity with 67.66%. However, the corresponding standard deviation, 13.21%, is also substantially higher than the others. The specificity is lower than the sensitivity in general, meaning that all models are better at identifying the true positives compared to the true negatives. The accuracy is approximately 50% for all models, except for the SVM which only has 34.13%. The large CNN model has the highest F1 score with 3.25%, while the small CNN model has the lowest with 2.82%. The F1 score is generally extremely low.

As seen from table 7.9, the results for the SVM segments are quite similar to those for the k-NN segments. The accuracy for all models, except the SVM model, is around 50% for both data sets. The SVM model deviates from the others concerning sensitivity, specificity, and accuracy. The sensitivity for the SVM model is 67.66% and 78.19% for the k-NN and

Table 7.8: Results for the six machine learning models for the k-NN segments with the feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6766 ±0.1321	0.3362 ±0.1414	0.3413 ±0.1373	0.0300 ±0.0012
k-NN	0.5449 ±0.0167	0.4908 ±0.0208	0.4916 ±0.0204	0.0312 ±0.0012
DT	0.5297 ±0.0531	0.5059 ±0.0306	0.5063 ±0.0298	0.0312 ±0.0030
CNN_s	0.4975 ±0.0362	0.4855 ±0.0268	0.4857 ±0.0260	0.0282 ±0.0011
CNN_m	0.5165 ±0.0520	0.5129 ±0.0393	0.5130 ±0.0380	0.0308 ±0.0016
CNN_l	0.5513 ±0.0356	0.5074 ±0.0137	0.5081 ±0.0133	0.0325 ±0.0019

SVM segments, respectively. This is substantially higher compared to the other models. The specificity is very low for the SVM model, scoring 33.62% and 21.31% for the two data sets. The low specificity affects the corresponding accuracy negatively, resulting in 34.13% for the k-NN segments and 22.14% for the SVM segments. The associated standard deviations are obtrusively high, 13.73% and 14.53%, respectively. The k-NN model has a sensitivity below 50%, 46.65%, for the SVM frequency-based segments, while the specificity is just above 50%. For both these measures, the standard deviation is 1.96%. The F1 score, for both the k-NN and SVM models, is 2.63% with a standard deviation of 0.01%. The results for the three first measures of the DT model are quite similar to each other, the sensitivity is highest, at 54.40%, while the specificity and accuracy are very close to 50%. The F1 score is 3.07% with a standard deviation of 0.29%.

Among the CNN models, the medium setup has the highest sensitivity, at 55.88%, while the small and large model has 52.47% and 50.11%, respectively. For the specificity, the medium CNN instead presents the lowest score among the CNN models, at 47.16%, while the small model has 48.08% and the large 52.52%. The large CNN is the only model that attains an accuracy above 50%, landing at 52.49%. The small and medium model presents an accuracy of 48.14% and 47.29%, respectively. The F1 scores are 2.84%, 2.97%, and 2.95% for the small, medium, and large models, respectively. The F1 scores for the medium and large CNN models are slightly lower for the SVM segments compared to the k-NN segments, while the opposite applies for the small CNN model.

Table 7.9: Results for the six machine learning models for the SVM segments with the feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.7819 ±0.1339	0.2131 ±0.1494	0.2214 ±0.1453	0.0263 ±0.0001
k-NN	0.4665 ±0.0196	0.5005 ±0.0196	0.5000 ±0.0179	0.0263 ±0.0001
DT	0.5440 ±0.0565	0.5027 ±0.0114	0.5032 ±0.0110	0.0307 ±0.0029
CNN_s	0.5247 ±0.0317	0.4808 ±0.0167	0.4814 ±0.0162	0.0284 ±0.0014
CNN_m	0.5588 ±0.0579	0.4716 ±0.0433	0.4729 ±0.0419	0.0297 ±0.0013
CNN_l	0.5011 ±0.0679	0.5252 ±0.0330	0.5249 ±0.0317	0.0295 ±0.0028

Due to the very small differences in results between the two data sets, the decision is made to only continue the analysis for one of the sets. Since the SVM data set contains a larger amount of files, this set is used for continued analysis.

The results of the large feature combination are presented in table 7.10 and figure 7.7. The results are quite similar to the base case with FHR and TCO as input. Once again, the SVM model differs from the rest. The results from the DT model are slightly better than before, with 59.89% sensitivity compared to the previous 54.40%. Both specificity and accuracy increase from around 50% for the base case to approximately 54% for the large feature combination. The F1 score is still very low, but it has increased from 3.07% to 3.66%. The difference between the various CNN models is limited. The medium CNN model has the highest specificity, 52.29%, as well as the highest F1 score with 3.07%. Both measures increase compared to the base case, where the specificity is 47.16% and the F1 score is 2.95%. The sensitivity decreases from 55.88% to 43.80%, implying that the model from the small feature combination is better at predicting low Apgar scores.

Table 7.10: Results for the six machine learning models for the SVM segments with the feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes, Peak Frequency (per hour) & Gestation Age.* The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.7489 ±0.2384	0.2431 ±0.2432	0.2504 ±0.2363	0.0279 ±0.0017
k-NN	0.4996 ±0.0164	0.4665 ±0.0169	0.4996 ±0.0164	0.0262 ±0.0011
DT	0.5989 ±0.0468	0.5430 ±0.0109	0.5438 ±0.0109	0.0366 ±0.0031
CNN_s	0.5110 ±0.0299	0.4951 ±0.0137	0.4953 ±0.0133	0.0284 ±0.0015
CNN_m	0.4380 ±0.0464	0.5229 ±0.0283	0.5213 ±0.0272	0.0325 ±0.0025
CNN_l	0.5390 ±0.0424	0.5074 ±0.0461	0.5078 ±0.0449	0.0307 ±0.0016

7.6 Independent of Stage

7.6.1 Segmentation

The mixed data set was assembled with files from all previous data sets, i.e., late, non-late, k-NN, and SVM segments. The training set consisted of 1200 files equally divided between these four categories and balanced between samples with high and low Apgar scores. The test set contained 6000 files, 1500 from each of the four categories. The test set included 93 samples with a low Apgar score, which constitutes 1.55%. This ratio is in line with the previous distributions, e.g. 1.55% for the late segments, 1.84% for the non-late, 1.5% for the k-NN, 1.44% for the SVM segments, and 1.65% for the whole data set.

7.6.2 Results

Table 7.11 and figure 7.8 present the results of the base case, with FHR and TOCO as input, for the mixed segments. All models present a sensitivity above 50%, where the k-NN has the highest result of 64.84%. The corresponding standard deviations vary, from 3.34% for the small CNN model to 7.97% for the SVM. The specificity is lower than the sensitivity for all models. The large CNN model has a specificity of 50.35% and a sensitivity of 50.43%. The lowest specificity is generated by the SVM model, at 40.01%, with a standard deviation of 7.20%. However, the specificity for the k-NN model is just slightly higher, at 41.32%, and with a standard deviation of 2.15%. The SVM model presents the lowest accuracy and F1 score among all models, at 40.34% and 3.09%, respectively. It is only the DT and the large CNN models that attain an accuracy above 50%, with 50.56% and 50.35%, respectively. The k-NN and DT models have the highest F1 score, 3.33% (0.13) and 3.32% (0.31), respectively.

Table 7.11: Results for the six machine learning models for the mixed segments with the feature combination *TOCO & FHR*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6129 ±0.0797	0.4001 ±0.0720	0.4034 ±0.0698	0.0309 ±0.0019
k-NN	0.6484 ±0.0409	0.4132 ±0.0215	0.4168 ±0.0206	0.0333 ±0.0013
DT	0.5473 ±0.0540	0.5049 ±0.0190	0.5056 ±0.0185	0.0332 ±0.0031
CNN_s	0.5366 ±0.0334	0.4890 ±0.0247	0.4898 ±0.0240	0.0316 ±0.0014
CNN_m	0.5538 ±0.0376	0.4845 ±0.0228	0.4856 ±0.0232	0.0316 ±0.0014
CNN_l	0.5043 ±0.0503	0.5035 ±0.0262	0.5035 ±0.0253	0.0305 ±0.0025

The results of the large feature combination are presented in table 7.12 and figure 7.9. All models except the large CNN model present a sensitivity above 50%, where the SVM and k-NN exceed 60%, with 62.04% and 64.19%, respectively. The SVM stands out by its corresponding standard deviation of 12.69%, whereas the others fall short of 7%. For the specificity, the SVM and k-NN models have the lowest results, landing at 40.64% and 41.71%. The DT model has the highest specificity and accuracy, 56.22% and 56.26%, respectively, while the SVM model has the lowest accuracy, 40.98%. The F1 scores extend from 3.12% for the large CNN model, to 3.99% for the DT model.

Comparing the results from the base case with the larger feature combination, there are, yet again, small differences. The F1 score from the k-NN model remains constant, at 3.33%, when the feature combination is extended. The corresponding value for the DT model increases from 3.32% to 3.99%, whereas the other models only present small improvements for the measure. However, no model performs worse with regards to the F1 score, for the large feature combination. The specificity is in general lower for the base case, meaning that the larger feature combination leads to models that are better at identifying the negative outcomes, i.e., the high Apgar scores.

Table 7.12: Results for the six machine learning models for the mixed segments with the feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Peak Frequency (per hour) & Gestation Age*. The sensitivity, specificity, accuracy and F1 score are presented, with mean and standard deviation from the 10-fold cross-validation.

	Sensitivity	Specificity	Accuracy	F1 score
SVM	0.6204 ±0.1269	0.4064 ±0.1425	0.4098 ±0.1384	0.0318 ±0.0024
k-NN	0.6419 ±0.0259	0.4179 ±0.0181	0.4214 ±0.0177	0.0333 ±0.0013
DT	0.5871 ±0.0460	0.5622 ±0.0128	0.5626 ±0.0123	0.0399 ±0.0027
CNN_s	0.5161 ±0.0512	0.5129 ±0.0337	0.5129 ±0.0329	0.0319 ±0.0032
CNN_m	0.5430 ±0.0544	0.4990 ±0.0563	0.4997 ±0.0547	0.0326 ±0.0015
CNN_l	0.4946 ±0.0598	0.5248 ±0.0233	0.5243 ±0.0225	0.0312 ±0.0031

7.7 Discussion

The results from this chapter were discouraging. No model performed considerably better than the fortuity, and the differences between the various segments were small. The best performing model, with regards to the F1 score, was the DT for the late segments with the larger feature combination which resulted in 5.26%. In general, the F1 scores oscillated around approximately 3%, and the rest of the performance measures were around 50-60%.

These results were not satisfying, and other methods, machine learning algorithms, setups, and/or features need to be tested in order to improve the performance. Table 7.13, contains a compilation of the best F1 score for each model with the corresponding setup, i.e., data set and feature combination. All models, except the SVM and small CNN, had the highest F1 scores for the late segments. Only the medium CNN model performed better with the basic feature combination. However, the difference between the base case and the larger feature combination for the same data set was just 0.06%. The SVM and medium CNN performed best for the non-late segments. None of the models had the highest F1 score for the frequency-based or mixed segments. There are numerous possible explanations for this outcome. Building on the assumption that there are considerable differences in the data between different stages of labor, it should not be surprising that the mixed segments yield poorer outcomes. It has earlier been concluded that the frequency-based segments differ substantially in the start and remaining time. This could indicate that they might not represent the same stage of labor, and thus might be harder for the machine learning algorithms to make predictions on. However, it might also be mostly coincidental since the results did not deviate that much from each other. There are too many sources of errors to be able to draw clear conclusions from the results in this chapter.

Table 7.13: Compilation of the best F1 scores for the six machine learning algorithms. For each model, the corresponding data set and feature combination are presented.

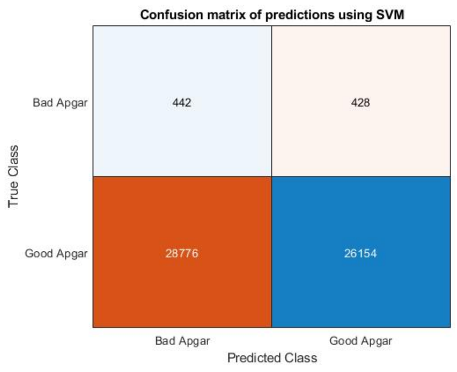
	Data Set	Feature Combination	F1 Score
SVM	Non-late	All Features	0.0356 ±0.0040
k-NN	Late	All Features	0.0408 ±0.0019
DT	Late	All Features	0.0526 ±0.0043
CNN_s	Non-late	All Features	0.0373 ±0.0025
CNN_m	Late	FHR & TOCO	0.0379 ±0.0030
CNN_l	Late	All Features	0.0415 ±0.0025

There are numerous explanations for the unsatisfactory results. The sets of training data were limited. The largest set consisted of the SVM segments, and contained 1510 samples compared to more than 10 000 files for the classification of late and non-late segments in the previous chapter. In general, machine learning algorithms depend on, and benefit from, large amounts of data. Even though the initial data set contains 98 238 samples, a large part of it does not consist of data of high enough quality or quantity to generate useful segments. In total, there are 1621 samples with low Apgar scores. Consequently, the largest possible, balanced data set could contain 3242 samples in theory. However, it is highly unlikely that all samples with low Apgar scores contain useful data that meet the requirements, meaning that the true limit is lower than the theoretical. For the data sets used in this chapter, there are additional requirements on the data, e.g. frequency of uterine contractions or that the last viable segment in a file is long enough, which further reduces the data sets. Larger data sets reduce the risk of overfitting. The problems derived from models with overfitting properties are explained thoroughly in section 3.7, as well as possible courses of action.

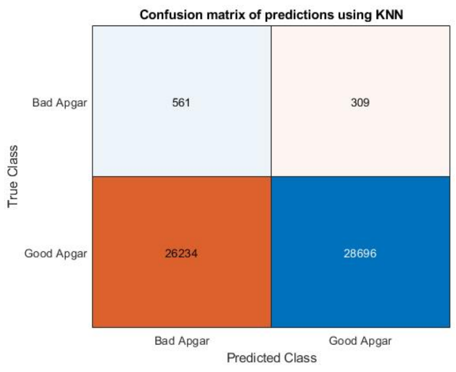
One course of action is augmentation of the data, for example by adding noise or including multiple, overlapping segments from the same CTG file. However, the latter would not have been possible to do for the late and frequency-based segments with the current setup. The late segments were extracted from the last 30-minutes of the files, and the frequency segments contained the first 30 minutes after a certain contraction frequency was achieved. If multiple segments were to be extracted, the criteria for both of these categories would need to be altered. For example, it would have been possible to extract overlapping segments from the last hour of each CTG file instead of just one segment from the last half hour. Depending on the size of the overlap as well as the length of the last segment, it would be possible to extract numerous segments instead of just one. For the frequency segments, one possibility could be to allow all segments with uterine contraction of a certain frequency in the data set. The allowed frequency could be set to an interval, to extend the possibilities further.

The comparison between the different data sets is hard to evaluate since the results did not deviate substantially from each other. The data was split for two reasons. Firstly, the assumption was that by assembling data sets within the same stage, the extracted segments would have more similarities, and thus improve the conditions for the machine learning algorithms. The other reason was that the different sets might contain specific, stage-dependent structures, which could be variously beneficial for the machine learning models to predict from. The small differences in results between the different data sets could depend on numerous reasons. It is possible that the structures in the data do not vary significantly between the stages, or at least, that there are no clear patterns from which the machine learning algorithms could predict the outcome. Another possible explanation is that the stages might not be as separated as intended. As previously mentioned, the start and remaining times for the frequency-based segments vary enormously. Even though time in itself is not a great indicator, since the courses of the deliveries vary considerably, it is still questionable whether these segments come from the same stage of labor. The late segments are extracted manually from the last half hour of the files. These segments should be more comparable, however, since there is no indication of when the baby is born, there is no guarantee of how late in the delivery these segments are extracted. The non-late segments are taken randomly from the whole file except for the last 30 minutes. Therefore, there ought to be quite considerable differences between these segments.

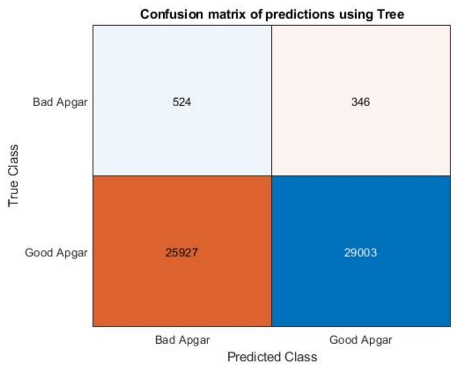
Another source of error is the TOCO measurements. The value of these is dependent on external factors, such as the tightness of the equipment and the thickness of the mother's abdomen wall. These two aspects affect the outcome of the measurements, and since the measuring equipment is taken on and off multiple times during the delivery, the values might differ substantially within one CTG file. In the previous chapter, an attempt was made to normalize the TOCO signal by dividing each number by the standard deviation of the whole segment. However, this did not lead to satisfying results and was thus not made for the case with the Apgar score as output. Ideally, the TOCO measurements would have been made with an intrauterine pressure catheter, to enable truly comparable measurements.



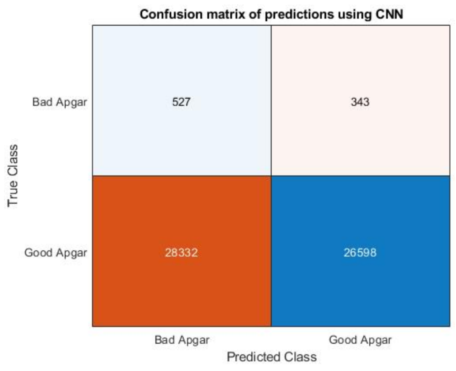
(a) SVM



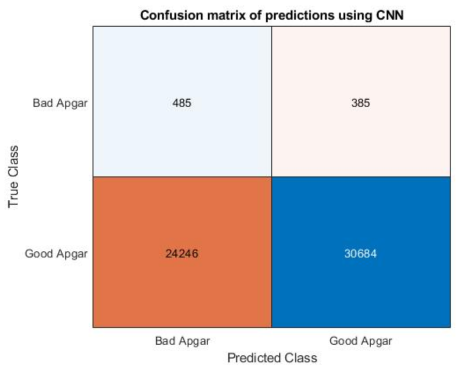
(b) k-NN



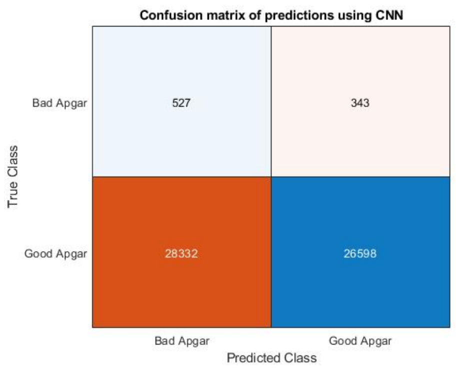
(c) DT



(d) CNN_s

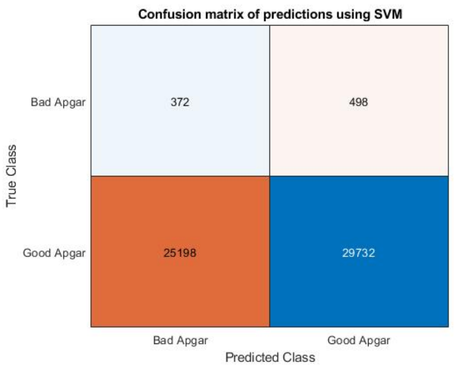


(e) CNN_m

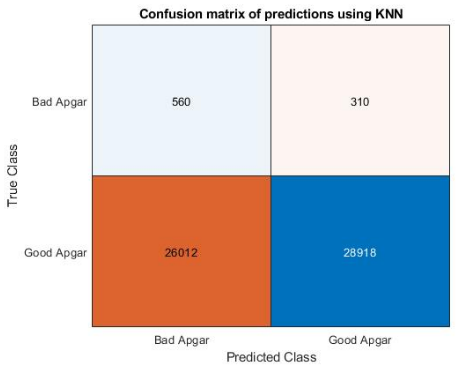


(f) CNN_l

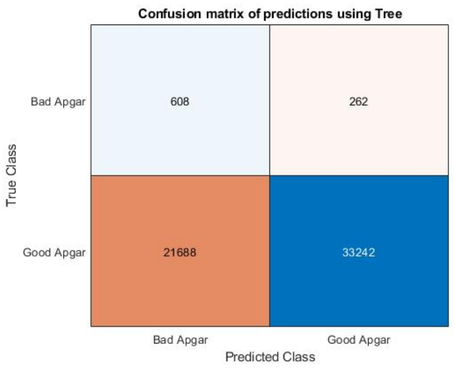
Figure 7.1: Confusion matrices of predictions using six different machine learning models for the late segments with feature combination *TOCO & FHR*. The predictions were made on test data with 5581 samples.



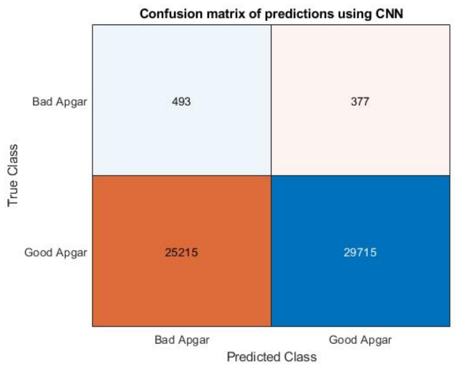
(a) SVM



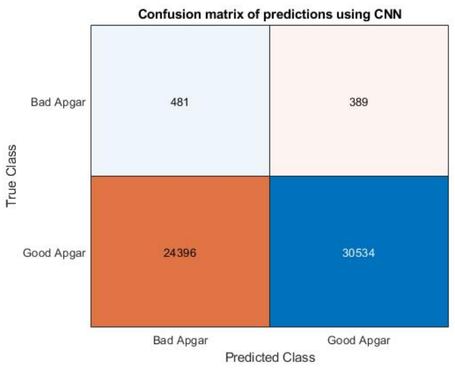
(b) k-NN



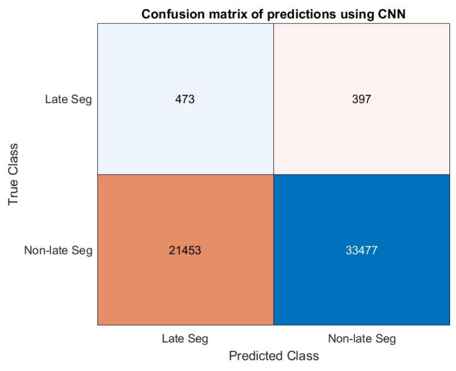
(c) DT



(d) CNN_s

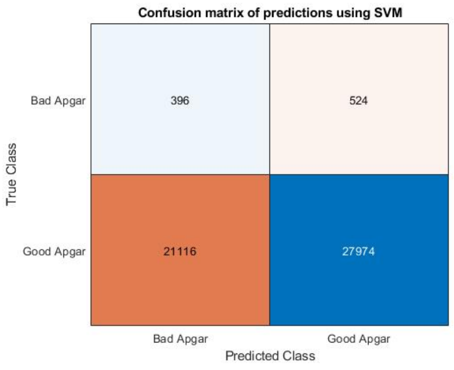


(e) CNN_m

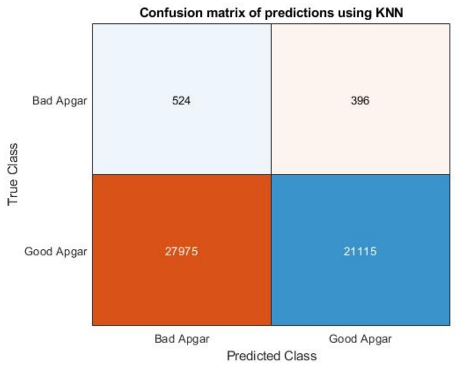


(f) CNN_l

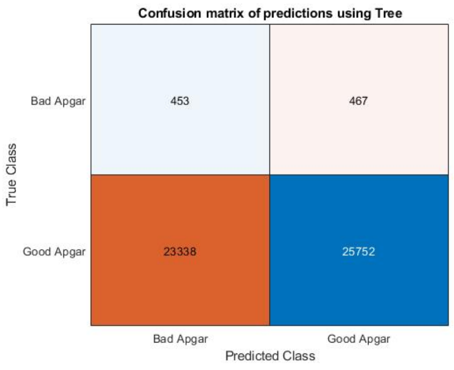
Figure 7.2: Confusion matrices of predictions using six different machine learning models for the late segments with feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes, Peak Frequency (per hour) & Gestational Age.* The predictions were made on test data with 5581 samples.



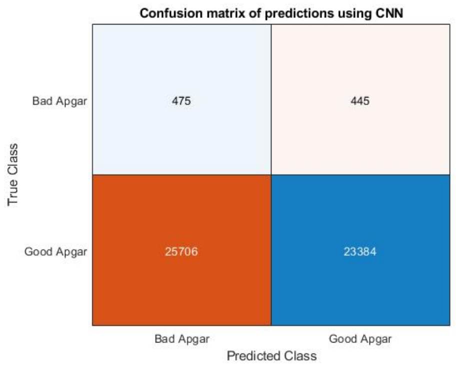
(a) SVM



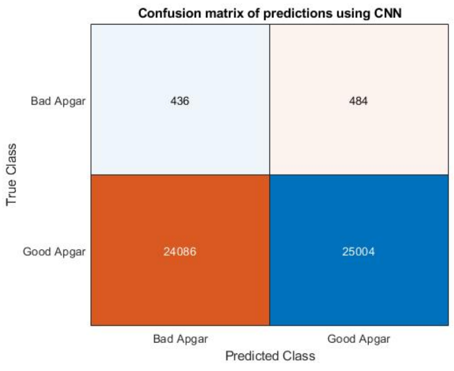
(b) k-NN



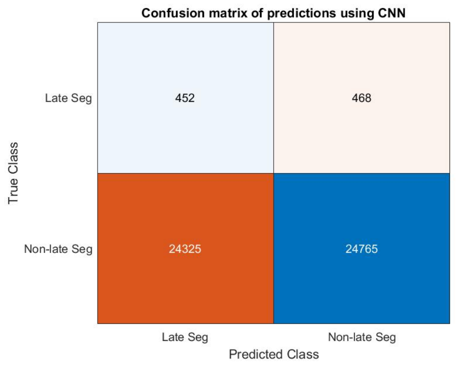
(c) DT



(d) CNN_s

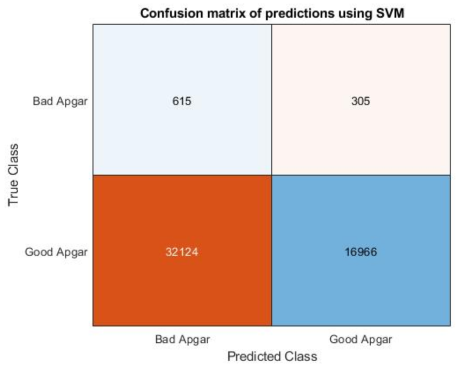


(e) CNN_m

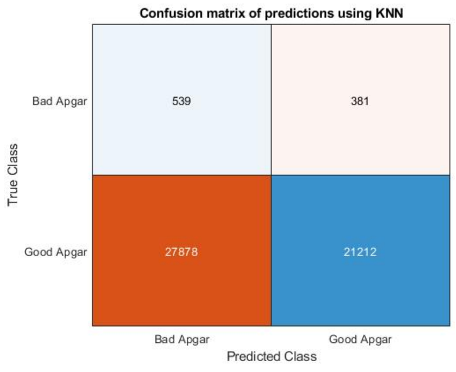


(f) CNN_l

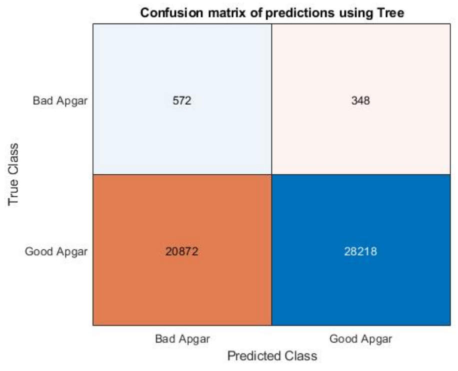
Figure 7.3: Confusion matrices of predictions using six different machine learning models for the non-late segments with feature combination *TOCO & FHR*. The predictions were made on test data with 5001 samples.



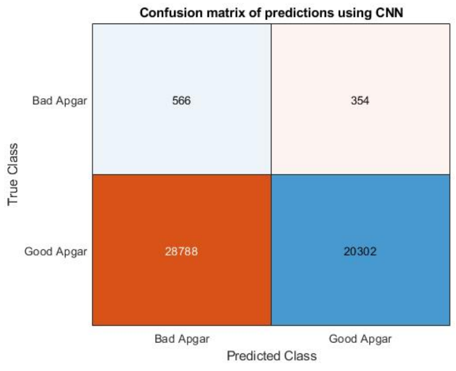
(a) SVM



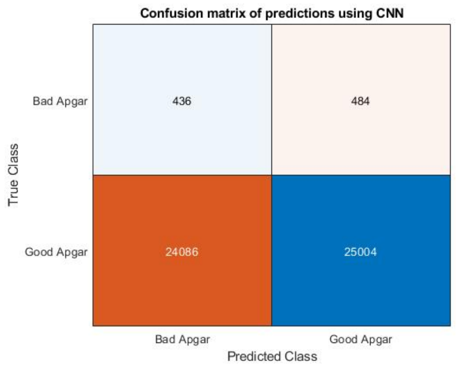
(b) k-NN



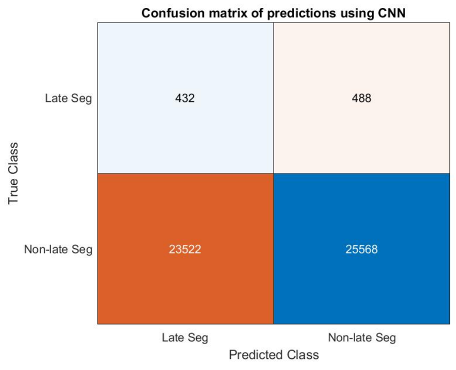
(c) DT



(d) CNN_s

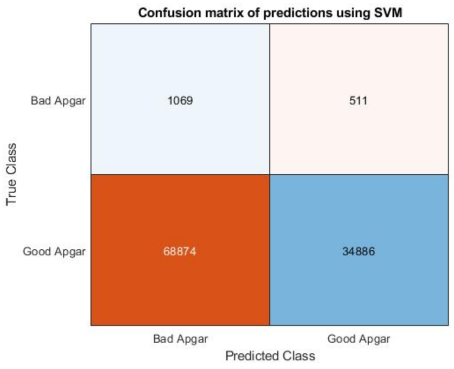


(e) CNN_m

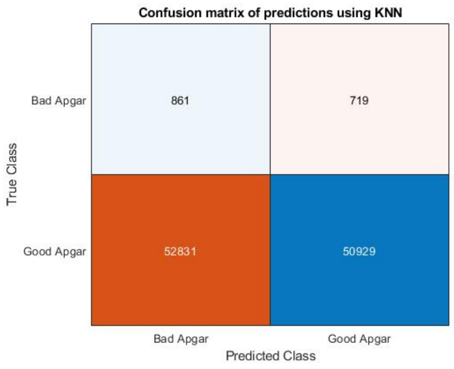


(f) CNN_l

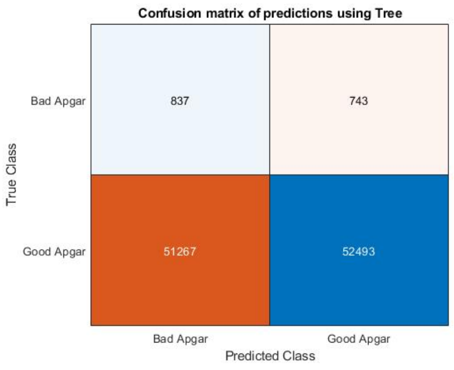
Figure 7.4: Confusion matrices of predictions using six different machine learning models for the non-late segments with feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Diabetes, Peak Frequency (per hour) & Gestational Age.* The predictions were made on test data with 5001 samples.



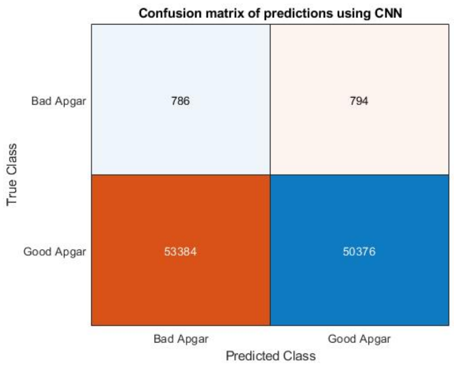
(a) SVM



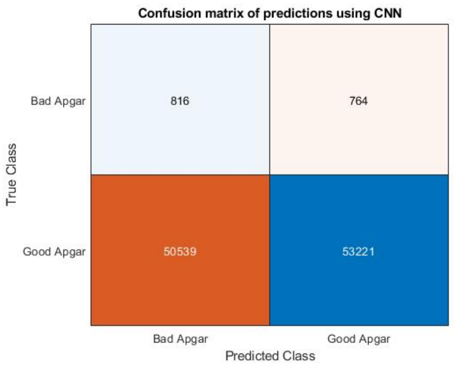
(b) k-NN



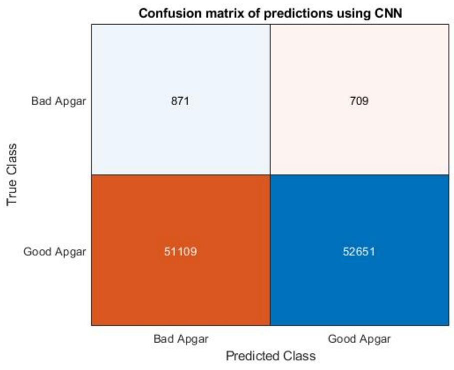
(c) DT



(d) CNN_s

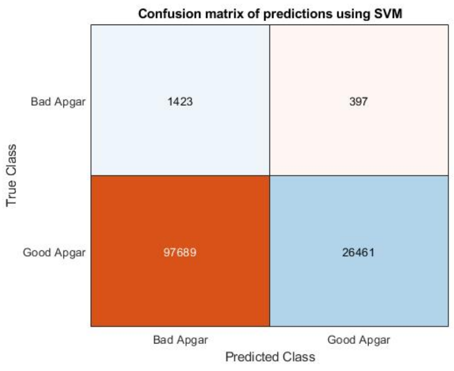


(e) CNN_m

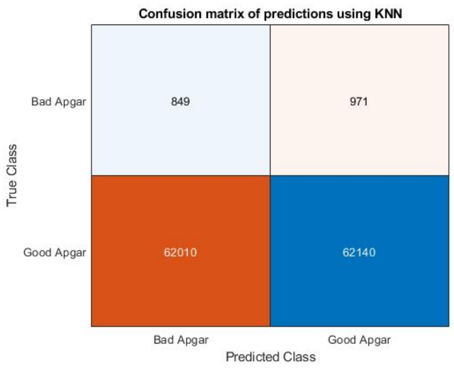


(f) CNN_l

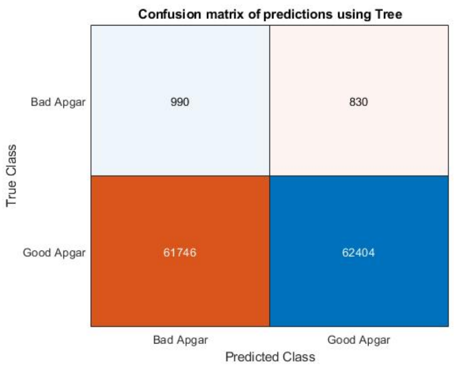
Figure 7.5: Confusion matrices of predictions using six different machine learning models for the k-NN segments with feature combination *TOCO & FHR*. The predictions were made on test data with 10534 samples.



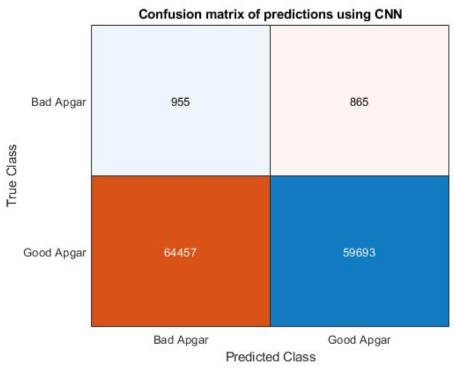
(a) SVM



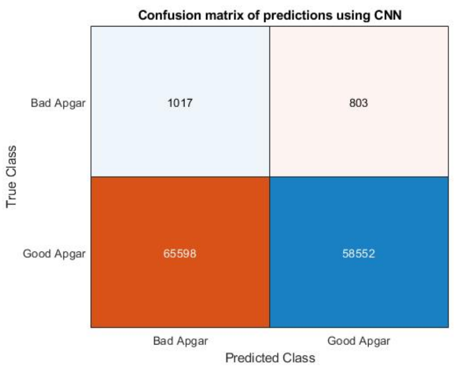
(b) k-NN



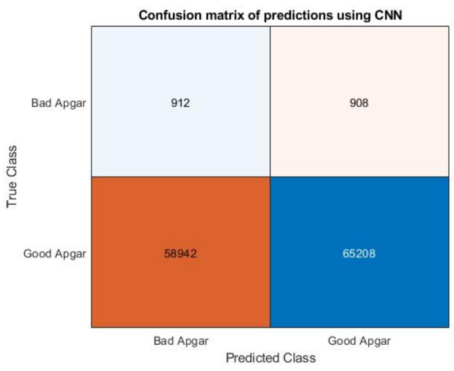
(c) DT



(d) CNN_s

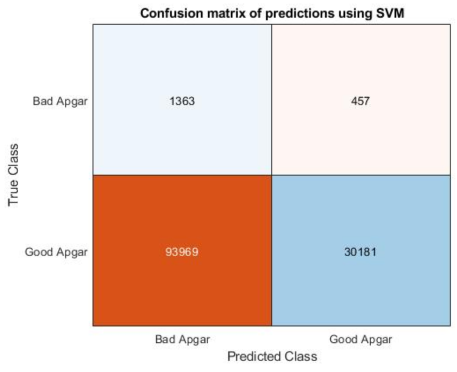


(e) CNN_m

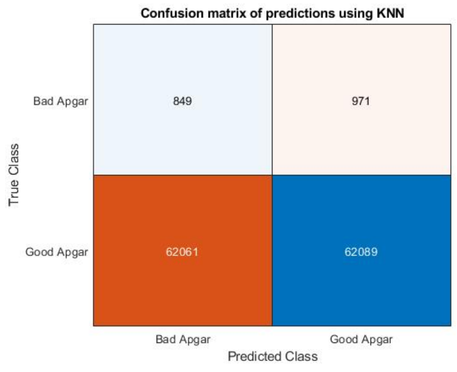


(f) CNN_l

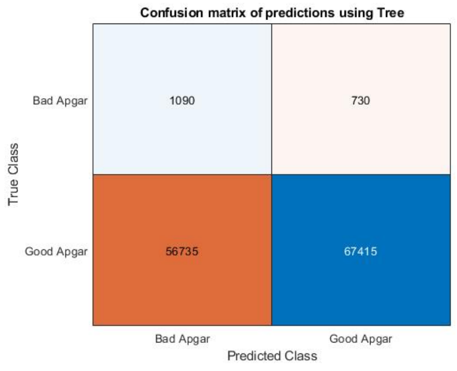
Figure 7.6: Confusion matrices of predictions using six different machine learning models for the SVM segments with feature combination *TOCO & FHR*. The predictions were made on test data with 12597 samples.



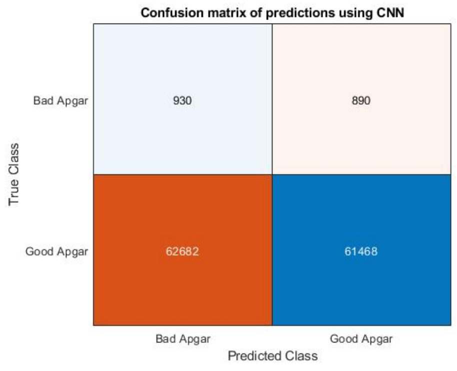
(a) SVM



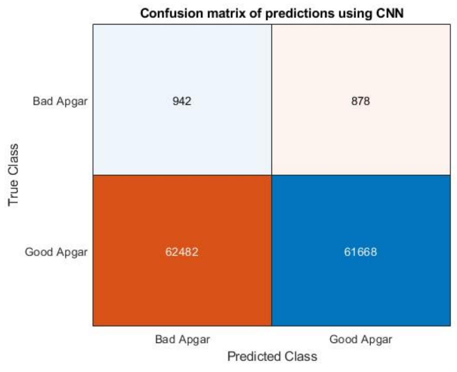
(b) k-NN



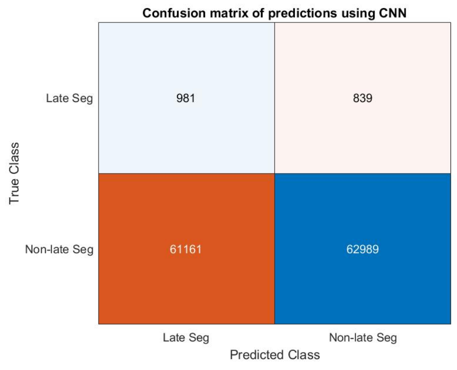
(c) DT



(d) CNN_s

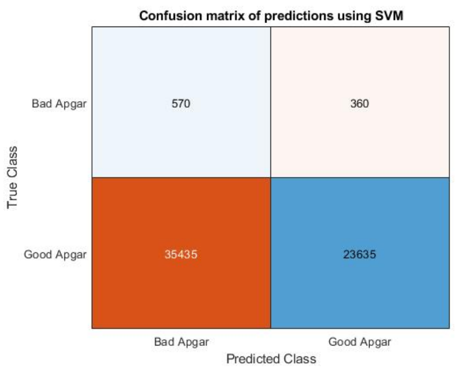


(e) CNN_m

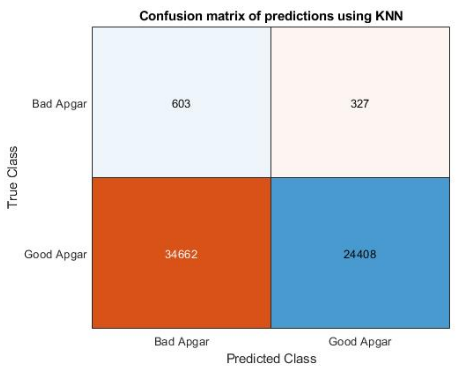


(f) CNN_l

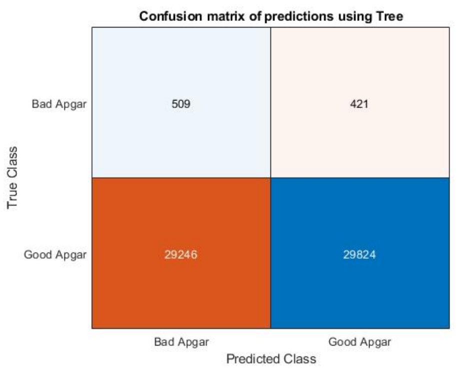
Figure 7.7: Confusion matrices of predictions using six different machine learning models for the SVM segments with feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Peak Frequency (per hour) & Gestation Age*. The predictions were made on test data with 12597 samples.



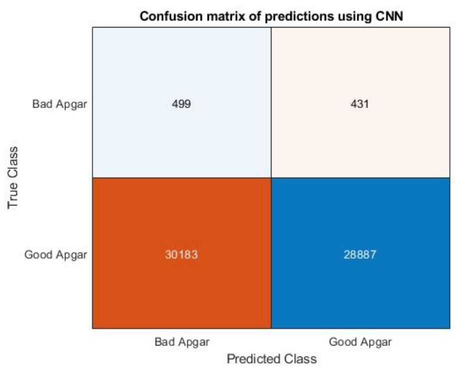
(a) SVM



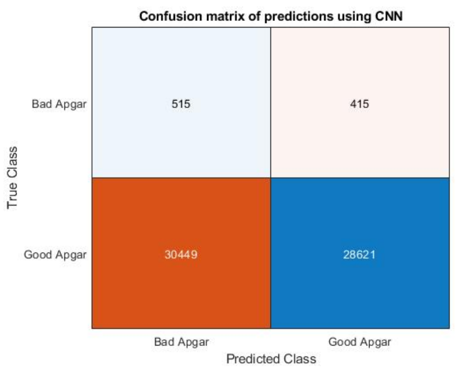
(b) k-NN



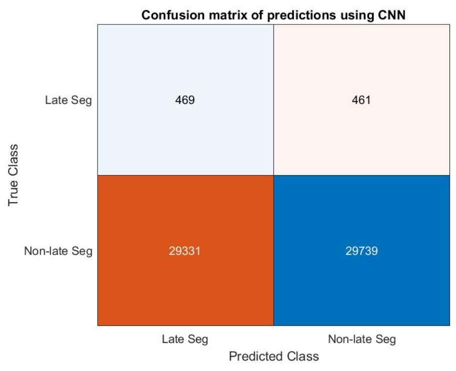
(c) DT



(d) CNN_s

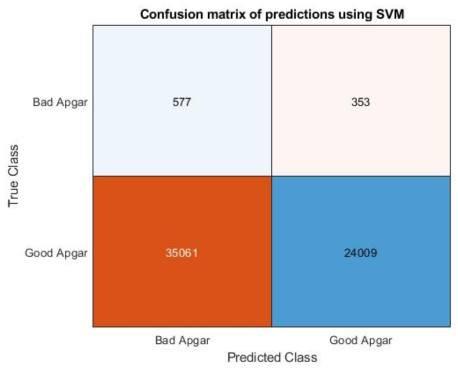


(e) CNN_m

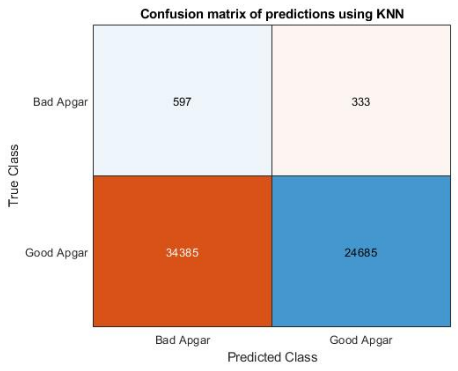


(f) CNN_l

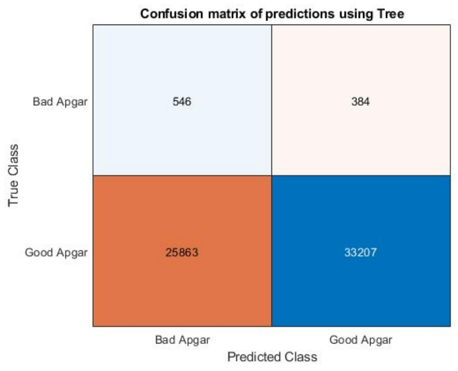
Figure 7.8: Confusion matrices of predictions using six different machine learning models for the mixed segments with feature combination *TOCO & FHR*. The predictions were made on test data with 6000 samples.



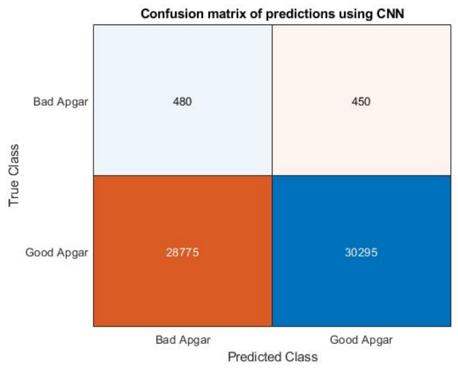
(a) SVM



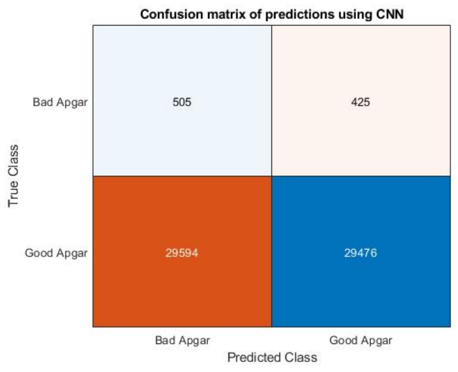
(b) k-NN



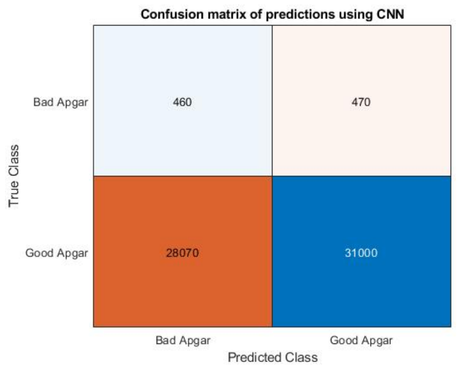
(c) DT



(d) CNN_s



(e) CNN_m



(f) CNN_l

Figure 7.9: Confusion matrices of predictions using six different machine learning models for the mixed segments with feature combination *TOCO, FHR, Parity, Smoking, Placental Abruption, Preeclampsia, Peak Frequency (per hour) & Gestation Age*. The predictions were made on test data with 6000 samples.

8. Conclusion

In the final chapter, the findings from this thesis are discussed, and potential reasons for the given results are accounted for. Further, possible improvements in the model performance are explored, and plausible steps are presented. Finally, recommendations and future outlooks within the field of modeling and interpreting CTG curves from labor are presented.

8.1 Conclusion

The primary aim of this thesis was to conduct the initial steps in developing a model that could predict the outcome of labor from continuous CTG data. One initial approach was to develop a peak detection model that could indicate uterine contractions in the TOCO curve. The results showed that by using a k-nearest neighbors model, it is possible to achieve a sensitivity of 98.70%, a specificity of 90.15%, an accuracy of 94.43%, and a F1 score of 94.65%. The obtained results imply good possibilities to investigate the dependency between fetal heart rate and the uterine contractions, more specifically, how the FHR variability responds to the pressure variations due to the contractions. Nevertheless, despite promising results, the peak detection model should be further improved by adding the duration and shape of the contraction, in addition to indicating the maximum.

Moreover, the ability to classify different stages of labor in the CTG data was investigated. The different stages of labor were extracted with two approaches: classification using pattern recognition and machine learning, and classification using the frequency of uterine contractions. The best model for stage classification was a convolutional neural network model with 23 consecutive layers, using TOCO and FHR as inputs. The model had a F1 score of 78.52% and could detect non-late segments with 75.60% sensitivity. For finding segments based on contraction frequency, the best model, regarding the number of segments, was obtained by a support vector machine. However, it is hard to evaluate the reliability of the results in both cases, since the time of the delivery is unknown.

When investigating the relationship between CTG and Apgar scores, for the obtained stages of labor from the previous part, no major distinctions in the results were found. Late segments, within the last 30 minutes of CTG measurement, got the highest results. However, the differences were small. Nevertheless, it could be an indication that there is more information in the CTG measurement closer to birth when labor work is more profound. The best model was a decision tree with 69.89% sensitivity, 60.52% specificity, 60.66% accuracy, and 5.26% F1 score. The imbalance between low and high Apgar scores in the data, has a negative impact on the F1 score, which partly explains the deviating value. Moreover, there seemed to be an added value in including the clinical parameters as features. The derived features did improve the results, even though they still were discouraging. However, to get more reassuring results for Apgar classification, there needs to be other and better features that can distinguish the two outcomes from each other.

8.2 Future Outlook

Based on the conclusions, further research is needed to determine the relationship between CTG and the outcome of labor. An initial step is to further develop the peak detection model to distinguish the duration of the uterine contractions, besides indicating where the contraction is peaking. There are also reasons to believe that there is information to be gathered from the integral of the uterine contraction. This information could be used in addition to the TOCO curve, to investigate how the fetal heart rate is affected by uterine contractions.

One of the main limiting factors is the absence of information that indicates the stage of labor. To better understand if stages of labor should be a segmenting factor when investigating the relationship between Apgar and CTG, one future research is to refine the data. We believe that a manual assessment by an obstetrician, where the different stages of labor are indicated for the CTG files would give more robust and reliable results. Ideally, the given signal and clinical data should have a parameter stating the time of birth and/or the cervical dilation. If this was realized, there could be a higher chance for machine learning algorithms to find patterns in the CTG data when comparing segments within the same stages of labor.

To further examine how the Apgar score of the fetus is related to the CTG curve, it would be beneficial to generate more data with low Apgar scores. Due to the imbalance between good and bad outcomes of labor, the data sets in the final chapter were very small. To overcome this problem, augmentation is one possible solution. After the augmented data is gathered, a model with the same or slightly altered feature combinations should be trained. Given the larger data sets, the models should yield better and more robust results.

References

[1] *Födda i Sverige.* URL: http://www.scb.se/hitta-statistik/sverige-i-siffror/manniskorna-i-sverige/fodda-i-sverige/ (visited on 05/06/2022).

[2] Källén, K. “Obstetrisk kvalitetssäkring och perinatal statistik”. In: *Obstetrik.* Ed. by Hagberg, H., Maršál, K., and Westgren, M. 2:2. Lund: Studentlitteratur, 2014, pp. 661–669.

[3] Hagberg, H. and Bennow, M. “Fosterasfyxi”. In: *Obstetrik.* Ed. by Hagberg, H., Maršál, K., and Westgren, M. 2:2. Lund: Studentlitteratur, 2014, pp. 219–226.

[4] Amer-Wåhlin, I. and Herbst, A. “Fosterövervakning under förlossningen”. In: *Obstetrik.* Ed. by Hagberg, H., Maršál, K., and Westgren, M. 2:2. Lund: Studentlitteratur, 2014, pp. 239–252.

[5] Cnattingius, S., Johansson, S., and Razaz, N. “Apgar Score and Risk of Neonatal Death among Preterm Infants”. In: *New England Journal of Medicine* 383.1 (July 2020), pp. 49–57. DOI: 10.1056/NEJMoa1915075.

[6] Spilka, J. et al. “Analysis of obstetricians’ decision making on CTG recordings”. In: *Journal of Biomedical Informatics* 51 (Oct. 2014), pp. 72–79. DOI: 10.1016/j.jbi.2014.04.010.

[7] Simiyu, I. N. et al. “Prevalence, severity and early outcomes of hypoxic ischemic encephalopathy among newborns at a tertiary hospital, in northern Tanzania”. In: *BMC Pediatrics* 17.1 (Dec. 2017), p. 131. DOI: 10.1186/s12887-017-0876-y.

[8] Chandraharan, E. “Rational approach to electronic fetal monitoring during labour in ’all’ resource settings”. In: *Sri Lanka Journal of Obstetrics and Gynaecology* 32 (Jan. 2012). DOI: 10.4038/sljog.v32i4.3988.

[9] 2016, S. B. (and Obstetrik och Gynekologi (SFOG), S. F. för. *Svenska riktlinjer för CTG-bedömning vid intrapartal fosterövervakning.* URL: https://www.sfog.se/media/312959/ctg-kort.pdf (visited on 03/31/2022).

[10] Technical Working Group, World Health Organization. “Care in Normal Birth: A Practical Guide”. In: *Birth* 24.2 (June 1997), pp. 121–123. DOI: 10.1111/j.1523-536X.1997.tb00352.x.

[11] Nordström, L. and Wiklund, I. “Förlossningens handläggning”. In: *Obstetrik.* Ed. by Hagberg, H., Karel Maršál, and Westgren, M. 2:2. Lund: Studentlitteratur, 2014, pp. 107–122.

[12] Hutchison, J., Mahdy, H., and Hutchison, J. “Stages of Labor”. In: *StatPearls.* Treasure Island (FL): StatPearls Publishing, 2022. URL: http://www.ncbi.nlm.nih.gov/books/NBK544290/ (visited on 04/22/2022).

[13] Fontenla-Romero, O., Alonso-Betanzos, A., and Guijarro-Berdinas, B. “Adaptive pattern recognition in the analysis of cardiotocographic records”. In: *IEEE Transactions on Neural Networks* 12.5 (Sept. 2001), pp. 1188–1195. DOI: 10.1109/72.950146.

[14] Şahin, H. and Subasi, A. “Classification Of The Cardiotocogram Data For Anticipation Of Fetal Risks Using Machine Learning Techniques”. In: *Applied Soft Computing* 33 (Apr. 2015). DOI: 10.1016/j.asoc.2015.04.038.

[15] Melnikov, A. D., Tsentalovich, Y. P., and Yanshole, V. V. “Deep Learning for the Precise Peak Detection in High-Resolution LC–MS Data”. In: *Analytical Chemistry* 92.1 (Jan. 2020), pp. 588–592. DOI: `10.1021/acs.analchem.9b04811`.

[16] Risum, A. B. and Bro, R. “Using deep learning to evaluate peaks in chromatographic data”. In: *Talanta* 204 (Nov. 2019), pp. 255–260. DOI: `10.1016/j.talanta.2019.05.053`.

[17] Tsymbal, O. *TOP 5 Machine Learning Algorithms For Business Applications.* URL: `https://mobidev.biz/blog/5-essential-machine-learning-techniques` (visited on 03/25/2022).

[18] *What Is Machine Learning? — How It Works, Techniques & Applications - MATLAB & Simulink.* URL: `https://se.mathworks.com/discovery/machine-learning.html` (visited on 04/11/2022).

[19] Hastie, T., Tibshirani, R., and Friedman, J. H. *The elements of statistical learning: data mining, inference, and prediction.* 2nd ed. Springer series in statistics. New York, NY: Springer, 2009.

[20] *Support Vector Machines for Binary Classification - MATLAB & Simulink - MathWorks Nordic.* URL: `https://se.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html` (visited on 04/11/2022).

[21] Ali, N., Neagu, D., and Trundle, P. “Evaluation of k-nearest neighbour classifier performance for heterogeneous data sets”. In: *SN Applied Sciences* 1.12 (Dec. 2019), p. 1559. DOI: `10.1007/s42452-019-1356-9`.

[22] Everitt, B. et al. “Miscellaneous Clustering Methods”. In: *Cluster Analysis.* John Wiley & Sons, Ltd, 2011, pp. 215–255. DOI: `10.1002/9780470977811.ch8`.

[23] Loh, W.-Y. “Classification and regression trees”. In: *WIREs Data Mining and Knowledge Discovery* 1.1 (Jan. 2011), pp. 14–23. DOI: `10.1002/widm.8`.

[24] *Fit binary decision tree for multiclass classification - MATLAB fitctree - MathWorks Nordic.* URL: `https://se.mathworks.com/help/stats/fitctree.html#butluiw_head` (visited on 04/22/2022).

[25] *What is a Convolutional Neural Network?* URL: `https://se.mathworks.com/discovery/convolutional-neural-network-matlab.html` (visited on 04/22/2022).

[26] Yamashita, R. et al. “Convolutional neural networks: an overview and application in radiology”. In: *Insights into Imaging* 9.4 (Aug. 2018), pp. 611–629. DOI: `10.1007/s13244-018-0639-9`.

[27] Goodfellow, I., Bengio, Y., and Courville, A. *Deep learning.* Adaptive computation and machine learning. Cambridge, Massachusetts: The MIT Press, 2016.

[28] Brownlee, J. *A Gentle Introduction to the Rectified Linear Unit (ReLU).* Jan. 2019. URL: `https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/` (visited on 05/18/2022).

[29] *Max pooling layer - MATLAB - MathWorks Nordic.* URL: `https://se.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.maxpooling2dlayer.html#d123e113993` (visited on 04/22/2022).

[30] *Specify Layers of Convolutional Neural Network - MATLAB & Simulink - MathWorks Nordic.* URL: `https://se.mathworks.com/help/deeplearning/ug/layers-of-a-convolutional-neural-network.html` (visited on 04/22/2022).

[31] *4. Fully Connected Deep Networks - TensorFlow for Deep Learning [Book].* URL: `https://www.oreilly.com/library/view/tensorflow-for-deep/9781491980446/ch04.html` (visited on 04/22/2022).

[32] *Softmax layer - MATLAB - MathWorks Nordic.* URL: https://se.mathworks.com/help/deeplearning/ref/nnet.cnn.layer.softmaxlayer.html (visited on 04/22/2022).

[33] *Classification output layer - MATLAB classificationLayer - MathWorks Nordic.* URL: https://se.mathworks.com/help/deeplearning/ref/classificationlayer.html (visited on 04/22/2022).

[34] Aszemi, N. M. and Dominic, P. "Hyperparameter Optimization in Convolutional Neural Network using Genetic Algorithms". In: *International Journal of Advanced Computer Science and Applications* 10.6 (2019). DOI: 10.14569/IJACSA.2019.0100638.

[35] Pandey, S. *How to choose the size of the convolution filter or Kernel size for CNN?* July 2020. URL: https://medium.com/analytics-vidhya/how-to-choose-the-size-of-the-convolution-filter-or-kernel-size-for-cnn-86a55a1e2d15 (visited on 05/02/2022).

[36] Brownlee, J. *A Gentle Introduction to Padding and Stride for Convolutional Neural Networks.* Apr. 2019. URL: https://machinelearningmastery.com/padding-and-stride-for-convolutional-neural-networks/ (visited on 04/22/2022).

[37] Patrikar, S. *Batch, Mini Batch & Stochastic Gradient Descent — by Sushant Patrikar — Towards Data Science.* Oct. 2019. URL: https://towardsdatascience.com/batch-mini-batch-stochastic-gradient-descent-7a62ecba642a (visited on 04/22/2022).

[38] Brownlee, J. *Difference Between a Batch and an Epoch in a Neural Network.* July 2018. URL: https://machinelearningmastery.com/difference-between-a-batch-and-an-epoch/ (visited on 04/22/2022).

[39] *Options for training deep learning neural network - MATLAB trainingOptions - MathWorks Nordic.* URL: https://se.mathworks.com/help/deeplearning/ref/trainingoptions.html (visited on 04/22/2022).

[40] Le, J. *12 Useful Things to Know about Machine Learning.* June 2018. URL: https://towardsdatascience.com/12-useful-things-to-know-about-machine-learning-487d3104e28 (visited on 04/22/2022).

[41] Pragati, B. *What is Overfitting in Deep Learning and How to Avoid It.* Apr. 2022. URL: https://www.v7labs.com/blog/overfitting,%20https://www.v7labs.com/blog/overfitting (visited on 04/14/2022).

[42] *Placental abruption - Symptoms and causes - Mayo Clinic.* URL: https://www.mayoclinic.org/diseases-conditions/placental-abruption/symptoms-causes/syc-20376458 (visited on 04/29/2022).

[43] *Preeclampsia - Symptoms and causes.* URL: https://www.mayoclinic.org/diseases-conditions/preeclampsia/symptoms-causes/syc-20355745 (visited on 04/29/2022).

[44] Hale, D. *Recursive Gaussian Filter.* Tech. rep. Colorado School of Mines, Golden, CO, 2006. URL: https://inside.mines.edu/~dhale/papers/Hale06RecursiveGaussianFilters.pdf.

[45] Scholkmann, F., Boss, J., and Wolf, M. "An Efficient Algorithm for Automatic Peak Detection in Noisy Periodic and Quasi-Periodic Signals". In: *Algorithms* 5.4 (Nov. 2012), pp. 588–603. DOI: 10.3390/a5040588.

[46] Maaten, L. van der and Hinton, G. "Visualizing Data using t-SNE". In: *Journal of Machine Learning Research* 9 (2008). Ed. by Bengio, Y., pp. 2579–2605. URL: https://www.jmlr.org/papers/volume9/vandermaaten08a/vandermaaten08a.pdf (visited on 04/26/2022).

[47] Soroker, A. *T-SNE Explained — Math and Intuition.* Aug. 2020. URL: `https://medium.com/swlh/t-sne-explained-math-and-intuition-94599ab164cf` (visited on 04/26/2022).

[48] D'souza, R. N., Huang, P.-Y., and Yeh, F.-C. "Structural Analysis and Optimization of Convolutional Neural Networks with a Small Sample Size". In: *Scientific Reports* 10.1 (Dec. 2020), p. 834. DOI: `10.1038/s41598-020-57866-2`.

[49] Persson, M. et al. "Five and 10 minute Apgar scores and risks of cerebral palsy and epilepsy: population based cohort study in Sweden". In: *BMJ* (Feb. 2018), k207. DOI: `10.1136/bmj.k207`.

[50] Persson, R. *Undersökningar av det nyfödda barnet.* May 2020. URL: `https://www.1177.se/barn--gravid/forlossning/efter-forlossningen/undersokningar-av-det-nyfodda-barnet/` (visited on 05/17/2022).

[51] *The Apgar Score.* Committee Opinion 664. The American College of Obstetricians and Gynecologists, Oct. 2015. URL: `https://www.acog.org/clinical/clinical-guidance/committee-opinion/articles/2015/10/the-apgar-score` (visited on 05/17/2022).

A. Appendix: Explanation of Data

Table A.1: Explanation CTG data

CTG-information	
FHR curve	
TOCO curve	
Sensor Type FHR	• No receiver • Ultrasound • FECG/ Electrode on Fetal Scalp
Sensor Type TOCO	• No receiver • External transducer
Signal Quality	• Poor • Okay • Good • Uncertain

Table A.2: Explanation Clinical Data

Clinical-information	
Study ID	
Parity	
Smoking	• Non-smoker • <10 cigarettes per day • >10 cigarettes per day • Unknown
Ablatio Placentae	
Preeclampsia	
Diabetes Type 1	
Delivery Type	• Unkown • Vaginal • Elective Cesarean Section • Emergency Cesarean Section • Urgent Cesarean Section • Vacuum Extraction, 1 • Vacuum Extraction, 2 • Vacuum Extraction, 3 • Forceps, 1 • Forceps, 2 • Forceps, 3
Gestaional Age	
Sex	
Birth Weight	
Weight Deviation	
Apgar 1 min	
Apgar 5 min	
Vital Status	• Not Dead • Dead Prior to Delivery • Dead During Delivery • Dead Past Delivery
UA pH	
Neonatal Care	
IVH - Intraventricular Bleeding	• None • Mild • Severe
HIE - Hypoxic Ischemic Encephalopathy	
Spasms	
Malformations	• None • Minor • Severe

B. Appendix: Specifications of CNN Model Settings

Note: NxM is size of each sample. N is the length of the time dependent vector plus constants. If $M > 1$ there are more than one time dependent vector, one in each column.

Table B.1: Small CNN

(a) Description of layers

No.	Layer	Parameters
1	Image Input Layer	N x M x 1
2	Convolution	20 channels 3xM convolutions
3	ReLU	
4	Max Pooling	3x1 max pooling with stride [1 1]
5	Fully Connected	2 neurons fully connected
6	Softmax	
7	Classification Output	crossentropyex

(b) Training Options

Options	Value
Initial learning rate	0.001
Batch Size	128
Number of Epochs	100

Table B.2: Medium CNN

(a) Description of layers

No.	Layer	Parameters
1	Image Input Layer	N x M x 1
2	Convolution	20 channels 3xM convolutions
3	ReLU	
4	Max Pooling	3x1 max pooling with stride [1 1]
5	Convolution	40 channels 3x1 convolutions
6	ReLU	
7	Max Pooling	3x1 max pooling with stride [1 1]
8	Convolution	60 channels 3x1 convolutions
9	ReLU	
10	Max Pooling	3x1 max pooling with stride [1 1]
11	Fully Connected	2 neurons fully connected
12	Softmax	
13	Classification Output	crossentropyex

(b) Training Options

Options	Value
Initial learning rate	0.00001
Batch Size	64
Number of Epochs	100

Table B.3: Large CNN

(a) Description of layers

No.	Layer	Parameters
1	Image Input Layer	N x M x 1
2	Convolution	20 channels 3xM convolutions
3	Batch Normalization	
4	ReLU	
5	Max Pooling	3x1 max pooling with stride [1 1]
6	Convolution	40 channels 3x1 convolutions
7	Batch Normalization	
8	ReLU	
9	Max Pooling	3x1 max pooling with stride [1 1]
10	Convolution	60 channels 3x1 convolutions
11	Batch Normalization	
12	ReLU	
13	Max Pooling	3x1 max pooling with stride [1 1]
14	Convolution	80 channels 3x1 convolutions
15	Batch Normalization	
16	ReLU	
17	Max Pooling	3x1 max pooling with stride [1 1]
18	Fully Connected	20 neurons fully connected layer
19	ReLU	
20	Dropout	25% dropout
21	Fully Connected	2 neurons fully connected
22	Softmax	
23	Classification Output	crossentropyex

(b) Training Options

Options	Value
Initial learning rate	0.00001
Batch Size	64
Number of Epochs	100

Master's Theses in Mathematical Sciences 2022:E50
ISSN 1404-6342

LUTFMS-3451-2022

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

http://www.maths.lu.se/