Quantification of Fetal Volume in Magnetic Resonance Images Using Deep Learning

Amanda Nilsson

Master's thesis
2021:E9

Lund University

Faculty of Engineering
Centre for Mathematical Sciences
Mathematics

CENTRUM SCIENTIARUM MATHEMATICARUM

Quantification of Fetal Volume in Magnetic Resonance Images Using Deep Learning

Amanda Nilsson

Supervisors: Einar Heiberg, Erik Hedström
Examiner: Karl Åström

Master's Thesis in Mathematical Sciences
Faculty of Engineering

LUND UNIVERSITY

ABSTRACT

Background: Fetal Magnetic Resonance Imaging is an imaging technique that visualizes the fetus during pregnancy. It is used for assessing and diagnosing disease, for monitoring high risk pregnancies, and for conducting research. By segmenting three-dimensional fetal magnetic resonance (MR) images, volume estimation of intrauterine structures is possible. The quantification of fetal body volume and placental volume is of both clinical and scientific relevance. If performed manually, the segmentation is a time-consuming procedure.

Purpose: The aim of this project was to create a fully automatic algorithm that segments and quantifies the volumes of the fetus and placenta in MR images.

Method: Deep learning, or more specifically a two-dimensional U-Net, was used for solving the task. The data used for training, validation, and testing of the models were manually delineated volumetric MR images of 25 fetuses, which were considered as ground truth. In all images the fetus, placenta, umbilical cord and amniotic fluid were delineated. In addition to segmenting the fetus and placenta, the model was also trained to segment the umbilical cord and amniotic fluid.

Result: For the automatic fetal body volume the median absolute volume difference was 2.3 % (std 4.9 percentage points) for five cases that were used for testing, when compared to the ground truth. For the placental volume, the median absolute volume difference was instead 23.9 % (std 34.4 percentage points).

Conclusion: In conclusion, the results suggest that a fully automatic method based on the U-Net can be used for quantification of fetal body volume. However, the quantification of placental volume was not satisfactory.

ACKNOWLEDGEMENT

I would like to thank the Lund Cardiac MR group - writing my Master's Thesis in the group has been both fun and rewarding. I am grateful for having gotten the opportunity to work with this project. I especially would like to thank my supervisors Einar Heiberg and Erik Hedström for their guidance, expertise, and the support they have given me during the process. In addition, I want to thank Daniel Salehi and Stefan Lupeskovic for their efforts and contributions to the project and Marjolein Piek for her valuable feedback. Finally, I want to thank my family and friends for always being there for me and believing in me - I highly appreciate it.

ABBREVIATIONS

ANN – Artificial Neural Network

AVD – Absolute Volume Difference

CMR – Cardiovascular Magnetic Resonance Imaging

CNN – Convolutional Neural Network

DL – Deep Learning

DSC – Dice Similarity Coefficient

EFW – Estimated Fetal Weight

FBV – Fetal Body Volume

FN – False Negative

FP – False Positive

GPU – Graphical Processing Unit

JSI – Jaccard Similarity Index

ML – Machine Learning

MR – Magnetic Resonance

MRI – Magnetic Resonance Imaging

ReLU – Rectified Linear Unit

RF – Radiofrequency

TE – Echo Time

TN – True Negative

TP – True Positive

TR – Repetition Time

CONTENTS

INTRODUCTION

Magnetic Resonance Imaging (MRI) can be used during pregnancy to investigate disease such as cardiovascular disease and for evaluating high risk pregnancies. Magnetic resonance (MR) images are excellent at displaying anatomy and can therefore be utilized to examine the anatomy of the intrauterine fetus and surrounding structures like the placenta [1]. Figure 1 shows a two-dimensional (2D) MR image of a fetus.

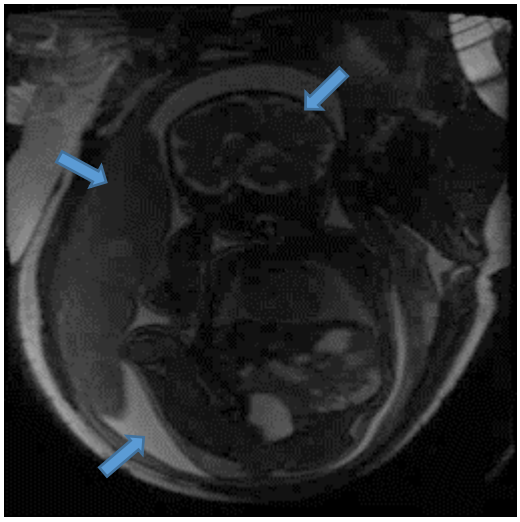

Figure 1: An image slice of a three-dimensional fetal MR image. Arrows point toward the placenta, amniotic fluid and fetus.

The clinical standard method for fetal imaging is 2D ultrasound. MRI may allow imaging in cases where ultrasound gives poor image quality [2, p. 348] and may offer additional information of relevance for investigating disease [3]. The disadvantage of MRI is that it is more complicated and expensive, less accessible and that the technique for fetal imaging is still evolving [3].

Since MRI generates three-dimensional (3D) images, it can be used for estimation of organ volumes – for instance fetal and placental volume. There are several reasons for estimating fetal body volume (FBV). For instance, fetal body volume can be used for estimation of fetal weight (EFW). This was first introduced by Baker et. al. [4] in 1994 using MRI. Since then, several studies have compared fetal weight estimation based on 3D MR images, with the more commonly used two-dimensional (2D) ultrasound method [4] [5] [6]. Normally ultrasound is used for estimation of fetal weight, in which case biometric measurements such as head circumference and femur length, among others, are used for making the estimate [6]. However, in several studies MRI has been shown to be more reliable than ultrasound for EFW [4] [5] [6]. Another application of fetal body volume is for investigating whether the lungs are developing normally or not by indexing the lung volume to the fetal body volume [2, pp. 230, 348]. Additionally, previous studies [7] [8] index blood flow to fetal weight (converted from fetal body volume) to enable comparison.

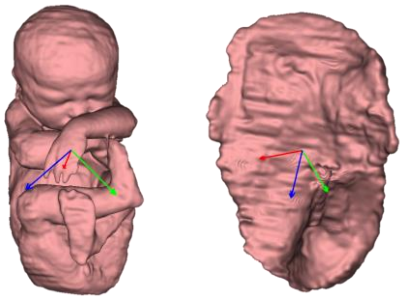

Figure 2: 3D-models of a fetus and a placenta. The models are based on manual delineations of MR images.

Estimation of fetal body and placental volume can be performed by manually delineating fetal MR images [9] [10], see example of manual delineations in Figure 2. However, manual delineation is a time-consuming procedure [9] [11]. Beside manual delineation, semi-automatic methods [11] [5] [6] for segmenting fetuses exist. The repeatability of estimated fetal body volume when using a semi-automatic method is investigated in [12], where the inter- and intra-observer variability was found to be low. Furthermore, the same study indicated that the experience of the operator is not of particular importance when using a semi-automatic method [12]. With semi-automatic software presented in [11] and [5] fetal

segmentation is stated to take about 5-10 minutes. Nevertheless, it would be beneficial with a fully automatic method, that is fast and does not need any human interaction. An automatic method may open the possibility to integrate volume estimation in the MRI scanner protocol. To the authors best knowledge, no fully automatic method for segmenting fetal volume exists today. On the other hand, Torrents-Barrena et. al. [13] has presented a fully automatic method for delineating the placenta.

With a fully automatic method, valuable time and resources could be saved and the use of fetal body volume could be further implemented in clinical practice and research. The use of estimated fetal body volume and placental volume could then lead to improved diagnostics and help answer research questions.

High quality images are essential for accurate delineation and volume estimation and can be limited by the presence of artifacts in the images. Motion artifacts are caused by for instance movement of the fetus or maternal breathing. With movement the contours of the tissues can become blurred or multiplied – making delineation difficult. Other artifacts include noise and fold-over artifacts. Fold-over artifacts can complicate segmentation by making it hard to distinguish the affected parts in the image [2, pp. 29, 66-70, 77, 78].

Another challenge with delineation is signal intensities of the tissues within the uterine cavity. The signal intensities differ between separate organs and tissues. For example, the brain and urine bladder have about the same intensity as the amniotic fluid, while the intensity differs from the skin, muscles, and amniotic sac. Therefore, simple thresholding cannot be used as a single method for delineation of the fetus [9].

Since the contrast between skin and amniotic fluid is high for fetal MR images of decent quality, thresholding tools can be used during delineation to identify the contours of the fetus by finding the border between the amniotic fluid and the skin. However, thresholding cannot be utilized if the amount of surrounding amniotic fluid is too small or the fetus is positioned close to the placenta or amniotic sac since the contrast will not be sufficient [9].

Research and clinical environment

This Master's Thesis project is associated with Lund Cardiac MR group and Medviso AB, which both closely work together. The Cardiac MR Group is situated at Clinical Physiology, Department of Clinical Sciences Lund, Lund University, Skåne University Hospital, Lund, Sweden, where both research and clinical diagnostics regarding cardiovascular disease is performed. One of the group's expertise include advanced fetal cardiovascular magnetic resonance imaging (fetal CMR). Medviso AB is a medical software company that provides medical imaging software for both research and clinical purposes. Physicians (and researchers) at the department would like to index flow measurements to placental and fetal volumes to account for differences in size as to be able to compare values during fetal growth, and between different pathologies and healthy fetuses. To implement this in clinical practise and research, a quick and accurate method for estimation of fetal and placental volume is necessary.

Purpose

The aim of this study is to investigate the possibility of using machine learning to create an algorithm that automatically segments MR fetal images and quantifies fetal body volume and placental volume accurately and reliably. The network architecture to be investigated is a convolutional neural network (CNN) called U-Net. The U-Net architecture was introduced by Ronneberg et. al. [14] in 2015 and has been successfully used in several biomedical imaging segmentation tasks and competitions [15]. Manually delineated fetal MR images will be used as ground truth.

Specifically, the following four classes shall be segmented:

- Fetus
- Placenta
- Umbilical cord
- Amniotic fluid.

Although there are four classes, the focus and goal are to successfully segment the fetus and placenta.

THEORY

The aim of this section is to give an overview and explanation of concepts and theories of importance for the project. First, some of the fundamental concepts of magnetic resonance imaging are described. Then, theories and principles regarding machine learning and methods for evaluation of models are provided.

Magnetic Resonance Imaging

Magnetic resonance imaging (MRI) is an imaging technique commonly used in medical applications. The technique is based on quantum spin of nuclei, a strong external magnetic field with gradients, and radiofrequency (RF) pulses. Quantum spin is when a nucleus with an uneven number of neutrons or protons, such as ^{1}H (a proton), spins around its own axis – generating a small magnetic field [16, pp. 1-2].

The strong external magnetic field changes the orientation of the otherwise unordered spinning nuclei by arranging them according to the external field. In addition, the nuclei start precessing around the axis of the field. The frequency of precession (ω) depends on the gyromagnetic constant γ and the magnetic field strength B_0 according to the Larmor equation, $\omega = \gamma B_0$ [16, p. 2].

By sending RF-pulses of a specific frequency, the spinning nuclei with the corresponding precession frequency become flipped for a short period of time before re-aligning with the strong magnetic field. With the flip, a magnetic field orthogonal to the external field is produced and measurable for a short period of time. The flip angle is affected by the energy of the RF-pulse and its length. The repetition time (TR) and echo time (TE) describe how often a new RF-pulse is sent out and after how long the signal is registered, respectively. The time it takes for a flipped nucleus to go back to its "un-flipped" state is tissue dependent. As a result, it is possible to alter the contrast of MR images by changing the repetition and echo time [16, pp. 2-4].

Machine Learning

In machine learning (ML), algorithms (also referred to as models) are trained with data (that is, *training data*) to perform different tasks, for example classification. The models have learnable parameters, called weights and biases, which are updated during training. This project uses supervised learning where labelled data is used for training the models. The labelled data is considered as ground truth, and the models are trained to produce outputs as close as possible to the ground truth labels [17, pp. 96-98, 103, 105, 106]. During training, a loss function is used to measure the closeness between the prediction and ground truth. This can have a large impact on the updating of the learnable parameters and on the outcome.

Artificial Neural Networks

Artificial neural networks (ANN) are inspired by, and constructed to somewhat resemble, the nervous system. The nervous system consists of connected cells called neurons that receive and transmit information. Signals are received through the dendrites of a neuron, while output signals travel through the axon of the neuron. The incoming signals can act either excitatory or inhibitory. If the sum of the incoming signals is strong enough, and thus exceeding a threshold, the neuron becomes triggered and sends out a signal. Likewise, an artificial neuron, also referred to as a node, receives inputs and based on the inputs produces an output [18, pp. 207-208]. A weighted sum of the inputs and an activation function determine the output from a node.

The nodes can be organized into layers. In feed-forward neural networks, which are used in this project, output from neurons in one layer can be sent to neurons in the following layers. The layer that receives

input to the feed-forward network is referred to as the *input layer*, while the layer that produces the output of the network is called the *output layer*. The layers in between are known as *hidden layers* [17, pp. 164-165]. An example of a feed-forward network can be seen in Figure 3. The illustrated ANN is a fully connected neural network, which means each node is connected to the nodes in the subsequent layer.

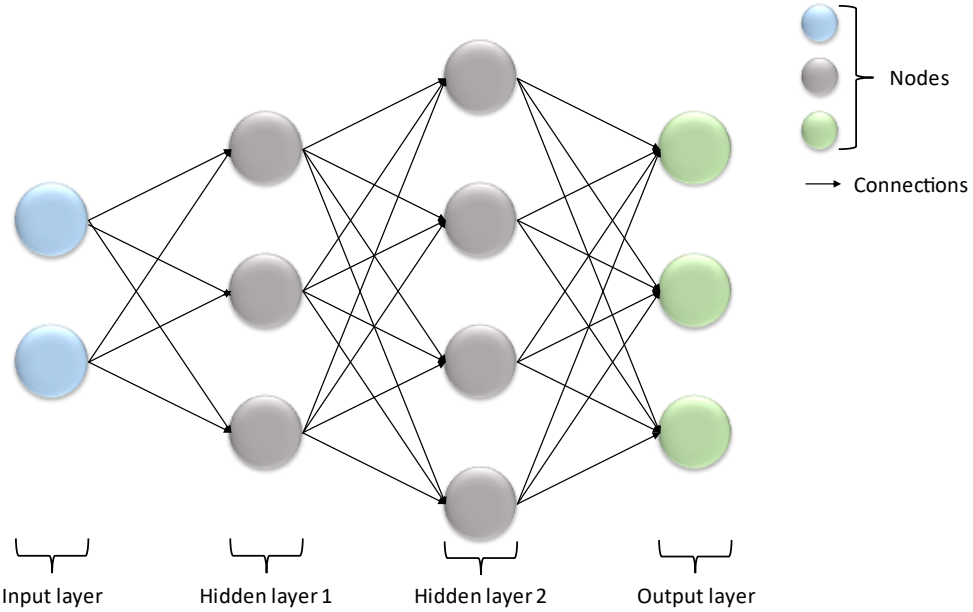

Figure 3: Fully connected feed forward neural network with four layers – first an input layer, then two hidden layers and lastly an output layer. Each node is connected to all nodes in the subsequent layer.

Each connection in the ANN has a learnable weight that determines the effect the output of a node has on a subsequent node. The output of a neuron is produced by computing the weighted sum of the inputs and applying an activation function. By training the ANN, thus updating its weights and biases, the network can be trained to produce a desired output based on an input [17, pp. 164-171].

Activation functions are important for ANNs. The activation function determines if and what output that should be generated from a node based on its inputs. Additionally, non-linear activation functions are necessary for models to be able to solve more than linear problems [17, p. 168]. Rectifying linear unit (ReLU) is a common activation function, defined as $\phi(x) = \max(0, x)$ [17, pp. 170, 171, 189]. Another activation function is the soft-max function, which can be used as output layer in models solving classification tasks. The soft-max function generates *n* values between 0 and 1 – one value for each class – where the sum of the values equals 1. The values represent a probability distribution given a specific input [17, pp. 180-181].

Deep Learning

Deep learning (DL) is a subgroup of machine learning that during the last 5-10 years have found major applications in medical image analysis [19]. Deep learning is characterized by ANNs with many layers (i.e. deep networks). A common application is semantic segmentation, that is, pixelwise classification of images [19]. In deep learning, training incorporates feature selection that helps improve the performance of the model [20].

Convolutional Neural Networks

Convolutional neural networks (CNNs) are a subgroup of feed-forward neural networks. Based on hundreds of papers on deep learning in medical image analysis Litjens, G. et al. [19] conclude that the use of CNNs is the most common approach for solving tasks within the subject.

Common layers in CNNs are convolutional layers, pooling layers, and fully connected layers. The convolutional layers are used for feature extraction and each convolution produces what is called a feature map. An example of a convolutional operation is 2D-convolution, which can be applied on images. 2D-convolution is completed by sliding a window (also known as filter or kernel) of a specified size over an image and performing a mathematical operation [21]. In convolutional layers the kernels contain the learnable weights and are updated during training to extract relevant features for the task at hand. This means that the weights are shared [19]. A convolutional layer can consist of several kernels, each producing its own feature map [20]. The operation for 2D-convolutions with a kernel of size $N \times N$ is given by

$$X_{mn} = (\boldsymbol{x} * w)_{mn} = \sum_{j=0}^{N-1} \sum_{i=0}^{N-1} w[i,j] \cdot \boldsymbol{x}[m-i, n-j],$$

where X_{mn} is the resulting feature map, x is the image, and w is the kernel [18, pp. 226, eq. 9.24]. An illustration of a convolutional operation can be seen in Figure 4.

Stride is the number of pixels a window moves over at a time. An image can be padded, which means that pixels are added around the image. The added pixels can for instance have intensity value zero [18, p. 226]. The stride and padding affect the dimensions of the resulting matrix when incorporated in a mathematical operation.

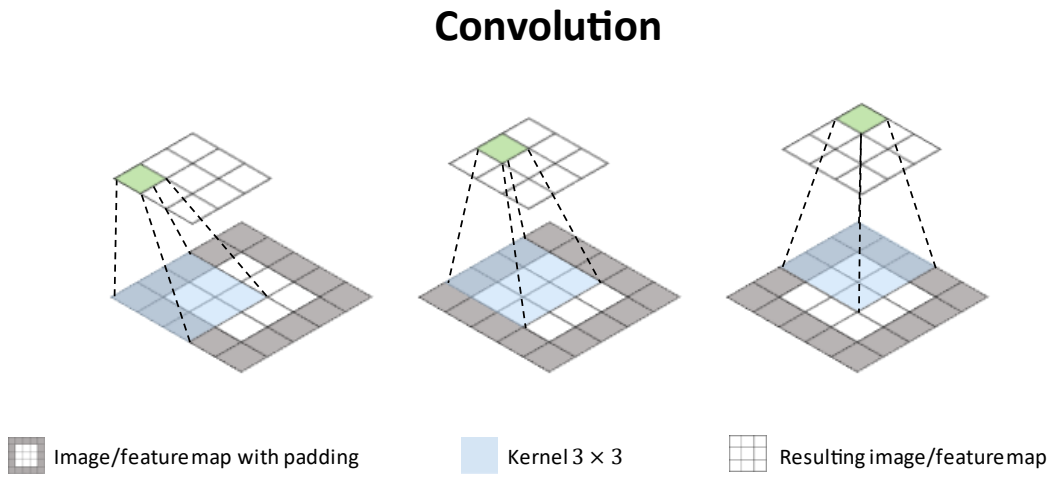

Figure 4: Illustration of three of the steps of a 2D-convolution. The operation is performed on an image or feature map of dimension 3×3 *with a kernel of dimension* 3×3*, padding and stride 1. The grey area around the original image/feature map represents the padding. Consequently, the resulting feature map has the same dimension* 3×3.

In contrast to the convolution layers, the pooling layers do not have learnable weights. Instead, the pooling layers are commonly used for height and width reduction of the feature maps [21]. Additionally, by reducing the dimensions of the feature maps the following convolutional layer will be able to extract new and larger structures/features [19]. The pooling operations are performed on grids (of a specified size) in the feature maps, where each grid operation results in a value [21]. For max pooling, which is the pooling operation that is most frequently used, the maximum value in each grid is selected and the rest are discarded, resulting in a new feature map [20]. An example of max pooling can be seen in Figure ***5***.

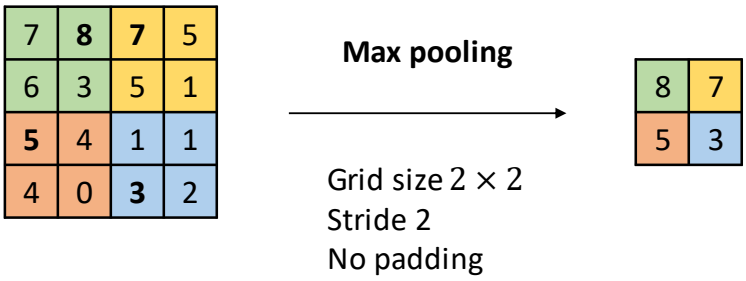

Figure 5: Illustration of a 2×2 *max pooling operation with stride 2 and no padding. The left matrix represents a feature map with dimension* 4×4*, where the maximum value in each coloured grid is highlighted. To the right is the resulting feature map with dimension* 2×2 *after max pooling.*

Training of Deep Learning Models

When training a model, its weights and biases are updated to optimize the performance of the model. How they are updated strongly depends on the chosen optimization method.

Adam (adaptive moment estimation) [22] is a stochastic optimization method using momentum, where subsets of the training data, called minibatches, are used for updating the learnable parameters. Consequently, the weights and biases are updated in an iterative process. The vector of weights and biases are referred to as θ. Furthermore, the parameters are updated to minimize a loss function $E(\theta)$, using its gradient $\nabla E(\theta)$ [22]. Basically, a loss function compares the output of the network with the ground truth – the lower the loss, the better. Cross entropy loss function is used for classification problems, such as semantic segmentation of images. Categorical cross entropy is defined as

$$E = -\frac{1}{N}\sum_{n=1}^{N}\sum_{i=1}^{K} d_{ni}\ln(y_{ni}),$$

where N is the number of samples, K is the number of classes, d is the ground truth and y is the prediction based on an input [23]. When using Adam, the weight updates are dependent on parameters called *learning rate*, *gradient decay factor* and *squared gradient decay factor* which are parameters in the optimization method. The learning rate influence how much the weights are updated at each iteration, and the gradient decay factor and squared gradient decay factor are decay rates of momentum estimates. The values of these parameters are set prior to starting a training [22].

Hyperparameters are parameters that control the training process or the network architecture, aside from weights and biases. The hyperparameters can be tuned to improve the performance of a model on unseen data [17, pp. 118, 119, 422, 423].

The learning rate can be divided into three separate hyperparameters – *initial learning rate*, *learning rate drop factor* and *learning rate drop period*. The initial learning rate is the learning rate at the start of the training, while the drop period and drop factor determines how often (in terms of epochs) and by how much the learning rate is decreased, respectively. An epoch is defined as the number of iterations it takes to go through a complete training set, and the maximum number of epochs determines for how many epochs the training will go on.

Another hyperparameter is the *L2 regularization strength*. When applying L2 regularization, a term that penalizes large weights is added to the loss function. The penalty can be controlled with the L2 regularization strength. By adjusting the regularization strength, overtraining, which is explained in the section Measure performance, can be prevented [17, pp. 117, 230].

Furthermore, the training data can be shuffled during training. By shuffling the data after each epoch, the order changes which reduce the risk of overtraining [24].

U-Net

In 2015, Ronneberg et. al. introduced the U-Net architecture [14], which is a CNN that is developed to solve segmentation tasks, especially within the biomedical field. An explanation of the basics of the U-Net is given below and a simplified version of the architecture is provided in Figure ***6***.

As explained by Ronneberg et. al. "The architecture consists of a contracting path to capture context and a symmetric expanding path that enables precise localization." [14]. The contracting path consists of repetitive steps, where every step includes convolutions with ReLU as activation function and max pooling after each convolution. Following each max pooling operation, the number of feature channels is increased by a factor two. The expanding path also consists of repetitive steps. However, each step includes up-sampling of the feature maps, "up-convolution", concatenation, and convolutions instead. The "up-convolution" reduces the number of feature channels by a factor two. In the concatenation phase skip-connections are used – concatenating each feature map from the expanding path with the feature map from the corresponding step in the contracting path. The last step of the expanding path ends with a third convolution. This final operation produces the output of the network [14]. The *encoder depth* is a hyperparameter which determines the number of steps in the contracting (and expanding) path, thereby affecting the number of layers and weights of the U-Net [25].

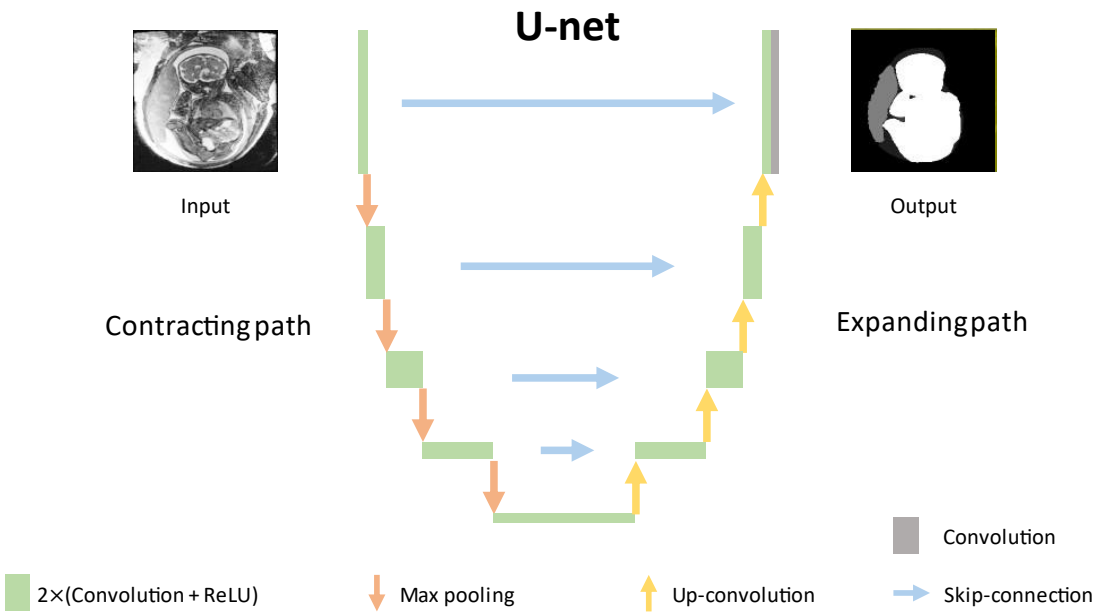

Figure 6: U-Net architecture with encoder depth 4. Inspired by the first figure in the original U-Net article [14].

Measure performance

By using evaluation metrics models can be optimized and their performance estimated. Furthermore, overfitting and underfitting can be detected. Overfitting means a model has adapted to noise in the data used for training. Consequently, an overfitted model has a high performance on the training data and low performance on previously unseen data. Signs of underfitting are on the other hand, both low performance on seen and unseen data [17, pp. 109-110].

Generalization performance is a measure of how well a model performs on previously unseen data [17, p. 108]. In this project unseen data is referred to as either *validation data* or *test data*. The validation

data is used for hyperparameter tuning, while the test data is used for estimating the generalization performance. As validation data is used for optimizing the model, the validation performance is biased. Therefore, generalization should be estimated by measuring the performance on the test data [17, p. 119].

K-fold cross-validation can be used for estimating the generalization performance. With k-fold cross-validation, the training data is split into *k* parts i.e., folds. In an iterative process, the different folds are used as either training or validation data. First, k-1 folds are used for training the network, and the remaining fold is used for validation of the network. In the next iteration another fold is used for validation and the k-1 folds are used for training, and so on [17, p. 120]. The average performance can be considered as an estimate of the generalization performance.

Several relevant metrics exist for evaluating the performance of a neural network. This section focuses on evaluating semantic segmentations, especially by looking at the classification on pixel level and comparing it with the ground truth.

If a pixel is correctly labelled, it is either a true positive (TP) or a true negative (TN). If it is a TP it is correctly predicted to belong to a certain class, while if it is a TN it is correctly predicted to *not* belong to that specific class. In case a pixel is instead incorrectly classified, it is either a false positive (FP) or a false negative (FN) depending on if it was predicted to belong to or *not* belong to a certain class.

Accuracy is a common evaluation metric that is defined as the percentage of correctly classified pixels, $\frac{TP+TN}{TP+TN+FP+FN}$. The global accuracy is the overall accuracy and only registers if a pixel is correctly classified or not. For unbalanced multi-class problems, the global accuracy is not good as it becomes biased towards the more frequently occurring classes. In those cases, per-class accuracy is more relevant as it is the average of the accuracy for each individual class [26].

Another metric is the Jaccard Similarity Index (JSI) which is defined as $\frac{A \cap B}{A \cup B}$, where A and B are two sets [26]. For instance, A can be a predicted segmentation and B the ground truth, see Figure 7. JSI can also be described as $\frac{TP}{TP+FP+FN}$ [26] [27]. A metric similar to the JSI, is the DICE Similarity Coefficient (DSC) which is defined as $\frac{2|A \cap B|}{|A|+|B|}$ or $\frac{2 \cdot TP}{2 \cdot TP+FP+FN}$ [26].

Boundary F1 (BF) score is a metric used for evaluating how well the boundaries of two segmentations correspond to one another. It is defined as

$$\frac{2 \cdot P \cdot R}{P + R},$$

where P is the precision and R is the recall [28]. In return, precision and recall are defined as $\frac{TP}{TP+FP}$ and $\frac{TP}{TP+FN}$, respectively [26].

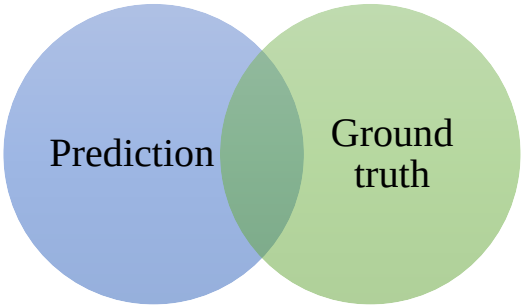

Figure 7: Venn diagram of two sets. The more overlap between the sets, the higher Jaccard Similarity Index.

For classification problems a confusion matrix can be used to visualize the performance of a model by showing predicted labels on one axis and true labels on the other. In a normalized confusion matrix, the predictions are given as a percentage of the total number of true labels for each class, see Figure 8 [29].

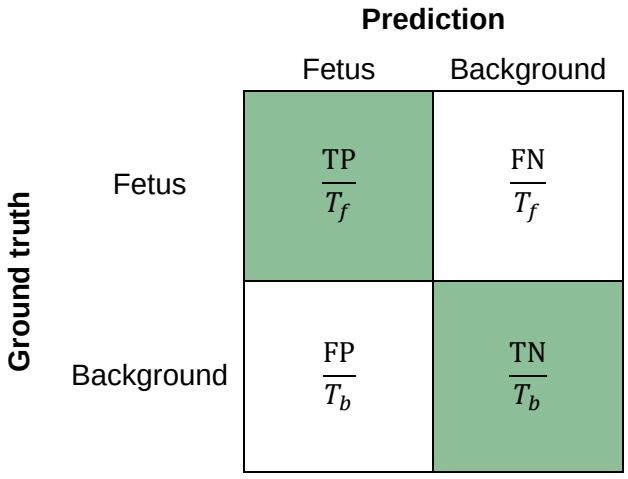

Figure 8: Example of a normalized confusion matrix for a binary classification problem. T_f is the total number of true fetus labels and T_b is the total number of true background labels.

Since the aim of this project is to automatically generate volume estimations, another relevant evaluation metric is the Absolute Volume Difference (AVD) which is defined as

$$\text{AVD} = \frac{\left|V_{\text{Segmentation}} - V_{\text{Ground truth}}\right|}{V_{\text{Ground truth}}} \cdot 100\ \%,$$

where the $V_{\text{Segmentation}}$ is the volume of a predicted segmentation and $V_{\text{Ground truth}}$ is the volume of the ground truth segmentation of an object [26]. Noteworthy, is that the AVD does not consider the overlap between segmentations.

Data Augmentation

To create machine learning models that perform well on unseen data, thus having a high generalization performance, it is desirable to have a large set of data for training the models [17, p. 236]. However, in medical applications, data is often limited due to the time-consuming procedure of acquiring and labelling data. A method for increasing generalization performance, especially in cases of limited data, is data augmentation [30] [17, p. 236]. Data augmentation can be performed either offline or on-the-fly – meaning it is performed before or during the training process, respectively.

When a model is to be used in a medical application, it is important that it works in a clinical setting. When it comes to MR images, the properties of the images, such as appearance, quality, and spatial

arrangement, can vary depending on for example acquisition protocol, scanner vendor and patient population [31]. Zhang et al. [31] propose a method for increasing domain generalization with data augmentation.

Geometric transformations such as rotation, scaling, mirroring, and translation can be performed to augment the spatial arrangement of an image. Rotation is performed in either two or three dimensions. Mirroring and rotation can be applied to simulate different orientations of an object (such as the fetus). Scaling transformations and translation aim to make the model invariant to the size and position of an object, respectively.

Brightness and contrast transformations are performed to augment pixel intensities, thereby affecting the image appearance. Furthermore, images can be blurred or sharpened, and noise can be added to simulate different image qualities [31].

DATA

The data used in this project consisted of volumetric (i.e., three-dimensional) MR images of 26 intrauterine human fetuses in gestational week 29 to 39, acquired at Skåne University Hospital in Lund. The project had ethical approval and the data was pseudonymized to exclude patient specific data.

A 1.5 T scanner (Aera, Siemens, Erlangen, Germany) was used for acquiring the volumetry fetal MR images. Steady-state free-precession (SSFP) was used as MRI sequence. During acquisition, the parameters for TE, TR and flip angle were typically 1.74 ms, 4 ms and 51 degrees, respectively.

As inclusion criteria the quality of the cases had to be good enough to perform accurate manual delineations. This meant cases with too many artefacts, such as motion and fold-over artifacts, were excluded (9 out of 35). The anatomy within the uterine cavity had to contain no gross abnormalities, as the goal was for the model was to be able to segment "normal" cases. Furthermore, only singleton pregnancies were included in this study, except for one twin pregnancy. However, the twin pregnancy was not manually delineated like the rest of the datasets and was only included for visual assessment of performance. Consequently, the images in 25 of the 26 cases included in this study were manually delineated.

Manual delineations of the data, performed with the software Segment 3DPrint (Medviso AB, Lund, Sweden), were used as ground truth, see example in Figure 9. The manual delineations were carried out by the author, a medical student, and a PhD student, under supervision of a physician expert in the field (one of the supervisors). The author did adjustments and corrections to all manual delineations before they were finalized. When using Segment 3DPrint to create manual delineations, tools such as, drawing, thresholding, smoothing and filling tools, amongst others, were used. For each case, manual delineations of the fetus, placenta, umbilical cord, and amniotic fluid were created. The amniotic fluid was created by delineating the amniotic sack and subtracting the other delineations from it – leaving only the amniotic fluid left in the delineation. As part of the final step of the manual delineations a closing (1 mm) and smoothing (2 mm) operation was applied to the delineations, to smooth the surfaces of the objects. Examples of manually delineated objects are seen in Figure 10.

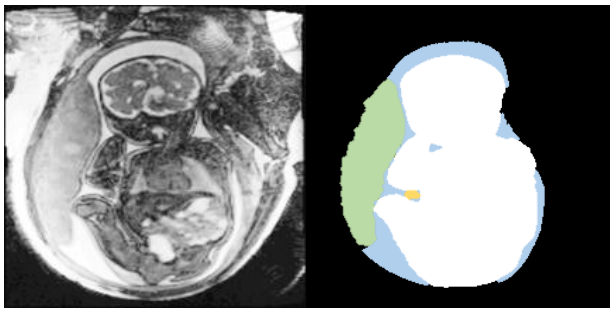

Figure 9: To the left is a 2D fetal MR image with adjusted signal intensities. To the right is the corresponding mask based on manual delineations, where the colours have been altered. White represents the fetus, green the placenta, yellow the umbilical cord, blue the amniotic fluid and black the background.

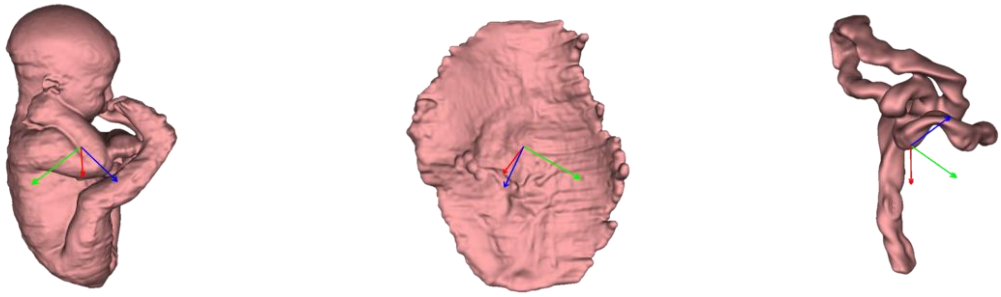

Figure 10: 3D-models of a fetus, a placenta and an umbilical cord. The models are based on manual delineations of a 3D fetal MRI image.

Furthermore, all volumetric images were made isotropic i.e., the resolution of the voxels were set to be the same in all dimensions. The images had different dimensions, but the median dimensions for the 25 singleton pregnancy cases were $512 \times 336 \times 307$. From these 25 cases 26,185 unique two-dimensional images could be extracted. Of all the pixels, 84.2 % are classified as background, 9.7 % as fetus, 2.9 % as placenta, 2.9 % as amniotic fluid and 0.3 % as umbilical cord.

METHODS

In this section the methodology of the study is presented. First, an overview is given, then a more in-depth description is provided in the subsections. A visual overview of the project methodology is shown in Figure 11.

Throughout the project, MATLAB [32] has been used as programming language. Several toolboxes, provided by MATLAB, has been used to process and transform data, as well as to train convolutional neural networks. Heavy computations, such as training of the neural networks have been performed on a compute server with four graphic cards, two NVIDIA Titan RTX and two NVIDIA GeForce RTX 3090.

Deep learning was used to perform automatic segmentation of fetal magnetic resonance images, allowing volumetry measurement of the classified objects. More specifically, a convolutional neural network (CNN) with U-Net structure [14] was trained to semantically segment fetal MR images. Semantic segmentation can be explained as classifying every pixel in an image, by assigning a class label to each pixel. The U-Net used a 2D-image as input and returned a semantically segmented 2D-image as output. The pixel labels used for classification were: *fetus*, *placenta*, *umbilical cord*, *amniotic fluid*, and *background*.

Since manual delineation of fetal MR images is time consuming and started at the initialization of the project, the manual delineations were carried out in parallel with the rest of the project.

The manually segmented MR images were used as ground truth for training and testing of the networks, as well as for validation during the training process. Since the amount of available data was limited, data augmentation was performed to generate more data for training of the networks. Considering the varying positions and postures of intrauterine fetuses the networks were trained on 2D images from all three image-axes in the volumetry images, instead of being trained on images from a specific anatomical plane.

Several experiments were carried out, on part of the data, to improve the performance of the automatic segmentation, for example by tuning hyperparameters, and testing different forms of pre-processing and data augmentation.

When the manual delineations were complete, k-fold cross-validation on eight models was performed. Based on the result, a model was selected, trained, and tested. Then, the segmentations were post-processed in different ways. The performance was evaluated using several metrics.

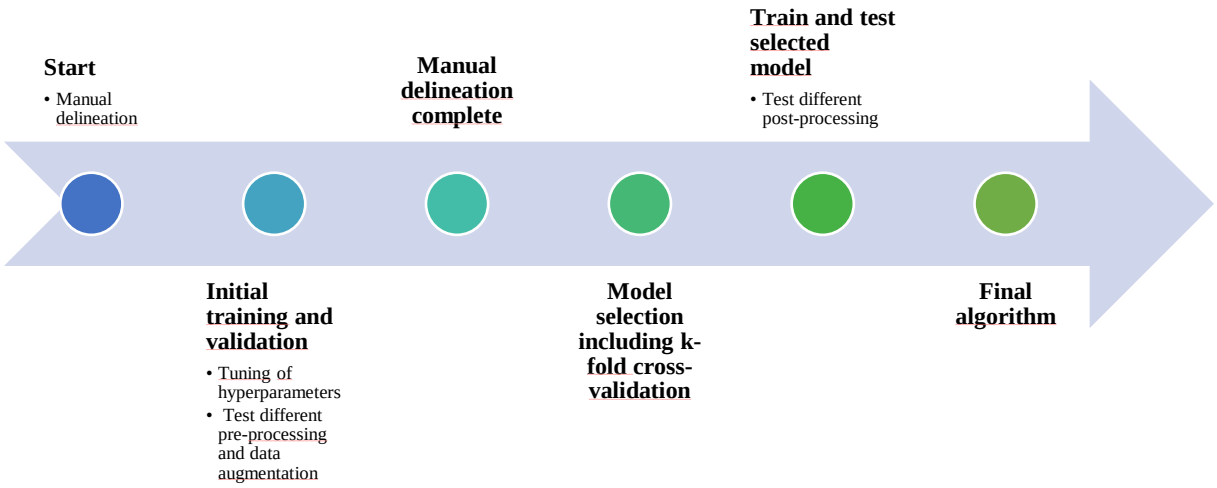

Figure 11: An overview of the process from start to finish, highlighting important steps in the methodology.

Ground Truth

As mentioned, manual delineations were used as ground truth. Each class (pixel label) corresponded to a pixel intensity value. As ground truth for each 2D image a "mask" was created by assigning the specific intensity values to every pixel based on the manual segmentations. Since the input of the U-Net was 2D images, image slices were extracted from the volumetry images. Because of the different postures and positions of prenatal fetuses, the image slices could not be extracted from a specific anatomical plane. Instead, 2D images were obtained from all three coordinate axes. This way the model could be trained on more data, hopefully generalize better, and become more robust. Figure 12 illustrates some of the differences that may exist between different fetal MR images.

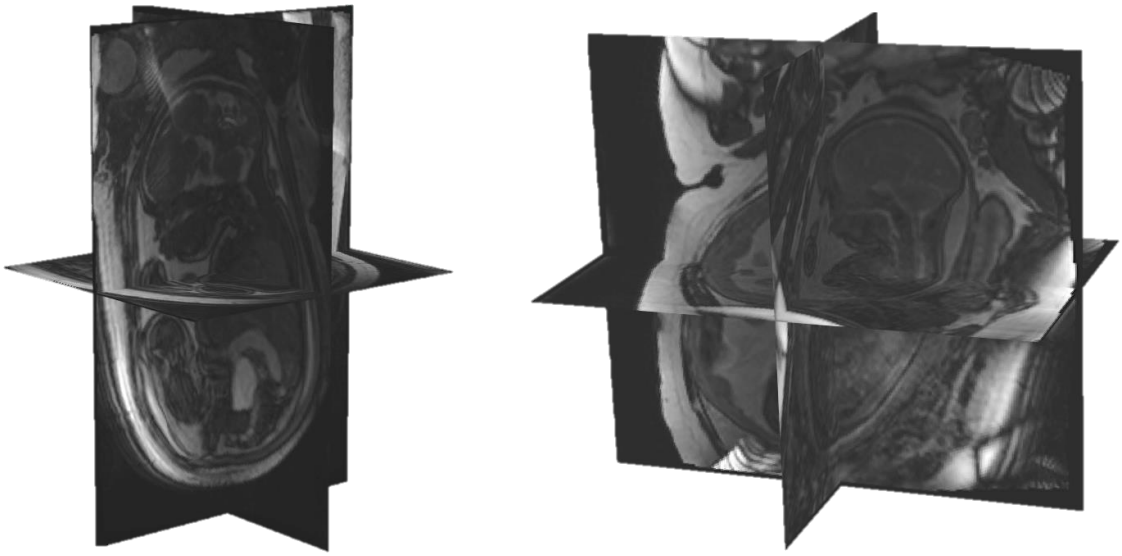

Figure 12: Slice planes for two different fetal MR images. Note the different positions of the fetuses, the different image dimensions and the brightness.

Pre-processing

Before the U-Net was applied, the images were pre-processed. Different methods for pre-processing were evaluated before settling with a final method for pre-processing. These experiments were

conducted under the same conditions, except for the method of pre-processing. For example, histogram equalization was performed on the images, either on the individual 2D images or on the complete 3D images. Histogram equalization adjusts the contrasts of an image by changing the intensities, making the histogram of the new image as flat as possible. Histogram equalization seemed to improve the performance of the networks and 3D histogram equalization was therefore used as part of the pre-processing. In all cases the 2D images and masks were resized, transformed to the data type uint8, and exported to PNG-files before being fed to the CNN. When resizing, the dimensions of the images and masks were resampled to 256×256 pixels.

Data Augmentation

Various forms of data augmentation were used to make the algorithm more robust and increase its generalization performance. Augmentations were only performed on the training data, and not on the validation data or test data. Consequently, the models were evaluated on "real" data. The augmentations were performed off-line i.e., prior to training the models. Throughout this study, different methods and combinations of augmentations were used. In this section, one combination of augmentations is presented – namely the combination used in in the latter part of the process when all manual delineations were complete. An overview of the specific augmentation settings can be seen in Table 1.

To decrease class imbalance, 2D images (and corresponding masks) containing foreground were oversampled 15 times, and then augmented in the same manner as the rest of the training data. Images containing only background were not oversampled.

The methods, probabilities and magnitudes of the data augmentation were heavily inspired by Zhang et. al. [31] and Isensee et. al. [15]. Zhang et. al. [31] propose a method for increasing domain generalization with data augmentation, while Isensee et. al. [15] suggest a deep learning framework for semantic segmentation of biomedical images.

The data augmentation included 3D rotation around an axis, with randomized rotation between -20 and 20 degrees. The angle was drawn from a uniform distribution. With 3D rotation the number of datasets used for training was "increased" and the network could be trained on more positions (seeing the objects from new perspectives). When 3D rotation was applied, it was applied prior to any other augmentation. Furthermore, the 3D rotation was applied once for each axis and 2D images and masks were extracted from a randomly chosen plane each. The 2D images and masks were then augmented in the same manner as the "original" data.

For the following augmentations, an individual probability and magnitude were set. The probability describes the likelihood of an augmentation being applied and the magnitude defines how much the augmentation is applied. Due to the probabilities and randomized magnitudes different combinations of augmentations were applied. Seeing that the images containing foreground were oversampled, many unique versions of these images were created. The augmentations were performed in the order described below.

Two-dimensional rotation was performed with a 20 % probability. The angle for rotation was randomly drawn from a uniform distribution of values between -15 and 15 degrees, that is U(-15, 15). Furthermore, scaling was implemented with the same probability as for rotation and the magnitude, in this case the rescaling factor, was drawn from the uniform distribution U(0.7, 1.4). If the rescaling factor was smaller than one an out-zoomed effect was achieved, and zero-padding was performed to make the output the same size as the input. For the same reason centre-cropping was performed if the rescaling factor was larger than one (achieving a zoomed in version of the input). Then, mirroring along the axes was applied

with a probability of 0.1 for each axis. These geometrical transformations increased the variation of positions, orientations, and "sizes" of the objects in the data.

To simulate cases of different image quality, data augmentation altering sharpness, blurriness and noise level was implemented. Sharpening was applied with a probability of 0.15. The magnitude was drawn from a uniform distribution of values between 0.7 and 1.3. The radius for the applied sharpening had a radius randomly drawn from U(1, 5). Blurring with Gaussian filter had a probability of 0.1 of being applied, and the magnitude was U(0.25, 1.5). Then, Gaussian noise was added with a probability of 0.15, mean zero, and standard deviation randomly chosen from the distribution U(0, 0.1). Furthermore, another form of noise called speckle, was added with probability 0.2 and standard deviation U(0.001, 0.1).

Finally, to imitate cases of various image appearance, augmentation affecting the brightness and contrast was performed. The brightness of the images was transformed with a probability of 0.15. When transforming the brightness, the pixel intensities in the image were multiplied by a factor randomly chosen from the uniform distribution of values between 0.7 and 1.3. There was a probability of 0.15 to transform the contrast of the images. First, the minimum and maximum pixel intensity values were stored. Then, the image was multiplied with a factor that was randomly selected from U(0.65, 1.5). Following this, the values that were smaller or larger than the stored minimum and maximum value were changed to the stored minimum and maximum, respectively.

Table 1: Data augmentation that was applied prior to training the presented models. The augmentations were applied with the specified probabilities and magnitudes in the order they are presented. U stands for uniform distribution.

Data augmentation	Probability	Magnitude	Other information
3D-rotation	-	U(-20°, 20°)	
2D-rotation	0.2	U(-15°, 15°)	
Scaling	0.2	U(0.7, 1.4)	Centre output
Mirroring	0.5	-	For each axis
Sharpening	0.15	U(0.7, 1.3)	Radius U(1, 5)
Blurring with Gaussian filter	0.1	U(0.25, 1.5)	
Gaussian noise	0.15	U(0, 0.1)	
Speckle	0.2	U(0.001, 0.1)	
Brightness	0.15	U(0.7, 1.3)	
Contrast	0.15	U(0.65, 1.5)	

Tuning of Hyperparameters

The hyperparameters were tuned to optimize the model. Due to limited amount of time and resources some hyperparameter settings were fixed, and a limited number of combinations of hyperparameters were tested. The CNN architecture U-Net and the optimizer Adam were used throughout the project. Like in [15], it was prioritized to use as large patch size as possible to extract as much context from the images as possible which resulted in using complete 2D images. Hyperparameters, such as learning rate, L2 regularization strength, encoder depth, minibatch size and maximum number of epochs were varied over the experiments. Furthermore, different kinds of pre-processing and data augmentations were used through the process, as explained in the section Pre-processing and the section Data Augmentation, respectively.

While the manual delineations were under construction, preliminary versions of the manually delineated data were used for initial tuning of hyperparameters. For instance, eight datasets were used for training

and one for validation of several different models. During the initial tuning, methods for pre-processing and data augmentation were evaluated, as well as different encoder depths, learning rate, learning rate drop period, learning rate drop factor, L2 regularization strength, minibatch size, and gradient decay factor.

In one experiment, networks were trained using sets of different combinations of initial learning rate, learning rate decay factor and L2 regularization strength during one experiment. In another, it was tested whether the encoder depth, set to 3, 4, 5 or 6, affected the performance. A third example of an experiment was when the gradient decay factor was changed between 0.85, 0.9 and 0.95.

When all manual delineations were complete, further hyperparameter tuning was performed, see Figure 13. First, the data was randomly split into two parts: *training data* and *test data*. The training data consisted of 20 of the datasets, while the test data consisted of the 5 remaining datasets.

K-fold cross-validation, or more precisely 4-fold cross-validation, was performed for model selection. This meant randomly splitting the training data into four parts, each part containing five different datasets. By splitting the data in this manner, each fetus existed in only one of the folds. Since the datasets varied in size, the different parts differed somewhat in size.

The 4-fold cross-validation was performed in combination with grid search to optimize the hyperparameters and select a model. With grid search, sets of different combinations of hyperparameters are used and tested systematically [17, pp. 427-429]. For every fold, networks of each combination of non-fixed hyperparameters were trained and validated. Results from the initial hyperparameter tuning were used as a guideline when choosing which hyperparameters to include in the grid search.

For each fold, a grid search with 8 combinations of hyperparameters were performed – i.e., 8 different models. Consequently, 32 (8×4) training processes were completed. Each training session took about 42 hours on one GPU, which resulted in a total of 1344 GPU hours (~ 56 GPU days), aside from initial tuning of hyperparameters which was performed prior to the 4-fold cross-validation and grid search. The models are referred to as N1-N8.

The hyperparameters that were changed during this grid search were initial learning rate, learning rate drop factor and L2 regularization strength, see Table 2. The other hyperparameters were fixed (see Table 3). The data was pre-processed by performing 3D histogram equalization.

Table 2: Hyperparameters that were changed during 4-fold cross-validation grid search.

Hyperparameter	
Learning rate	[0.001, 0.0005]
Learning rate drop factor	[0.1, 0.5]
L2 regularization strength	[0.0001, 0.0005]

Table 3: Fixed hyperparameters during 4-fold cross-validation grid search.

Hyperparameter	
Encoder depth	4
Optimization method	Adam
Gradient decay factor	0.9000
Squared gradient decay factor	0.9990
Epochs	6
Minimum batch size	32
Shuffle	Every epoch

The performance of the eight models from the grid search was measured by computing the average validation accuracy for the networks trained with the same set of hyperparameters. The model (with hyperparameter settings) that performed best on average was selected.

Then, the selected model was trained on the training data (all four folds). The generalization performance was estimated by measuring the performance of the model on previously unseen data, in this case the part that is referred to as test data.

When testing the selected model, it was also investigated whether the segmentations improved when implementing voting. With voting, the automatic segmentation was performed from all three planes, instead of one. Then, voting decided the class label for each voxel. Voting was implemented to make the model more robust.

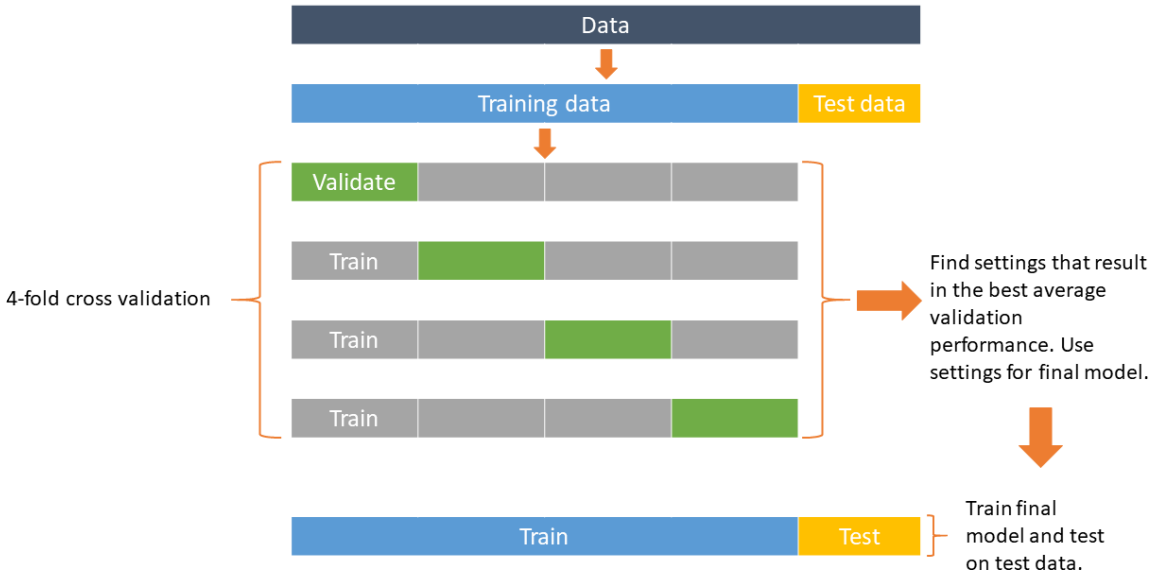

Figure 13: Process for tuning hyperparameters – illustrating splitting of the data and 4-fold cross validation.

Post-processing

To further improve the performance of the selected model various combinations of post-processing were performed on the segmentations from the models. Methods for post-processing that were investigated were 3D-filling and "Keeping the largest object", which are available functions in Segment 3DPrint. With 3D-filling, three-dimensional cavities in the objects were filled. By using the function "Keeping the largest object", all disjointed objects belonging to the same class were removed except for the largest one. These functions were only implemented on the classes fetus and placenta since they were the focus of this project.

Evaluation methods

Several metrics were used to evaluate the models. During hyperparameter tuning, the performance for each model was evaluated by computing the global validation accuracy. The validation accuracies for the classes fetus and placenta were also computed as part of the evaluation. Both the training accuracy (minibatch accuracy) and the global validation accuracy were used to detect overtraining.

When evaluating the selected model and methods for post-processing, several more metrics were used, including JSI, BF score, AVD, and normalized confusion matrix. The final algorithm was chosen based on the results.

RESULTS

To begin with, the results for the initial tuning of hyperparameters are presented. The performance was higher for the networks that used histogram equalization as pre-processing, than for the ones that did not. Furthermore, when using a larger mini-batch size, such as 32, the accuracy fluctuated less between iterations during training than when using a smaller minibatch size such as 2 or 4.

For the experiment where different combinations of initial learning rate, learning rate drop factor and L2 regularization strength were evaluated the validation performance of the networks were quite uniform. Similarly, the networks in the experiment with different gradient decay factors all had about the same validation performance. However, out of the networks that were trained with different encoder depths, the network with encoder depth 4 had the highest global validation accuracy. Although, all four networks had overall similar results.

Next to be presented is the results for the models in the 4-fold cross-validation and grid search. During the training process, the average global training accuracy and corresponding average global validation accuracy for each network followed about the same curve, see Figure 14 and Figure 15 for comparison. Average validation accuracies for each trained network can be seen in Table 4, together with the standard deviation between the folds. As can be seen in Table 4, the performance was about the same for all networks and the standard deviation between the folds was relatively small. However, network *N5* (initial learning rate 0.0005, learning rate drop factor 0.1 and L2 regularization strength 0.0001), had the highest average global validation accuracy before rounding the values.

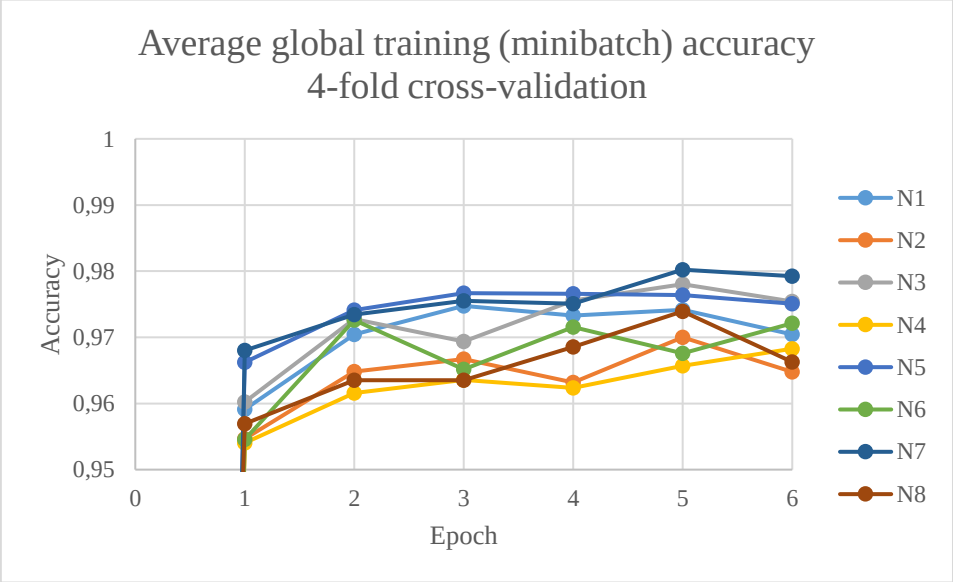

Figure 14: Average global training (i.e. minibatch) accuracy at each epoch during the training process for the 4-fold cross-validation.

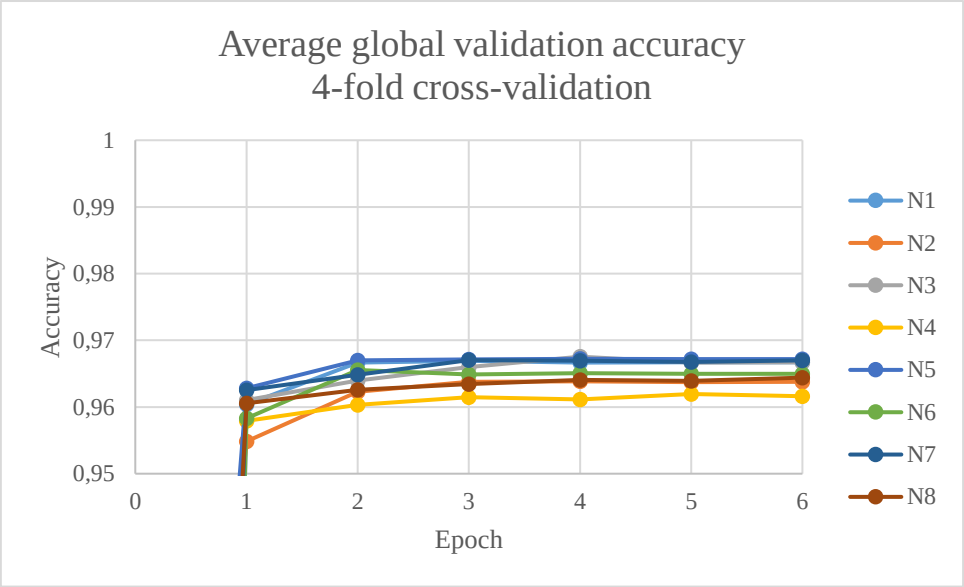

Figure 15: Average global validation accuracy at each epoch during the training process for the 4-fold cross-validation.

Table 4: Average validation accuracy between the four folds for the trained models in the 4-fold cross-validation, including the standard deviation. The selected model N5 is highlighted in bold.

Model	**Average global accuracy (std)**	**Average accuracy for the class fetus (std)**	**Average accuracy for the class placenta (std)**
N1	0.967 (0.009)	0.947 (0.023)	0.682 (0.053)
N2	0.964 (0.008)	0.938 (0.021)	0.663 (0.075)
N3	0.967 (0.007)	0.949 (0.020)	0.651 (0.070)
N4	0.962 (0.006)	0.906 (0.037)	0.624 (0.078)
N5	**0.967 (0.008)**	**0.948 (0.020)**	**0.664 (0.049)**
N6	0.965 (0.008)	0.947 (0.019)	0.683 (0.067)
N7	0.967 (0.008)	0.949 (0.018)	0.654 (0.067)
N8	0.964 (0.007)	0.932 (0.022)	0.631 (0.049)

Moving forward, the results for the selected model i.e., network *N5*, is presented. Without voting or post-processing, the average global test accuracy was 96.8 % for the five test cases. In comparison, it was 97.3 % when voting was applied, for more details see Table 5. Similarly, the accuracy, JSI and BF score for each class were higher for all foreground classes when voting was used, as can be seen when comparing the test performance presented in

Table 6 and Table 7. The normalized confusion matrix for when voting has been implemented is presented in Figure 16. The segmentations of the test cases and the twin case took between 29 and 91 seconds each on one of the GPUs when using the model with voting.

Table 5: Test performance for the selected model that was evaluated on five test cases.

	Global accuracy	Mean accuracy	Mean JSI	Weighted JSI	Mean BF Score
No voting	0.968	0.762	0.694	0.941	0.790
Voting	0.973	0.778	0.733	0.948	0.826

Table 6: Performance for the selected model on the test data for each class without voting.

Class	Accuracy	JSI	BF Score
Fetus	0.924	0.888	0.769
Placenta	0.595	0.522	0.437
Umbilical cord	0.610	0.519	0.688
Amniotic fluid	0.686	0.570	0.668
Background	0.993	0.972	0.899

Table 7: Performance on the test data for the selected model on each class with voting.

Class	Accuracy	JSI	BF Score
Fetus	0.939	0.918	0.802
Placenta	0.596	0.544	0.451
Umbilical cord	0.658	0.614	0.776
Amniotic fluid	0.701	0.615	0.699
Background	0.996	0.974	0.895

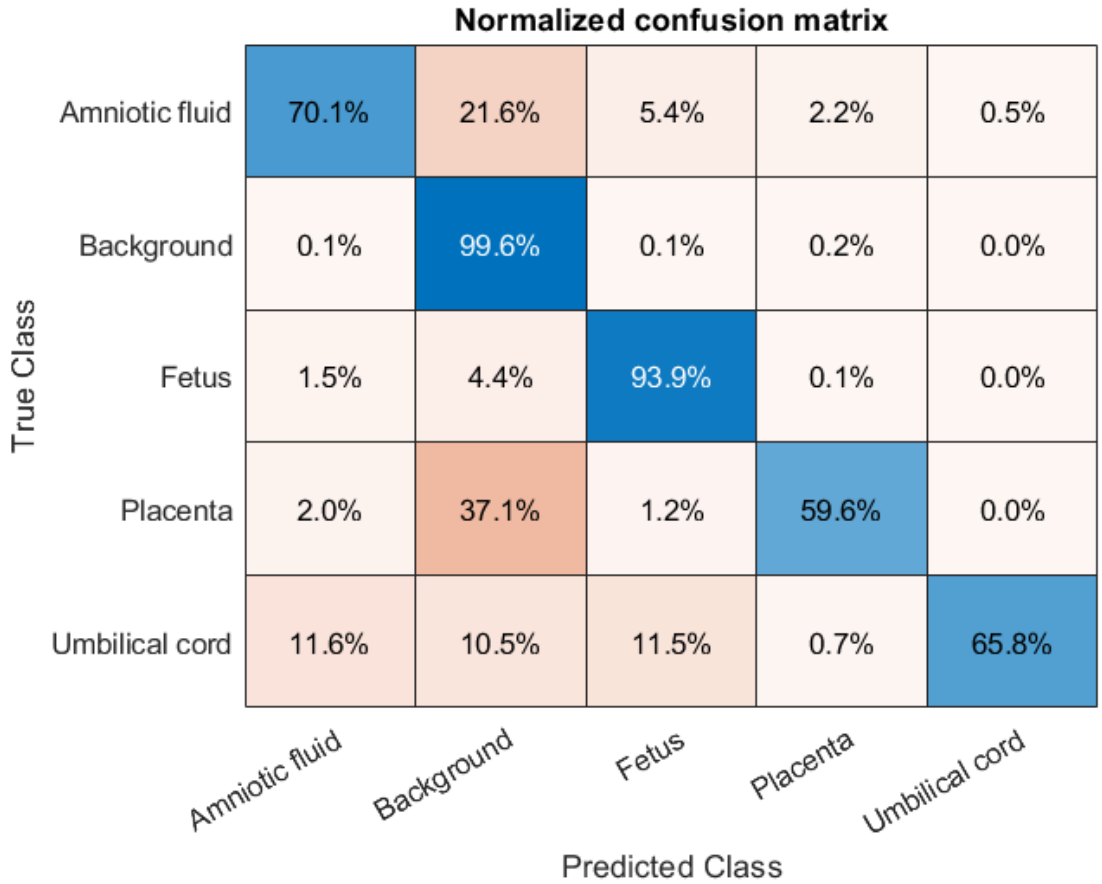

Figure 16: Normalized confusion matrix for the test data that was segmented using the selected model and voting.

Now to the results for the individual five test cases and volume quantifications. 3D-models of the fetuses that have been segmented with the selected model using voting are seen in Figure 17 together with their respective AVD and JSI. More details regarding the performance of the model on the individual cases are visible in Table 8.

Volume estimations and the corresponding absolute volume difference for each test case are presented in Table 9. For the class fetus the mean and median AVD for the five test cases were 4.5 % (std 4.9 percentage points) and 2.3 %, respectively. In comparison, the mean and median AVD were 3.3 % (std 2.7 percentage points) and 2.6 % (min 0.06 %, max 8.6 %) respectively for the 20 validation cases that were evaluated on the selected model with voting from the corresponding fold in the 4-fold cross-validation. A plot of the predicted fetal body volumes versus the ground truth is seen in Figure 18. For the class placenta the mean AVD was 31.2 % (std 34.4 percentage points) and the median AVD was 23.9 % for the test cases. For the 20 validation cases the mean and median AVD were instead 26.6 % (std 25.3 percentage points) and 18.3 %.

AVD: 13.2 % JSI: 0.832	AVD: 2.3 % JSI: 0.948	AVD: 2.0 % JSI: 0.952	AVD: 2.8 % JSI: 0.929	AVD: 2.1 % JSI: 0.949

Figure 17: 3D-models of fetuses from the five test cases – test case 1 to 5 from the left. The segmentations have been generated with the selected model using voting. The absolute volume difference and Jaccard Similarity Index for each fetus is presented.

Table 8: Evaluation of the performance of the selected model with voting on each test case. The performance metrics are the average per-class accuracy and JSI, the accuracy and JSI for the class fetus, and the accuracy and JSI for the class placenta.

Test case	**Mean**		**Fetus**		**Placenta**	
	Accuracy	**JSI**	**Accuracy**	**JSI**	**Accuracy**	**JSI**
1	0.511	0.482	0.849	0.832	0.153	0.151
2	0.829	0.773	0.963	0.948	0.806	0.695
3	0.768	0.768	0.967	0.952	0.327	0.266
4	0.815	0.766	0.978	0.929	0.704	0.666
5	0.863	0.814	0.965	0.949	0.904	0.949

Table 9: Ground truth and estimated volume of the fetuses and placentas in the five test cases. The absolute volume difference (AVD) is also included.

Test case	**Fetus**			**Placenta**		
	Ground truth [ml]	Model [ml]	AVD [%]	Ground Truth [ml]	Model [ml]	AVD [%]
1	3627	3150	13.2	860	142	83.5
2	2358	2304	2.3	456	441	3.3
3	2665	2614	2.0	674	371	45.0
4	2997	3081	2.8	844	642	23.9
5	3441	3369	2.1	878	879	0.1

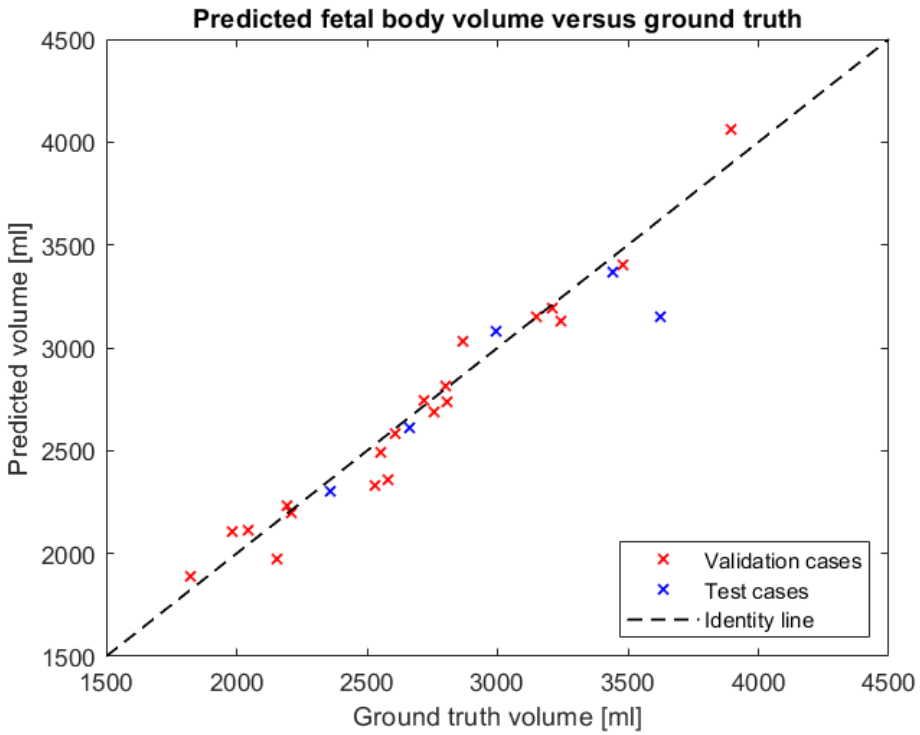

Figure 18: Fetal body volume estimated with model N5 with voting versus ground truth. Red represents the validation cases from the 4-fold cross-validation and blue the test cases. The datapoints should ideally be on the identity line (y=x) meaning the predicted volumes equal the ground truth.

Furthermore, a 3D-model of the fetuses in the twin-case is seen in Figure 19. The segmentation of the twins was generated using the selected model with voting.

Figure 19: 3D-model of twins. The segmentation was created with the selected model using voting.

The average results for the selected model when implementing post-processing on the test cases is seen in Table 10.

Table 10: Results for the evaluation of post-processing methods. For the five test cases the average accuracy and JSI is presented for the classes fetus and placenta, respectively. In all cases, the selected model with voting has been used prior to any post-processing.

Post-processing method	**Fetus**		**Placenta**	
	Average accuracy	**Average JSI**	**Average accuracy**	**Average JSI**
No post-processing	0.939	0.918	0.596	0.544
3D-filling	0.940	0.918	0.596	0.544
Keeping the largest object	0.939	0.918	0.585	0.549
3D-filling + Keeping the largest object	0.939	0.918	0.585	0.549

An overview of the final algorithm for segmentation of fetuses in MR images and quantification of fetal body volume is presented in Figure 20, which illustrates all major steps from input to output.

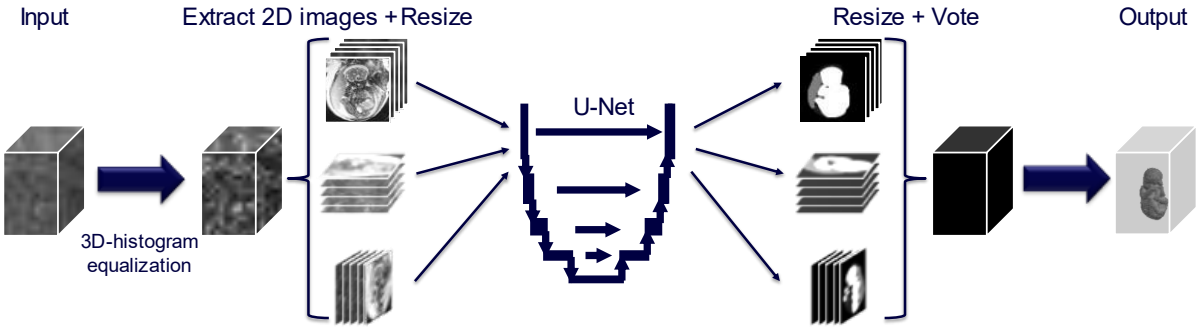

Figure 20: The figure shows an overview of the resulting model. A 3D image is used as input to the model. In the first step, the image intensities are modified as part of the pre-processing. Then, 2D images are extracted from all three dimensions and resized. The images are then fed to the U-Net one at a time – creating 2D masks. Thereafter, the masks are resized, and voting is implemented to select the final voxel labels. As a final step, a 3D segmentation of the class fetus is extracted from the 3D mask – enabling volume quantification.

DISCUSSION

The 4-fold cross-validation showed a uniform result, indicating little dependence on single cases. Even though the hyperparameter settings differed, there was only small differences between the average validation performances of the networks. Furthermore, the standard deviation of the validation performances between the folds was relatively small, see Table 4. This indicates that the achieved performances of the networks are likely not due to randomness and that the measured performance is realistic.

However, as the measure global accuracy is only representative for balanced classes it is important to analyse how the models perform for the individual classes. Since there is a class imbalance where a large proportion of the data is labelled as background, the global accuracy will be biased and strongly depend on the network´s ability to segment the background.

From the 4-fold cross-validation (Table 4), it can be concluded that none of the models were successful at segmenting the placenta. The average placental validation accuracies for the folds were not satisfactory (~ 65 %) and varied between and within the folds.

While the models in the 4-fold cross-validation did not manage to segment the placenta, they were far better at segmenting the fetus. The average validation accuracy for the folds for the class fetus, were above 90 % for all eight models and had a quite low standard deviation between the folds (Table 4).

It was no clear choice which model was best since the models performed almost equally. However, the one with the highest average global validation accuracy was chosen. Although, the end result would probably not have differed much if another model would have been selected instead.

The performance of the selected model on the test cases was, as expected, similar to the validation performance of the same model that had been trained during 4-fold cross-validation. Voting improved the segmentations (Table 5), and especially JSI and BF score, which seems reasonable as this utilizes more image information. Despite the benefits of voting, it is important to be aware of the added time factor, which is a factor of 3. However, as the automated segmentation is performed off-line and not on the scanner and took between half to one and a half minute it was clear that the advantages of voting outweighed the time factor and should be part of the algorithm.

Using voting, the average accuracy and JSI for the class fetus were 93.9 % and 91.8 %, respectively. By looking at the result for the five individual test cases in Table 8, the accuracy was around 96-97 % for all fetuses except for one that had 84.9 % in accuracy. Additionally, the model generally had a much lower performance for that specific case, see Table 8. By looking at the 3D model of the segmented fetus in the left image of Figure 17, it is clear that the algorithm especially failed to segment part of the skull. It should be noted that this particular case was experienced as one of the more difficult datasets to manually delineate and that there was an artifact on part of the head making the delineation more difficult.

The results indicate that the model is both accurate and reliable for quantifying fetal body volume – with a high accuracy and JSI, and low AVD and standard deviation. The results for the fetal body volume estimations can somewhat be compared with the results from studies where fetal weight has been estimated. Nevertheless, it is important to note that the estimated fetal weight includes an additional parameter of uncertainty – namely the density of the fetus which can vary. Therefore, a larger margin of error can be expected when estimating fetal weight. Furthermore, the time between MRI examination and birth influence the accuracy when estimating birth weight. In the studies [4] [33] [34] the average or median percentage error for EFW was between 2 % and 4.8 %. Similarly, the median AVD (or median

percentage error) for fetal body volume estimation was 2.3 % in this study. As can be seen in Figure 18 the performance of the model seems to be independent of the size of the fetus. Due to the good results for the fetus, the algorithm has been integrated in Segment 3DPrint and is thereby available for the Cardiac MR-group to be used for quantification of fetal body volume in both research and clinical practise. The algorithm will be exclusively available to the Cardiac MR-group and to fetal MR images generated in the same manner as the ones included in this study until further testing has been performed. When utilizing the algorithm, visual assessment of the generated segmentation should always be performed to validate and ensure the quality of the prediction before using the quantified fetal body volume. If necessary, adjustments should be made to the segmentation.

The segmentations of the placentas were inconsistent, with accuracies varying between 15 % and 90 % for the test cases. The normalized confusion matrix in Figure 16 provides a clue to the result. On average 37.1 % of the pixels that should have been labelled as placenta were falsely classified as background, which strongly indicates that the algorithm has trouble differentiating the placenta from the background. It is not completely clear why the result for the placenta was worse than for the fetus. Perhaps it is a more challenging imaging segmentation task due to the properties and varying shape and position of the placenta. The features of the fetus might be easier for the model to identify than the ones for the placenta. It would be interesting to investigate this further and compare inter-observer variability.

When selecting which cases to include and exclude, it was sometimes challenging to determine whether the quality of the images was adequate – or if they contained too many artefacts. Which level of quality that is sufficient for inclusion is rather difficult to decide and could be further discussed. To further expand the application of the model, multiple pregnancy cases and cases with noticeable anatomical anomalies could be included in future models. Additionally, cases from other weeks of gestation could be incorporated.

Post-processing did not seem to improve the performance particularly. For the class fetus, this suggests that the model is successful enough to mainly segment the fetus as a single object and to not create false cavities. On the other hand, the post-processing method “Keeping the largest object” seemed to affect the segmentation of the placenta. This could indicate that several objects are created when segmenting the placenta with the model, which was confirmed when visually assessing the predictions.

Beside for quantification of fetal body volume, the method presented in this study could be used for other purposes such as 3D-modelling and 3D-printing of the fetus. For this reason, it would be interesting to investigate post-processing further. Post-processing could perhaps be used for other purposes – for example to generate more visually appealing 3D-models using smoothing.

In this study, manual delineations were considered as ground truth. For the manual delineations, it would have been interesting to investigate the inter-observer variability and compare it with the automatic segmentations. Furthermore, this study focused on volume quantification of the fetus and placenta, since it is of clinical relevance and was requested by the physicians and researchers in the Lund Cardiac MR group. However, it would be interesting to explore volume quantification of the amniotic fluid further as it also can be of clinical interest.

During this project, a single network architecture has been used. Perhaps other architectures could have yielded better results. Furthermore, the optimal set of hyperparameters might not have been found during tuning of hyperparameters. This leaves room for further improvement of the model. It would for instance be interesting to investigate how a 3D U-Net performs on the task, since more spatial information is used than for the 2D U-Net.

Perhaps another loss function, such as dice loss, could have been used instead of cross entropy error to make up for the class imbalance. However, the excellent result in AVD indicates that although the class balance the segmentation have no clear bias. In addition, it the Jaccard Similarity Index possibly could have been used as evaluation metric when tuning the model.

Furthermore, it would be interesting to use data from various sites, and data that has been collected using different MRI protocols and scanners from different vendors since the data used in this study was collected at a single hospital, using one specific acquisition protocol. Perhaps this could make the model even more robust.

Ideally, all data would have been manually delineated before training, validating, or testing any of the deep neural networks. Then, the data could have been randomly split into training data, validation data and test data, with no overlapping or bias on how the data was split. However, in this case the delineations were not complete when starting this study. Instead, the manual delineations were performed in parallel with training and tuning the networks. It was first unsure how many datasets that would be possible to include. Therefore, the manual delineations were initially performed on the datasets that, by visual assessment, looked easiest to manually delineate. To make use of time, initial tuning of hyperparameters was performed using the delineated datasets available at that time. The datasets that were last to be delineated could then have been used for testing. However, the non-random order of delineation could in that case affect the estimated performance of the networks, introducing bias. To avoid having the data split into parts considered more or less difficult, the data was randomly split before starting k-fold cross-validation. However, this meant that some of the datasets that had been used previously would now be used for testing. This could lead to an overly optimistic estimate of generalization performance. It is hard to say how much this will influence the results and reliability of the results, especially since the overlap was for initial tuning of hyperparameters and that the final model was solely selected based on the result from the k-fold cross-validation, from which there was no overlap between training, validation, and testing data.

CONCLUSION

The aim of creating a fully automatic model was fulfilled. However, the model failed to segment the placenta and estimate its volume in a satisfactory manner. The model was considerably better at segmenting the fetus and was successful at quantifying fetal body volume. Furthermore, the results indicate that the method is both accurate and reliable for quantifying fetal body volume. Nevertheless, it would be interesting to test the model on MR images from other sites. The algorithm is integrated in Segment 3DPrint, and thereby available for the Cardiac MR-group to be used for quantification of fetal body volume in both research and clinical practise. This will most likely save time and resources and enable further implementation of fetal body volume estimation. In the end, this could lead to improved diagnostics.

REFERENCES

[1] S. N. Saleem, "Fetal MRI: An approach to practice: A review," *Journal of Advanced Research,* vol. 5, pp. 507-523, 2014.

[2] Prayer, Daniela; SpringerLink (Online service), Fetal MRI, Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 230, 348.

[3] A. Ljubic, A. Cetkovic, A. Novakov Mikic, J. Dukanac Stamenkovic, I. Jovanovic, T. Stosic Opincal and D. Damnjanovic, "Ultrasound vs MRI in Diagnosis of Fetal and Maternal Complications," *Donald School Journal of Ultrasound in Obstetrics and Gynecology,* vol. 5, no. 3, pp. 231-242, 2011.

[4] P. N. Baker, I. R. Johnson, P. A. Gowland, J. Hykin, P. R. Harvey, A. Freeman, V. Adams, B. S. Worthington and P. Mansfield, "Fetal weight estimation by echo-planar magnetic resonance imaging," *Lancet,* vol. 343, no. 8898, pp. 644-645, 12 March 1994.

[5] J. Uotilia, P. Dastidar, T. Heinonen, P. Ryymin, R. Punnonen and E. Laasonen, "Magnetic resonance imaging compared to ultrasonography in fetal weight and volume estimation in diabetic and normal pregnancy," *Acta obstetricia et gynecologica Scandinavica,* vol. 79, no. 4, pp. 255-259, April 2000.

[6] C. Kadji, M. Camus, M. Klass, S. Fellas, V. Cecotti, V. Dütemeyer, J. C. Jani, M. M. Cannie and R. De Angelis, "Prenatal prediction of postnatal large-for-dates neonates using a simplified MRI method: comparison with conventional 2D ultrasound," *Ultrasound in Obstetrics and Gynecology,* vol. 52, no. 2, pp. 250-257, 2018.

[7] M. Seed, J. F. P. van Amerom, S.-J. Yoo, B. Al Nafisi, L. Grosse-Wortmann, E. Jaeggi, M. S. Jansz and C. K. Macgowan, "Feasibility of quantification of the distribution of blood flow in the normal human fetal circulation using CMR: a cross-sectional study," *Journal of Cardiovascular Magnetic Resonance,* vol. 14, no. 79, 2012.

[8] M. Prsa, L. Sun, J. van Amerom, S.-J. Yoo, L. Grosse-Wortmann, E. Jaeggi, C. Macgowan and M. Seed, "Reference Ranges of Blood Flow in the Major Vessels of the Normal Human Circulation at Term by Phase-Contrast Magnetic Resonance Imaging," *Criculation: Cardiovascular Imaging,* vol. 7, pp. 663-670, 2014.

[9] R. A. Kubik-Huch, S. Wildermuth, L. Cettuzzi, A. Rake, B. Siefert, R. Chaoui and B. Marincek, "Fetus and Uteroplacental Unit: Fast MR Imaging with Three-dimensional Reconstruction and Volumetry - Feasibility Study," *Radiology,* vol. 219, no. 2, pp. 567-573, 2001.

[10] R. A. Kubik-Huch, S. Wildermuth, L. Cettuzzi, A. Rake, B. Seifert, R. Chaoui, N. Teodorovic and B. Marincek, "Fetal Volumetry Based on 3D Reconstructions of 2D Single-Shot Fast Spin-Echo MR Data Sets," *Proc. Intl. Soc. Mag. Reson. Med,* vol. 9, p. 591, 2001.

[11] C. Kadji, M. De Groof, M. F. Camus, R. De Angelis, S. Fellas, M. Klass, V. Cecotti, V. Dütemeyer, E. Barakat, M. M. Cannie and J. C. Jani, "The Use of a Software-Assisted Method to Estimate Fetal Weight at and Near Term Using Magnetic Resonance Imaging," *Fetal Diagnosis and Therapy,* vol. 4, pp. 307-313, 2017.

[12] C. Kadji, M. F. Camus, E. Bevilacqua, M. M. Cannie, T. Cos Sanchez and J. C. Jani, "Repeatability of estimated fetal weight: Comparison between MR imaging versus 2D ultrasound in at- and near-term patients," *European Journal of Radiology,* vol. 91, pp. 35-40, 2017.

[13] J. Torrents-Barrena, G. Piella, N. Masoller, E. Gratacós, E. Eixarch, M. Ceresa and M. Ángel González Ballester, "Fully automatic 3D reconstruction of the placenta and its peripheral vascularity in intrauterine fetal MRI," *Medical Image Analysis,* vol. 2019, no. 54, pp. 263-279, 2019.

[14] O. Ronneberg, P. Fischer and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *Springer Verlag (Lecture Notes in Computer Science),* 2015.

[15] F. Isensee, P. F. Jaeger, S. A. A. Kohl, J. Petersen and K. H. Maier-Hein, "Automated Design of Deep Learning Methods for Biomedical Image Segmentation," 2019.

[16] J. Bogaert, S. Dymarkowski, A. M. Taylor and V. Muthurangu, Eds., Clinical Cardiac MRI, 2nd ed., Vols. Medical Radiology - Diagnostic Imaging, Springer-Verlag Berlin Heidelberg, 2012.

[17] I. Goodfellow, Y. Bengio and A. Courville, Deep Learning, MIT Press, 2016.

[18] D. Zhang and SpringerLink (Online service), Fundamentals of Image Data Mining: Analysis, Features, Classification, 1st ed., Springer Nature Switzerland AG, 2019, pp. 207-208, 226, eq. 9.24.

[19] G. Litjens, T. Kooi, B. E. Bejnordi, A. A. A. Setio, F. Ciompi, M. Ghafoorian, J. A. W. M. van der Laak, B. van Ginneken and C. I. Sánchez, "A survey on deep learning in medical image analysis," *Medical Image Analysis,* vol. 42, pp. 60-88, 2017.

[20] S. Soffer, A. Ben-Cohen, O. Shimon, M. M. Amitai, H. Greenspan and E. Klang, "Convolutional Neural Networks for Radiologic Images: A Radiologist's Guide," *Radiology,* vol. 290, no. 3, pp. 590-606, 2019.

[21] F. Altaf, S. M. S. Islam, N. Akhtar and N. K. Janjua, "Going Deep in Medical Image Analysis: Concepts, Methods, Challenges and Future Directions," 2019.

[22] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," 2017 (Published as a conference paper at ICLR 2015).

[23] "Crossentropy," The Mathworks, Inc., [Online]. Available: https://se.mathworks.com/help/deeplearning/ref/dlarray.crossentropy.html. [Accessed 20 03 2021].

[24] D. Müller and F. Kramer, "MIScnn: A Framework for Medical Image Segmentation with Convolutional Neural Networks and Deep Learning," 2019.

[25] "unetLayers," The Mathworks, Inc., [Online]. Available: https://se.mathworks.com/help/vision/ref/unetlayers.html. [Accessed 21 March 2021].

[26] I. Rizwan I Haque and J. Neubert, "Deep learning approaches to biomedical image segmentation," *Informatics in Medicine Unlocked,* vol. 18, 2020.

[27] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn and A. Zisserman, "The Pascal Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision,* vol. 88, no. 2, pp. 303-338, 2010.

[28] G. Csurka, D. Larlus and F. Perronnin, “What is a good evaluation measure for semantic segmentation?,” in *BMVC 2013 - Electronic Proceedings of the British Machine Vision Conference 2013*, Bristol, United Kingdom, 2013.

[29] “ConfusionMatrixChart Properties,” The MathWorks, Inc., [Online]. Available: https://se.mathworks.com/help/stats/mlearnlib.graphics.chart.confusionmatrixchart-properties.html. [Accessed 01 04 2021].

[30] P. Zhang, Y. Deng, X. Tang, Y. Zhong and X. Li, “A Survey on Deep Learning of Small Sample in Biomedical Image Analysis,” 2019.

[31] L. Zhang, X. Wang, D. Yang, T. Sanford, S. Harmon, B. Turkbey, H. Roth, A. Myronenko, D. Xu and Z. Xu, “When Unseen Domain Generalization is Unnecessary? Rethinking Data Augmentation,” 2019.

[32] The Mathworks Inc., MATLAB, ver. R2019a (9.6.0.1174912), 2019.

[33] L. Lo Kadji, C. Kadji, M. Cannie, Y. Kacem, B. Strizek, M. Mbonyumutwa, F. Wuyts and J. Jani, “Determination of fetal body volume measurement at term with magnetic resonance imaging: effect of various factors,” *The Journal of Maternal-Fetal & Neonatal Medicine*, vol. 26, no. 12, pp. 1254-1258, 2013.

[34] K. Liao, L. Tang, C. Peng, L. Chen, R. Chen, L. Huang, P. Liu and C. Chen, “A modified model can improve the accuracy of foetal weight estimation by magnetic resonance imaging,” *European Journal of Radiology*, vol. 110, pp. 242-248, 2019.

Master's Theses in Mathematical Sciences 2021:E9
ISSN 1404-6342

LUTFMA-3439-2021

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

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