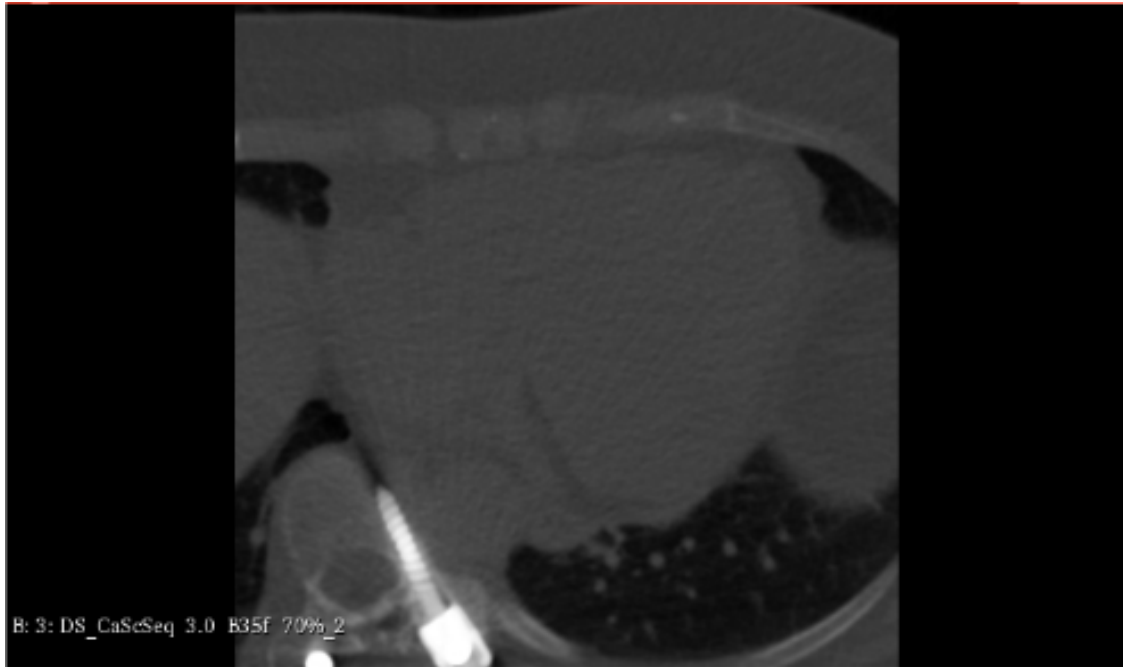




**CHALMERS**  
UNIVERSITY OF TECHNOLOGY



# Identifying outlier data in CT images using convolutional neural networks

Implementation of a 3D Autoencoder

Master's thesis in Biomedical Engineering

Cajsa Hjohlman

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY

Gothenburg, Sweden 2022

[www.chalmers.se](http://www.chalmers.se)



MASTER'S THESIS 2022

# Identifying outlier data in CT images using convolutional neural networks

Implementation of a 3D Autoencoder

CAJSA HJOHLMAN



**CHALMERS**  
UNIVERSITY OF TECHNOLOGY

Department of Electrical Engineering  
CHALMERS UNIVERSITY OF TECHNOLOGY  
Gothenburg, Sweden 2022

Identifying outlier data in CT images using convolutional neural networks  
Implementation of a 3D Autoencoder  
Cajsa Hjohlman

© CAJSA HJOHLMAN, 2022.

Supervisor: Ida Häggström, Electrical Engineering  
Examiner: Fredrik Kahl, Electrical Engineering

Master's Thesis 2022  
Electrical Engineering  
Chalmers University of Technology  
SE-412 96 Gothenburg  
Telephone +46 31 772 1000

Cover: An axial center slice of an identified outlier CT image.

Typeset in  $\LaTeX$   
Printed by Chalmers Reproservice  
Gothenburg, Sweden 2022

Identifying outlier data in CT images using convolutional neural networks  
Cajsa Hjohlman  
Department of Electrical Engineering  
Chalmers University of Technology

---

## Abstract

Deep learning can be a useful tool when working with medical images for tasks such as classification or segmentation. One issue affecting model performance both when training and when implementing a deep learning network is the existence of outlier images. This project aims to address this issue by developing a deep learning model for outlier detection in CT images. A 3D autoencoder is implemented with a reconstruction task, with the hypothesis that the model will be worse at reconstructing outlier images due to unlearned outlier features. The trained model is able to reconstruct images adequately, although without detail.

Outliers are identified via a masked reconstruction error, resulting in 6 out of 14 outliers with visually identifiable outlier causes such as an artificial heart valve and spinal screws. A feature analysis shows low correlation between features and reconstruction errors, which signifies the need for improving the autoencoder's reconstruction performance. There is instead a high correlation between features and image sizes, also visible in the reconstruction performance of the model. This image size bias is likely due to an unbalanced dataset. The project shows promising results for the hypothesis of outlier detection via reconstruction error, and the method can be greatly improved with an improved reconstruction performance and reduced image size bias.

## Acknowledgements

First, I would like to thank my supervisor Ida Häggström for guidance and for always being available for questions and discussions, this project would not have happened without you. I would also like to thank Edvin, Linn and Sara for advice during the writing process. Lastly, I am thankful to Elias and Emma for all the support you have provided throughout the project.

Cajsa Hjohlman, Gothenburg, June 2022



# List of Acronyms

AE	Autoencoder
ANN	Artificial Neural Network
CNN	Convolutional Neural Networks
CT	Computed Tomography
DBSCAN	Density-Based Spatial Clustering of Applications with Noise
DL	Deep Learning
GAN	Generative Adversarial Networks
HU	Hounsfield Units
$MSE_m$	Masked reconstruction error, (MSE calculated for masked images)
MSE	Mean squared error
NN	Neural Network
OD	Outlier Detection
SGD	Stochastic Gradient Descent
SNIC	Swedish National Infrastructure for Computing
UMAP	Uniform Manifold Approximation and Projection
VAE	Variational Autoencoder



# Contents

<b>List of Acronyms</b>	<b>vi</b>
<b>List of Figures</b>	<b>xi</b>
<b>List of Tables</b>	<b>xiii</b>
<b>1 Introduction</b>	<b>1</b>
1.1 Related work . . . . .	2
1.2 Scope . . . . .	3
<b>2 Theory</b>	<b>4</b>
2.1 Traditional outlier detection . . . . .	4
2.2 Deep learning . . . . .	5
2.2.1 Convolutional neural networks . . . . .	6
2.2.2 Model training . . . . .	7
2.3 Deep learning outlier detection for medical image analysis . . . . .	7
2.3.1 Autoencoder . . . . .	8
2.4 Computed Tomography . . . . .	8
2.5 SCAPIS . . . . .	9
<b>3 Methods</b>	<b>10</b>
3.1 Project resources . . . . .	10
3.2 Preprocessing data . . . . .	10
3.3 Model architecture . . . . .	11
3.4 Model training . . . . .	12
3.5 Outlier detection via reconstruction errors . . . . .	13
3.6 Model evaluation . . . . .	13
<b>4 Results</b>	<b>14</b>
4.1 Reconstruction performance . . . . .	14
4.2 Detected outliers via MSEM threshold . . . . .	16
4.3 Feature analysis . . . . .	22
<b>5 Discussion</b>	<b>26</b>
5.1 Autoencoder reconstruction improvements . . . . .	26
5.1.1 Image size bias . . . . .	27
5.2 Detected outliers . . . . .	27

5.3	Feature analysis . . . . .	28
5.4	Future work . . . . .	29
<b>6</b>	<b>Conclusion</b>	<b>30</b>
<b>A</b>	<b>Appendix A</b>	<b>I</b>

# List of Figures

2.1	An artificial neural network with input layer, several hidden layers, and an output layer. . . . .	5
2.2	A schematic figure of a 2D fully connected layer. Each output value is the scalar product between the input and a different set of layer weights. . . . .	5
2.3	A schematic figure of a 2D convolutional layer with one kernel of size 3x3. Each output value is the scalar product between a small section of the input and the kernel. . . . .	6
2.4	A schematic figure of an autoencoder consisting of an encoder that reduces the input dimensions to a feature vector, and a decoder that reconstructs the input from the feature vector. . . . .	8
3.1	The encoder architecture divided into blocks. Each block consists of several color coded layers, see bottom of figure for layer types. The input is an original image $I$ of size 1x87x87x273, and the output is a feature vector $Z$ of size 1024x1x1x1. The dimensions correspond to [ <i>features</i> x <i>width</i> x <i>height</i> x <i>depth</i> ]. The third block has the same architecture as the second block aside from a higher number of features $f$ . . . . .	11
3.2	The decoder architecture divided into color coded layers, see bottom of figure for layer types. The input is a feature vector $Z$ of size 1024x1x1x1 and the output is a reconstructed image $\tilde{I}$ of size 1x87x87x273. The dimensions correspond to [ <i>features</i> x <i>width</i> x <i>height</i> x <i>depth</i> ]. . . . .	12
4.1	Axial slice of an original image from the testing dataset along with corresponding reconstruction and the absolute difference between the two. . . . .	14
4.2	MSEs as a function of the pixel counts for the test dataset. . . . .	15
4.3	Reconstruction errors against max pixel values. . . . .	15
4.4	Reconstruction errors against mean pixel values. . . . .	15
4.5	$MSE_m$ s as a function of the pixel counts for the test dataset. . . . .	16
4.6	The masking of scan 148 from the testing dataset in a center axial slice. The white pixels in the mask show pixel indices that are included in the $MSE_m$ calculation. . . . .	16
4.7	A histogram over the validation datasets $MSE_m$ s. The vertical line at $MSE_m = 0.2710$ is calculated according to equation 3.4. . . . .	17

---

4.8	A histogram over the testing datasets $MSE_m$ . The vertical line at $MSE_m = 0.2710$ is the threshold for outlier identification. . . . .	17
4.9	Histogram of the 14 identified outliers pixel values. . . . .	19
4.10	Histogram of top 10 inliers pixel values. . . . .	19
4.11	Original and reconstructed images of outlier C, and the absolute error between the two. . . . .	20
4.12	Original and reconstructed images of outlier E, and the absolute error between the two. . . . .	20
4.13	Original and reconstructed images of outlier G, and the absolute error between the two. . . . .	21
4.14	Original and reconstructed images of outlier I, and the absolute error between the two. . . . .	21
4.15	Original and reconstructed images of outlier K, and the absolute error between the two. . . . .	22
4.16	Original and reconstructed images of outlier N, and the absolute error between the two. . . . .	22
4.17	UMAP dimension reduction of the feature vectors, colored by pixel counts. . . . .	23
4.18	UMAP dimension reduction of the feature vectors, colored by $MSE_m$ . . . . .	23
4.19	Identified outliers features marked in the UMAP feature plot. . . . .	24
4.20	UMAP dimension reduction of the feature vectors clustered using DBSCAN. . . . .	24
4.21	Normalized pixel count histograms for all three clusters. . . . .	25
4.22	Normalized $MSE_m$ histograms for all three clusters. . . . .	25
A.1	Slices in three orthogonal planes of outlier C. . . . .	I
A.2	Slices in three orthogonal planes of outlier E. . . . .	I
A.3	Slices in three orthogonal planes of outlier G. . . . .	I
A.4	Slices in three orthogonal planes of outlier I. . . . .	II
A.5	Slices in three orthogonal planes of outlier K. . . . .	II
A.6	Slices in three orthogonal planes of outlier N. . . . .	II

# List of Tables

2.1	Table of HU units [1]. . . . .	9
3.1	Hyperparameters used for training the AE. . . . .	12
4.1	Mean values of entire training dataset. . . . .	17
4.2	Identified outliers $MSE_{ms}$ , mean pixel values and max pixel values. . .	18

# 1

## Introduction

The medical sector is one of many fields with an increasing interest in implementing deep learning (DL) models. As the medical field has become more digitized, more data is being created, and computational resources are increasing, the possibilities to incorporate DL models in different areas has expanded. One such example is for medical image analysis, where medical images can be analyzed using DL models instead of by clinicians, aiming to increase both efficiency and accuracy of the practice [2]. DL models can perform tasks such as image segmentation, image classification, and object detection, which can be used for for example detecting or predicting diseases [2].

A common task for DL models in medical image analysis is for outlier detection (OD) [3]. Outliers are data points that stands out from the majority of the data, which can be referred to as the inliers [3]. Outliers can be either a single data point, or a small group [4]. It is not always the case that strict classification of inliers or outliers is used, but rather an outlier score to evaluate to what extent a data sample is an outlier [5]. In a supervised setting OD is a classification task, while OD in an unsupervised setting is more difficult, depending on how outliers are defined.

In the case of image analysis, outliers are images that have features that stand apart from the majority of images. One cause for a medical image to be an outlier can be deviating patient anatomy, possibly caused by a sickness [4]. Another cause can be variations during the imaging process, for instance the positioning of the patient or noise in the environment [6, 3]. The issue of outliers in a set of medical images can be two fold. Firstly, the existence of outliers in a set of training data can cause longer training time and reduce the performance of the DL model [2]. Secondly, a fully trained DL model can not be expected to perform well on images that deviate from the data it has been trained on [7].

To address these issues, this projects aims to develop an unsupervised OD model for medical images. The aim is to use the model for preprocessing training data, as well as flagging images that are unfit for being processed by a DL model. The approach taken here is to create a 3D autoencoder (AE) with the task to encode and then reconstruct images. The network is expected to learn to reconstruct typical features of the inlier images, while it is assumed that outlier images will be harder to reconstruct due to outlier specific features, unrepresented in the training dataset.

This relates high reconstruction errors to outlier images, creating a method for identifying outliers.

## 1.1 Related work

Several studies have been done using DL for OD in medical images, usually using unsupervised learning implementing either AEs, Generative Adversarial Network (GAN), or combinations of the two [6, 8, 9, 10, 11]. As with most unsupervised models, they are trained on unlabeled data, but then evaluated on a set of labeled data [5]. Some studies have focused specifically on handling 3D medical images, and others highlight 3D networks as a future step to improve model performance [10, 11, 6].

Nakao et al. [6], successfully combined a variational autoencoder (VAE) and a GAN network for OD in chest X-ray images. They point out that a downside of using only a VAE for OD can be reduced resolution of the images. To compensate for this, during training, they include a discriminator that improves the decoders reconstruction ability and a code discriminator to shift the latent vectors towards a Gaussian distribution.

Guo et al. [8] combine a cascade VAE with a discriminator. The encoder and decoder are separated into two parts each, and a halfway latent vector is extracted between the encoder parts. This halfway latent vector is then input into a second VAE. The purpose of having multiple networks is that they can extract different level features, which are then combined to reconstruct the image. A discriminator is used to distinguish between reconstructed and original images, and outliers are identified based on the discriminators results along with the reconstruction errors.

Another recent study combining AE and GAN models was done by Zhang et al. [9]. They presented the Improved Adversarial Autoencoder. It consists of an autoencoder with a chain of convolution block, and a discriminator. The discriminator is used to classify between original and generated images, and also to calculate latent vectors for the original and generated images. Evaluating their model on several datasets, including a set of CT images, they show that the AE model outperforms several other models at detecting outliers.

Cheng et al. [12], presented a Transformation Invariant Autoencoder, and highlight two central aspects when training. First, that they used rotated images for training the AE. Second, that they used self-paced learning, meaning that in each epoch they identify inliers and use only them as training data in the next epoch, avoiding training on outliers. This was to hinder the model from learning to reconstruct outlier images, as this would reduce the models ability to separate outliers and inliers. They concluded an improved outlier detection of 10% compared to several other AE models.

The issue of small medical image datasets was addressed by Romero et al. [13]. One method of increasing the amount of training data is to use data augmentation,

varying images slightly by for example rotation, cropping or flipping. They also propose the use of transfer learning, where the model is first trained on a "source" task, often times images from ImageNet, and then re-trained on the "target" task, the actual images the model is aimed at processing. During target training some weights can be kept constant from the source training. They concluded that best results were achieved using source images anatomically similar to the target images.

Unlike previously mentioned studies that focus on 2D images, Simarro Viana et al. [10] present a model for OD in 3D images. Their model is based on the 2D f-AnoGAN model, combining a GAN network and an encoder, where the generator of the GAN network generates images from the latent space created by the encoder. The 3D model showed a 4% improved performance compared to the 2D model.

To address the topic of 3D medical image analysis, Chen et al. [11], presented a set of Med3D models. These models are 3D versions of 2D ResNet models such as ResNet10, ResNet18, ResNet34 and ResNet50. The models have been trained on 3DSeg-8, a dataset of 3D medical images. The purpose of the Med3D models was to create pretrained 3D models that could be transferred to medical imaging tasks such as segmentation and classification. They show that pretraining the models on 3DSeg-8 improved both model accuracy and training time for medical image segmentation and classification tasks, compared to pretraining on non-medical datasets.

## 1.2 Scope

Unlike several previous studies the aim of this project will not be to identify outliers with a specific pathological cause, but rather to define outliers in a set of images independent of the underlying cause. Visual assessment will not be done by any clinicians. No patient information other than the CT images will be used.

Only unsupervised learning will be used for model training since the provided data is unlabeled. The models will be based on convolutional neural network (CNN) structures. Limitations for model improvements include the size of the set of training data as well as computational power and the time frame of the project.

# 2

## Theory

This section covers basic information on deep learning, medical image outlier detection, and CT imaging, as well as resources used for this project.

### 2.1 Traditional outlier detection

There are several different methods for detecting outliers. A rule of thumb is that points that deviate from the dataset mean with more than three times the standard deviation can be classified as outliers [14]. More advanced, traditional methods for OD in data include density-based methods, statistical methods, and clustering methods [14]. Density based methods calculate a points density, for example using the average distance to its nearest neighbors, and compares it to the density of its neighbors. Statistical based methods estimate a distribution of the data, fit the distribution to the data and then identify data that does not fit to the model as outliers. Clustering methods use a clustering algorithm to group together data, and points not assigned a cluster can be defined as outliers.

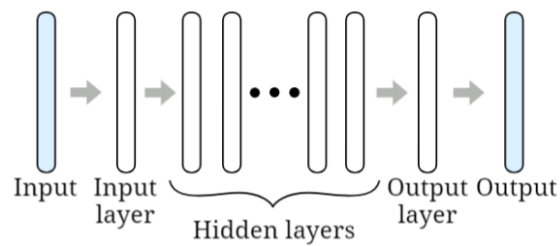
One common clustering method is Density-Based Spatial Clustering of Applications with Noise (DBSCAN) [15]. The DBSCAN algorithm requires two parameters, a distance  $eps$  and a minimal number of points  $minPts$ . The algorithm starts with picking out one point, and if more than  $minPts$  points are within the radius  $eps$  of the point, also referred to as the  $eps$ -neighborhood of the point, the point is labeled as a core point and a cluster is initiated. If not, the point is labeled as a noise point and the algorithm moves on to a new point until a core point is found. When a core point is found, all points within its  $eps$ -neighborhood are added to the cluster, and points belonging to any of these points  $eps$ -neighborhoods are also added to the cluster. This can result in a set of noisy points that have not been assigned a cluster, and can be labeled as outliers.

Although DBSCAN can cluster high dimensionality data, a dimension reduction can be done prior to the clustering. One method of doing this is using Uniform Manifold Approximation and Projection (UMAP) [16]. The UMAP algorithm assumes the data to be uniformly distributed on a manifold and constructs a high dimensional topological representation of the data. Then a low dimensional topological representation of the data is adjusted so that the cross-entropy between the two

representations is minimized [16].

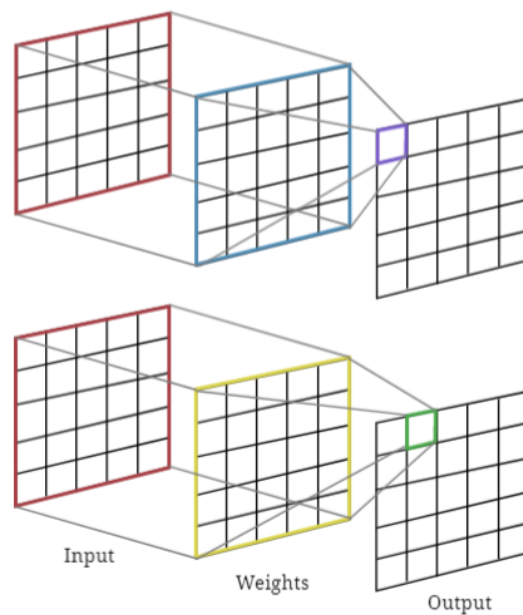
## 2.2 Deep learning

Artificial neural networks (ANN) are machine learning (ML) models inspired by the system of neurons sending signals in the human brain [17]. An ANN consists of an input layer, a set of hidden layers, and a final output layer, each layer consisting of a set of parallel neurons [17], see figure 2.1.



**Figure 2.1:** An artificial neural network with input layer, several hidden layers, and an output layer.

Fully-connected layers are the backbone of traditional ANNs [18]. In a fully connected layer every neuron in the layer is directly connected to all neurons in the previous layer. Due to the high number of layer parameters they are computationally expensive to train [18]. A schematic example of a 2D fully connected layer is presented in figure 2.2. The output array values are the scalar product of the input array and a different set of layer weights.

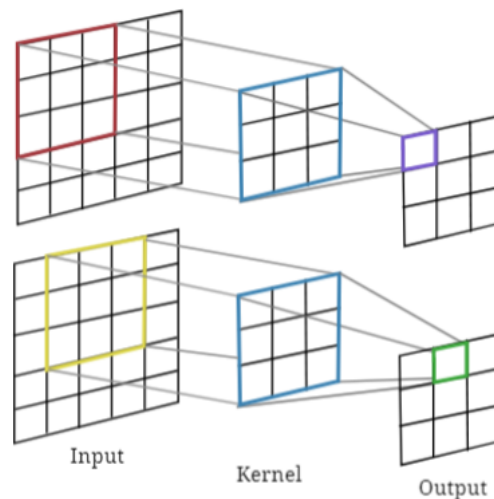


**Figure 2.2:** A schematic figure of a 2D fully connected layer. Each output value is the scalar product between the input and a different set of layer weights.

DL networks are ANNs that are deep in the sense that they have many hidden layers [18]. Convolutional neural network (CNNs) use convolutional layers, and are very common for image analysis tasks due to their spatial processing of images.

### 2.2.1 Convolutional neural networks

CNNs are useful for image analysis since they find patterns in the input image, so called features, while being computationally inexpensive [18]. This is done by the use of convolutional layers. In a convolutional layer the neurons make up one or several convolutional kernels, which are convolved with the layer input [17], see Figure 2.3. The kernel is moved across the image and the choice of zero padding and stride decides the size of the output image. The kernels act as filters, sensitive to specific features [2]. As an example, a CNN that has been trained on images of cars, will contain convolutional kernels that are sensitive for features such as the shape of a car tire or exhaust pipe.



**Figure 2.3:** A schematic figure of a 2D convolutional layer with one kernel of size  $3 \times 3$ . Each output value is the scalar product between a small section of the input and the kernel.

Aside from convolutional layers, it is common for CNNs to also have fully-connected layers, pooling layers, and batch normalization layers [2]. Pooling layers are used to downsample the layer input to reduce the dimension of the network [18]. Batch normalization normalizes the output of a layer, which accelerates and stabilizes the learning process [2]. The input layer design depends on the input data, the number of neurons must match the size of the input vector. The output layer of the network is designed depending on what the task of the CNN is. For example for a binary classification task, the output layer can consist of two neurons, each giving an output corresponding to the respective classes.

Neurons in a CNN have connections to neurons in the previous and next layer, and each connection has an individual weight [17]. The weighted inputs to a neuron are

summed, and fed through an activation function to calculate the output signal of the neuron to the next layer [17]. For image analysis it is common to use a non-linear activation function such as ReLu [2]. The sizes of the weights affect what information is sent through the network, and the process of adjusting the weights is done during training.

### 2.2.2 Model training

Training an ANN is an iterative, gradient based process where the model adjusts its internal weights to minimize a cost function. Depending on what kind of data is used and the task at hand, different kinds of learning can be implemented. Supervised learning requires that the training data has corresponding labels, and the NN learns to map the input data to its corresponding label [18]. The model can then be used for classification tasks of unlabeled data. Unsupervised learning on the other hand uses unlabeled data, and instead learns to extract features from the input [10]. Unsupervised learning is useful for tasks such as clustering.

The process of training is done by passing the data through the network in batches. For each batch, a loss is calculated based on a defined cost function. Then the model gradients are calculated using back-propagation. The model weights are then updated according to the gradients using an optimizer method so that the defined loss is minimized. One common optimization method is stochastic gradient descent (SGD), where the model weights  $\bar{w}$ , are adjusted based on the loss  $Q(\bar{w})$ , and a learning rate  $\eta$ , according to equation 2.1,

$$\bar{w} := \bar{w} - \eta Q(\bar{w}). \quad (2.1)$$

One pass of all the data is referred to as an epoch, and training usually requires many epochs before the loss converges to a final value. Hyperparameters are preset parameters and includes number of epochs, batch size and learning rate.

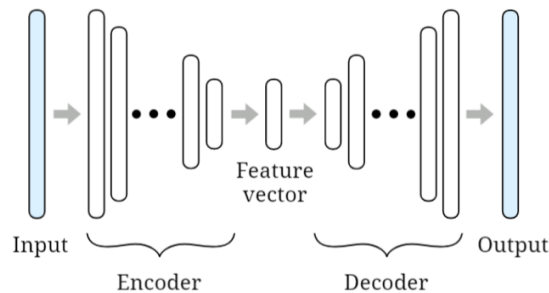
## 2.3 Deep learning outlier detection for medical image analysis

The ability DL models have of learning image representations is effective for many implementations, such as medical image analysis. Better processing times and larger sets of data are both reasons why DL methods are becoming increasingly favorable in medical image analysis [3]. Although, an obstacle in implementing DL models for medical image analysis is the limited availability of training data [3], usually only having access to small datasets [13]. This is partly due to the extensive and costly process of acquiring medical images [2]. Acquiring labeled data is extra costly, since this requires manual annotation by someone with medical knowledge [2]. Therefore, the amount of labeled data is especially scarce, which has led to a wide adoption of semi- and unsupervised learning methods [4]. Most common are unsupervised DL

models such as AEs and GAN networks, utilizing a CNN structure with convolutional filters for feature extraction [4, 14].

### 2.3.1 Autoencoder

AE networks consist of an encoder part that compresses the input image to a low dimensional feature vector, and a decoder part that reconstructs an image from the feature vector [4], see Figure 2.4.



**Figure 2.4:** A schematic figure of an autoencoder consisting of an encoder that reduces the input dimensions to a feature vector, and a decoder that reconstructs the input from the feature vector.

The AE is trained with the goal to minimize the difference between the original image and the reconstructed image, known as the reconstruction loss [3], or the restoration error [12]. If the AE can successfully reconstruct an image from the latent vector with only a small reconstruction error, then the latent vector contains all essential information of the image, and the AE has learned the essential features of the image. An example of a more advanced AE is the VAE, where the latent vector is reduced to a vector of distribution parameters that estimate the distribution of the input images [3].

AEs can be used for OD by analyzing the reconstruction errors. It is expected that the AE will be worse at reconstructing outlier images, since these will contain features that the AE has not learned, resulting in larger reconstruction losses for outlier images [4, 14, 12].

## 2.4 Computed Tomography

Computed tomography (CT) scans use an X-ray source to send X-rays through the body, and one or several detectors on the opposite side [19]. The source and detectors are rotated around the body, measuring how much of the beam is absorbed in the tissue it passes [19]. From this it is possible to calculate a 2D image, where each pixel shows the absorption of the corresponding 3D voxel of tissue [19]. The measurements are repeated several times, successively moving the body forward between measurements to get a set of 2D images, representing slices of the 3D body

[19]. CT image pixel values are measured in Hounsfield Units (HU), which relates to the tissues attenuation. It is common for CT images to have lowest value of -1024 HU, and some specific HU values are presented in table 2.1.

**Table 2.1:** Table of HU units [1].

Material	HU
Air	-1000
Destiled water	0
Dense bone	2000
Metal	> 3000

## 2.5 SCAPIS

The Swedish CArdioPulmonary bioImage Study (SCAPIS), is an ongoing project in medical image analysis, with the main focus to predict cardiovascular disease [20]. The project has via six university hospitals created a large dataset of health information to be used for research. The dataset contains health information on 30154 randomly selected people, both female and male, and between the ages of 50 and 64. The dataset is very extensive, including information from questionnaires, blood samples, blood pressure tests, ECG and high resolution CT scans.

# 3

## Methods

This section presents the project resources, data preprocessing, autoencoder architecture and training, and outlier detection methods used.

### 3.1 Project resources

For this project a set of 7184 unlabeled Coronary Computed Tomography Angiography images from the SCAPIS project were provided [20]. Computational resources from the Swedish National Infrastructure for Computing (SNIC) was used for training. As the project used medical data, the SNIC cluster Bianca for sensitive data was used. This cluster included several NVIDIA A100 GPUs that were used when training the autoencoder [21]. The framework used for building and training the autoencoder was Pytorch version 1.9.0 [22].

### 3.2 Preprocessing data

The provided dataset consisted of 7184 images in DICOM format with varying pixel spacing and slice thickness. Resampling was done to achieve a symmetric voxel size of  $3 \times 3 \times 3 \text{mm}^3$  for all images, as this was the largest slice spacing in the dataset. The images were then converted to HU, with a lower cut-off at -1024 HU to ensure a uniform background value. For each image, pixel values were extracted and saved as binary files, discarding all patient information and improving network training speed. All image pixel values  $x_i$  were normalized according to equation 3.1 as,

$$\bar{x}_i = \frac{x_i - \mu}{std}, \quad (3.1)$$

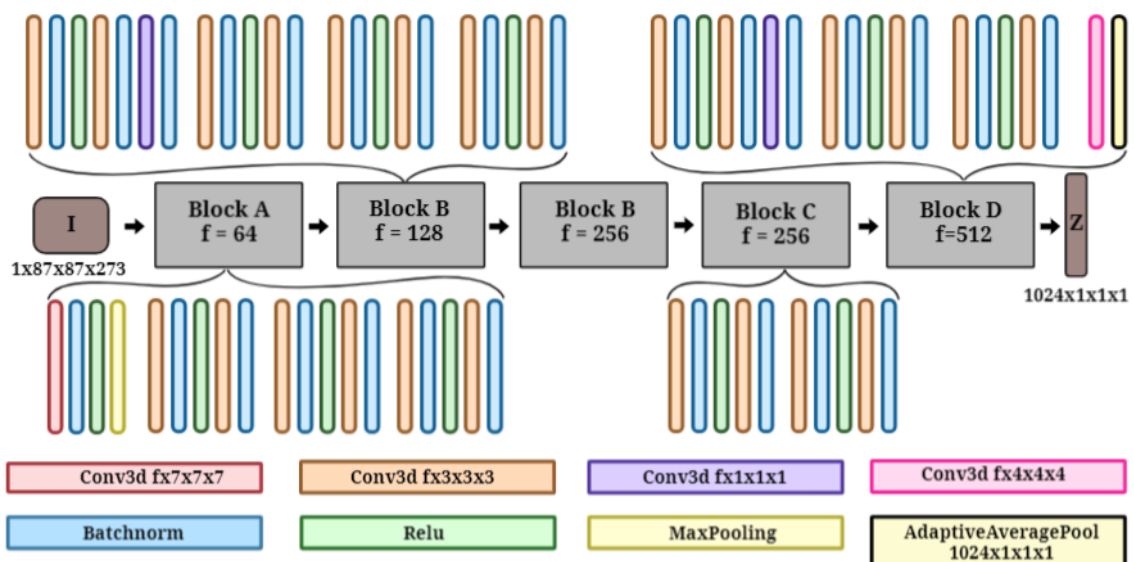
where  $\bar{x}_i$  is the normalized pixel value, and  $\mu$  and  $std$  are the mean and standard deviation calculated from all pixel values in the whole dataset.

To ensure uniform image sizes, all images were zero padded up to the size of  $87 \times 87 \times 273$  pixels. This was the largest size in the dataset and ensured minimal information loss. Zero padding was done randomly, in the sense that the original image was placed at a random position in a size  $87 \times 87 \times 273$  zero matrix. This

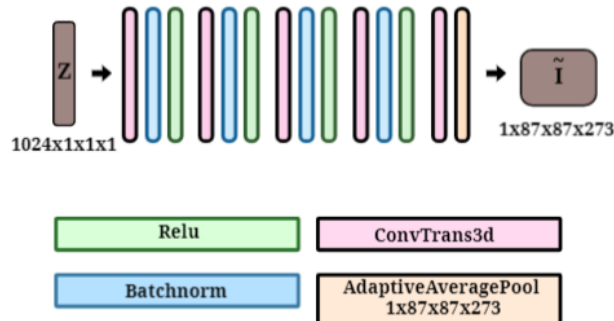
was expected to improve the networks ability to reconstruct pixel values along the borders, as compared to if all images had been symmetrically zero padded along the borders.

### 3.3 Model architecture

The encoder architecture used was based on a pretrained Med3D model presented by Chen et al. [11]. This model was based on ResNet34 and was chosen after initial testing. The pretrained model was loaded according to the available code documentation [23]. The final sequential layer was exchanged for a convolutional layer followed by an adaptive average pooling layer that outputs a latent vector of 1024 features. The decoder architecture was designed as a simple mirror of the encoder. The full AE architecture is presented in figures 3.1 and 3.2.



**Figure 3.1:** The encoder architecture divided into blocks. Each block consists of several color coded layers, see bottom of figure for layer types. The input is an original image  $I$  of size  $1 \times 87 \times 87 \times 273$ , and the output is a feature vector  $Z$  of size  $1024 \times 1 \times 1 \times 1$ . The dimensions correspond to  $[features \times width \times height \times depth]$ . The third block has the same architecture as the second block aside from a higher number of features  $f$ .



**Figure 3.2:** The decoder architecture divided into color coded layers, see bottom of figure for layer types. The input is a feature vector  $Z$  of size  $1024 \times 1 \times 1 \times 1$  and the output is a reconstructed image  $\tilde{I}$  of size  $1 \times 87 \times 87 \times 273$ . The dimensions correspond to [*features* x *width* x *height* x *depth*].

### 3.4 Model training

10 % of the images were randomly selected as a validation dataset and 10 % were randomly selected as a testing dataset. The remaining 80 % of the images were used for the training dataset. This resulted in a training dataset of 6466 images for model training, a validation dataset of 718 images for model validation between every epoch, and a test dataset of 718 images for evaluating the fully trained model. Model optimizer, cost function, and hyperparameters were chosen after initial testing of a total of two optimizers, two cost functions, and six learning rate values in different combinations. For each test the number of epochs was set so that convergence of the testing loss was visible. The final chosen model optimizer was stochastic gradient decent (SGD), and hyperparameters used for model training are presented in table 3.1.

**Table 3.1:** Hyperparameters used for training the AE.

Hyperparameter	value
Number of epochs	1600
Batch sizes	10
Learning rate	0.05
Momentum	0.9
Weight decay	$10^{-5}$

The cost function chosen for calculating the loss during training was the Mean Squared Error (MSE) Loss which is calculated as follows. For a batch of  $N$  images

with  $n$  number of pixels, the MSE Loss  $L$  is calculated as the batch norm MSE of each original and reconstructed image pair according to,

$$L = \frac{1}{N} \sum_{i=1}^N l_i. \quad (3.2)$$

Here  $l_i$  is the MSE for one original and reconstructed image pair according to,

$$l = \frac{1}{n} \sum_{j=1}^n (x_j - \tilde{x}_j)^2. \quad (3.3)$$

Here  $x_j$  is the value of pixel  $j$  in the original image and  $\tilde{x}_j$  is the value of corresponding pixel in the reconstructed image.

### 3.5 Outlier detection via reconstruction errors

With the fully trained model, both the validation and testing datasets were reconstructed and masked reconstruction errors  $MSE_m$  were calculated. The  $MSE_m$ s were calculated by first masking the original images to remove background pixels below -990. This cut-off value was chosen by visual assessment of the masked images. Then the MSE was calculated from the masked original image and the corresponding reconstructed image pixels, according to Equation 3.3. The reason for using the  $MSE_m$  instead of the MSE for identifying outliers was that the MSE showed strong correlation to the size of the original images, and it was interesting to investigate which outliers would be identified if the image size was not taken into account.

The outlier threshold was set based on the  $MSE_m$  distribution of the validation dataset, according to the outlier rule of thumb presented in Section 2.1. The mean  $\mu$  and standard deviation  $std$  of the validation dataset was calculated and the outlier threshold was set according to equation 3.4 as,

$$T = 3 \cdot std + \mu. \quad (3.4)$$

From this threshold, outliers in the testing dataset were identified.

### 3.6 Model evaluation

The common practice of evaluating unsupervised models on labeled datasets [5], was not a possibility for this project since there were no labels provided for the data. Instead the models reconstruction performance was evaluated visually, and the identified outliers were assessed. Outliers were assessed visually, via their pixel distributions, and via a feature space analysis of the latent vectors. The feature space analysis was done by extracting the latent vectors of the test dataset, then reducing the latent vector dimensions by UMAP dimension reduction. The features were then plotted and clustered using DBSCAN with parameters  $eps = 0.5$  and  $minPts = 5$ . The results of the clustering was not sensitive to the choice of parameters.

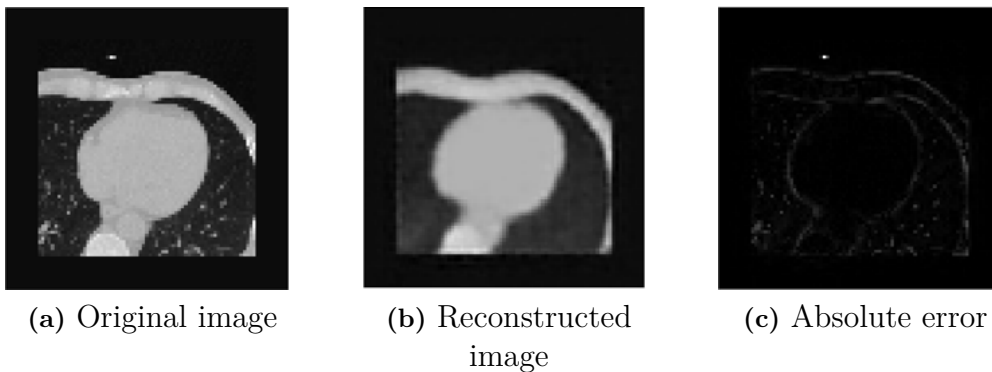
# 4

## Results

This chapter presents the reconstruction performance of the autoencoder along with the outliers identified using the masked reconstruction error. This is followed by the results of the feature space analysis.

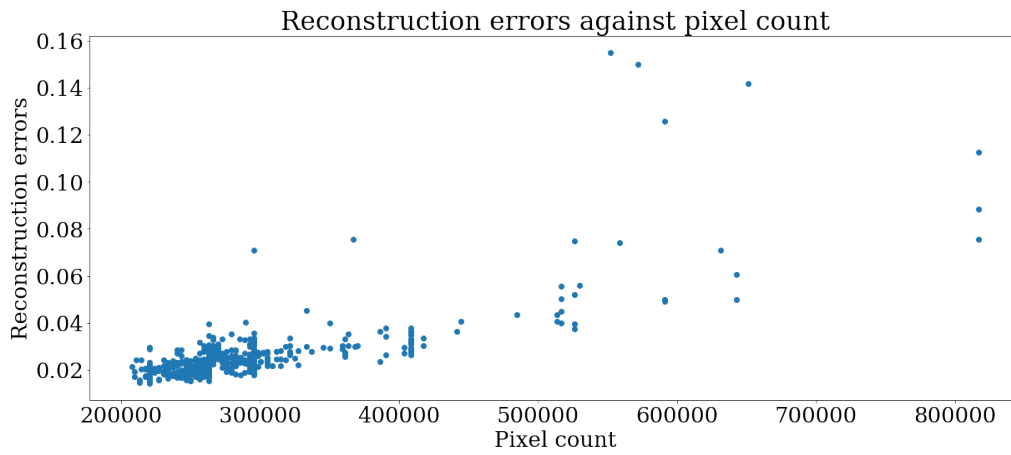
### 4.1 Reconstruction performance

A randomly selected inlier image from the testing dataset with corresponding reconstruction is presented in Figure 4.1. Visual assessment shows a strong resemblance to the original image, although without detail and with significant errors around edges in the image. A small light dot outside of the body caused by a cable or contrast tube is not reconstructed.

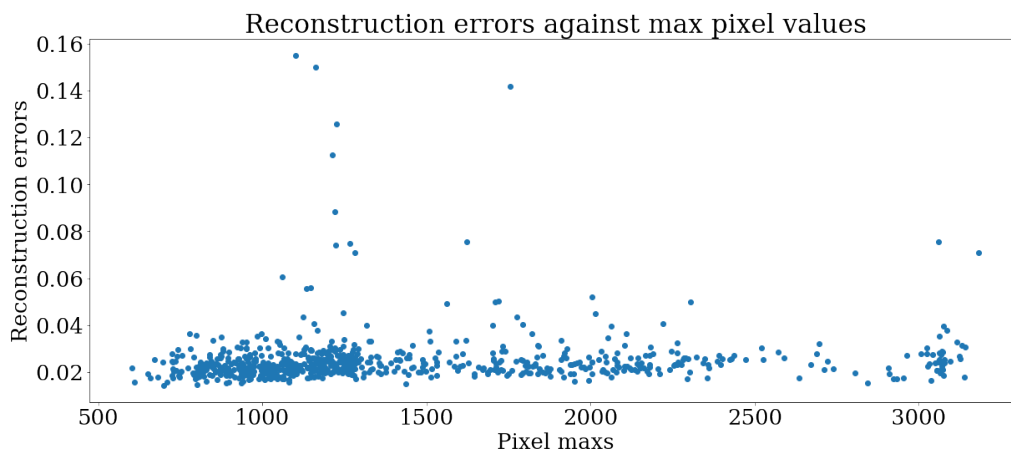


**Figure 4.1:** Axial slice of an original image from the testing dataset along with corresponding reconstruction and the absolute difference between the two.

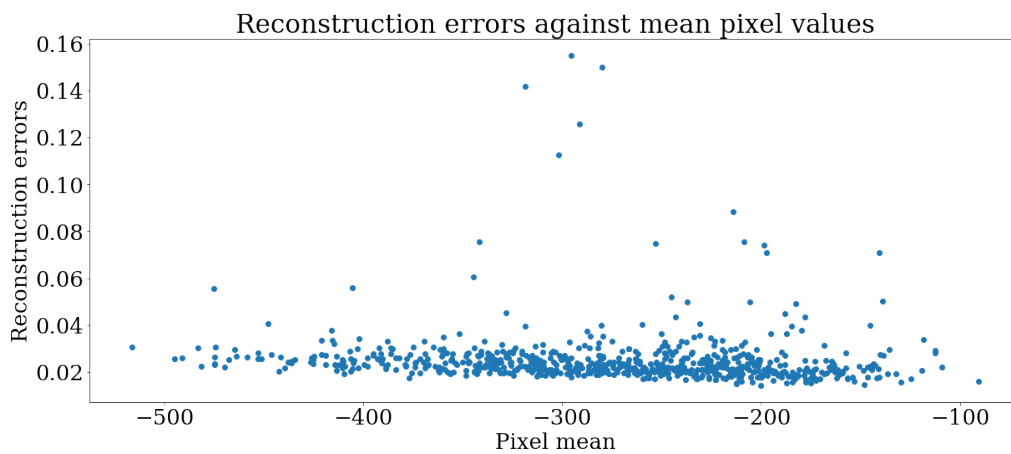
In Figure 4.2, the reconstruction errors calculated as MSE according to Equation 3.3, are plotted as a function of the number of pixels in the original images before zero padding, from here on referred to as the pixel count. There is a visible tendency for high reconstruction errors for images with a high pixel count. As can be seen in Figures 4.3 and 4.4, no such correlation between reconstruction errors and max pixel values, or mean pixel values is seen.



**Figure 4.2:** MSEs as a function of the pixel counts for the test dataset.

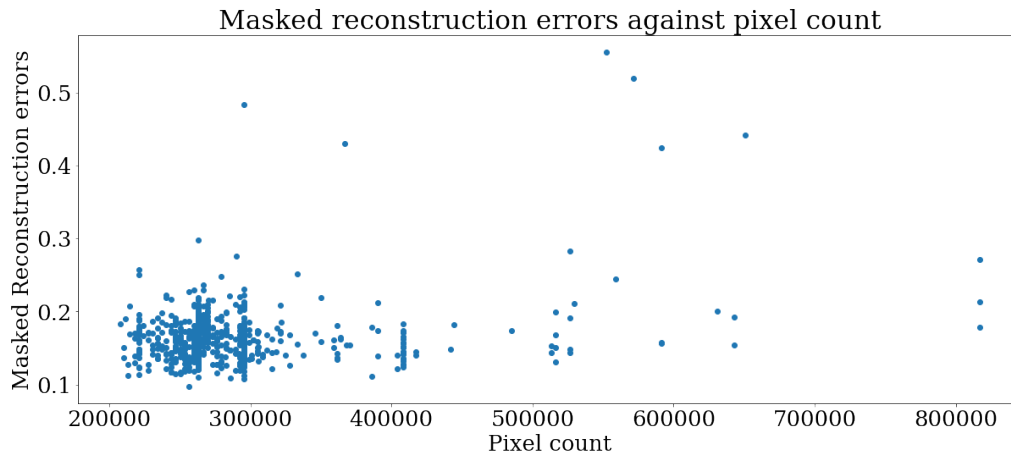


**Figure 4.3:** Reconstruction errors against max pixel values.



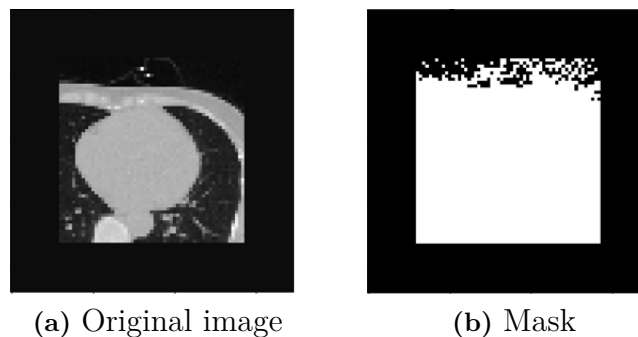
**Figure 4.4:** Reconstruction errors against mean pixel values.

Figure 4.5 shows instead the  $MSE_m$ s plotted against the total count of pixels in the original images. Here the correlation between high pixel counts and high  $MSE_m$  is clearly reduced, although not completely removed.



**Figure 4.5:**  $MSE_m$ s as a function of the pixel counts for the test dataset.

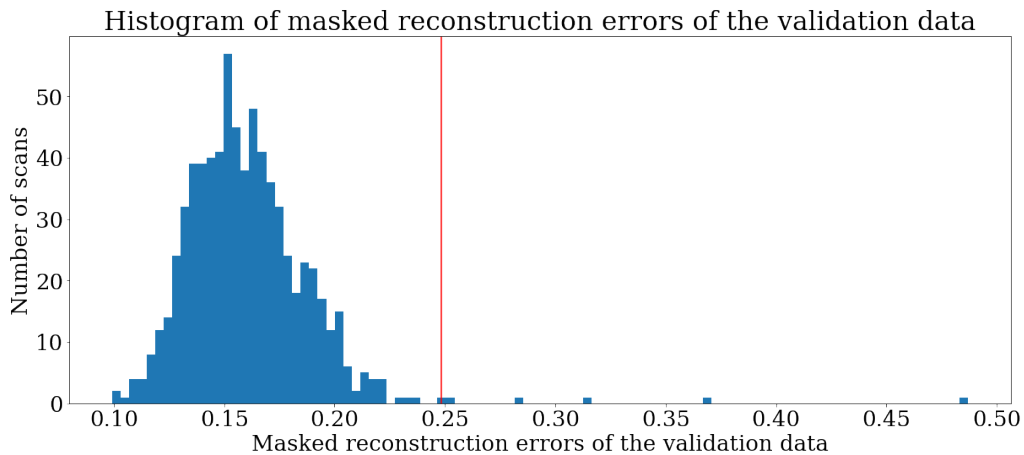
An image mask used for calculating the  $MSE_m$  is presented in Figure 4.6. The masking is mostly well fit to the edges of the original image. Some noise is visible in the part of the mask corresponding to the background pixels of the original image.



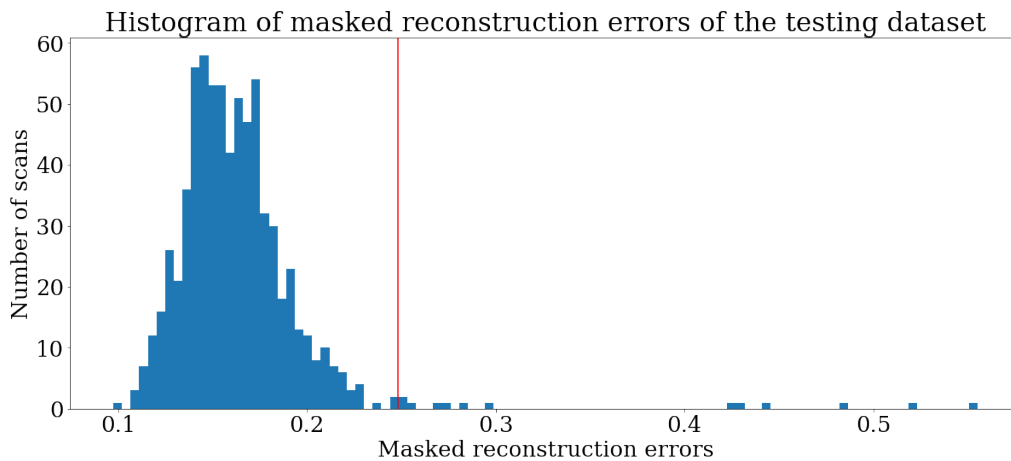
**Figure 4.6:** The masking of scan 148 from the testing dataset in a center axial slice. The white pixels in the mask show pixel indices that are included in the  $MSE_m$  calculation.

## 4.2 Detected outliers via $MSE_m$ threshold

Histograms of the  $MSE_m$ s for the validation and testing datasets are presented in Figures 4.7 and 4.8. The outlier threshold  $T$ , calculated based on the  $MSE_m$  distribution of the validation dataset is set at  $T = 0.2474$ , and is visible as a red vertical line in both figures. The distributions are very similar with a slightly lower mean for the validation dataset,  $\mu_v = 0.1605$ , than for the testing dataset,  $\mu_t = 0.1640$ . Using the outlier threshold, 14 outliers are identified in the testing dataset.



**Figure 4.7:** A histogram over the validation datasets  $MSE_m$ . The vertical line at  $MSE_m = 0.2710$  is calculated according to equation 3.4.



**Figure 4.8:** A histogram over the testing datasets  $MSE_m$ . The vertical line at  $MSE_m = 0.2710$  is the threshold for outlier identification.

Table 4.1, presents mean values calculated on the entire training dataset of the  $MSE_m$ s, pixel mean values, pixel max values, and number of pixels. The identified outliers from the testing dataset are presented in Table 4.2, along with each image's  $MSE_m$ , pixel mean value, pixel max value, and number of pixels. The outliers are named A - N with A being the outlier with the highest reconstruction error.

**Table 4.1:** Mean values of entire training dataset.

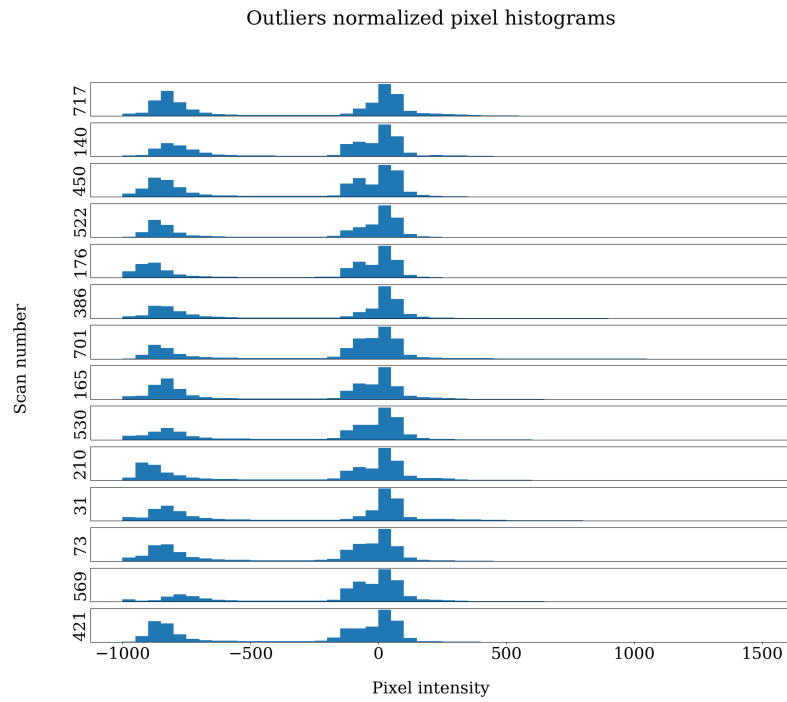
$MSE_m$	Mean pixel value	Max pixel value	Pixel count
0.1640	-276	1442	285671

Pixel histograms of the identified outliers and the 10 top identified inliers (according to  $MSE_m$ ) are presented in figures 4.9 and 4.10. The pixel histograms show similar

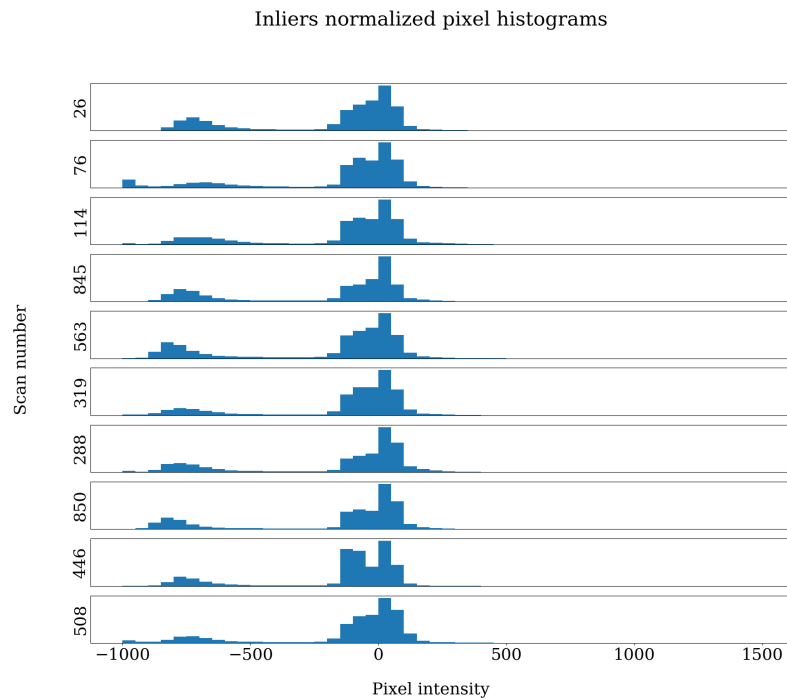
**Table 4.2:** Identified outliers  $MSE_{m^s}$ , mean pixel values and max pixel values.

Outlier	A	B	C	D	E	F	G	H	I	J	K	L	M	N
Scan	210	522	569	450	176	386	717	140	530	421	73	165	701	31
$MSE_m$	0.5555	0.5196	0.4837	0.4414	0.4301	0.4239	0.2974	0.2832	0.2758	0.2711	0.2569	0.2509	0.2507	0.2484
Mean	-296	-280	-140	-319	-342	-291	-318	-253	-260	-302	-320	-328	-202	-286
Max	1101	1162	3183	1754	3061	1225	3076	1267	1794	1162	2033	1247	1277	3063
Pixel count	552330	571824	295659	650905	367137	591318	263169	526338	289737	816998	220932	333396	220932	279414

distributions for inliers and outliers, although outliers show slightly more pixels in the lower pixel value range.

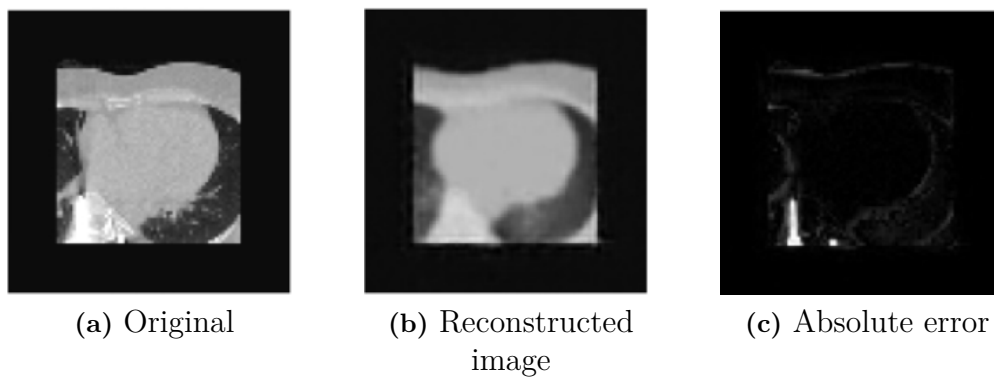


**Figure 4.9:** Histogram of the 14 identified outliers pixel values.

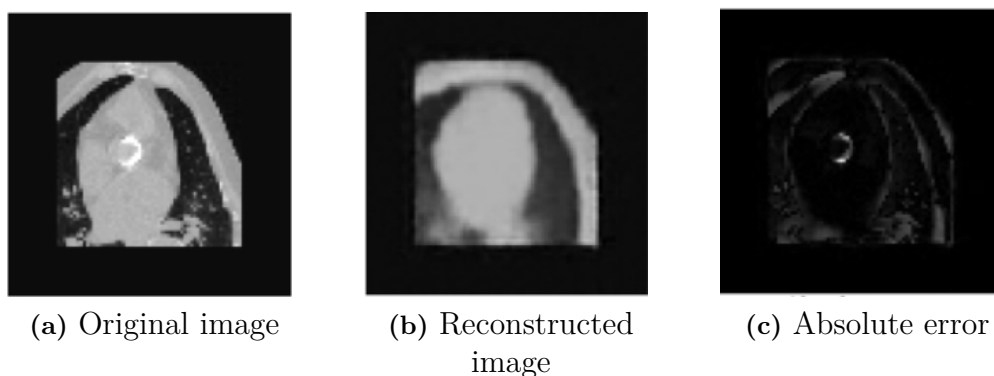


**Figure 4.10:** Histogram of top 10 inliers pixel values.

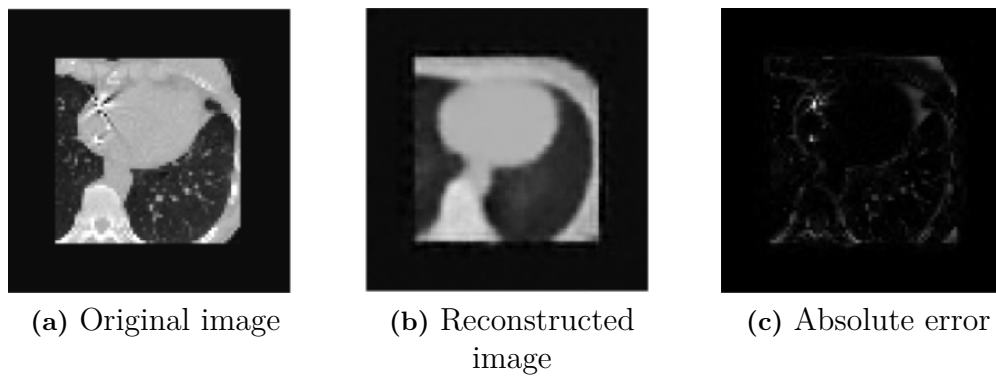
Comparing values in table 4.2 to the mean values of the training dataset presented in Table 4.1, outliers A,B,D,F,H, and J all have very high pixel counts, and belong to the 3% of training images with more than 500000 pixels. Outliers C, E, G, and N all have high max pixel values, and belong to the 5% of training images with max pixel values above 3000 HU. Visual assessment of the identified outliers showed that outliers C,E, and G contained significant oddities. An axial slice of outlier C is presented in Figure 4.11, and shows a spinal screw that has not been reconstructed. It is clear that the screw pixels contribute strongly to the high  $MSE_m$  due to the high pixel values and the large number of pixels. Outlier E is presented in Figure 4.12 and contains an artificial heart valve that has not been reconstructed. Outlier G is presented in Figure 4.13 and contains cables within the body, possibly from a pacemaker, that have not been reconstructed. More detailed orthogonal slices of outliers C, E and G, are found in Appendix A.



**Figure 4.11:** Original and reconstructed images of outlier C, and the absolute error between the two.

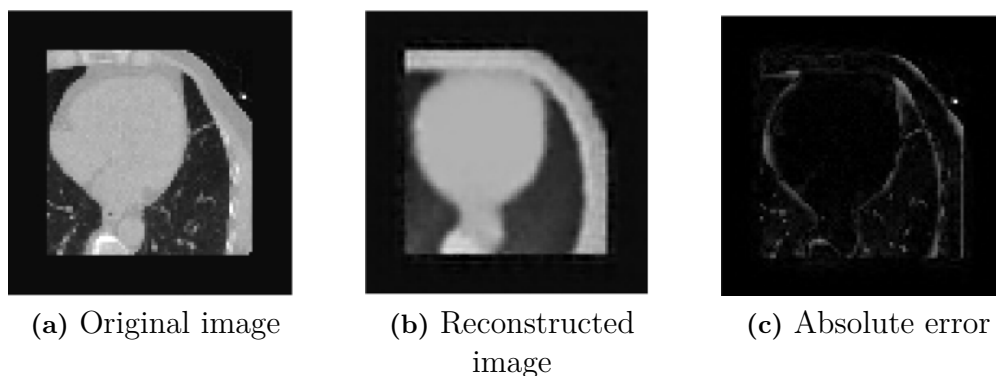


**Figure 4.12:** Original and reconstructed images of outlier E, and the absolute error between the two.

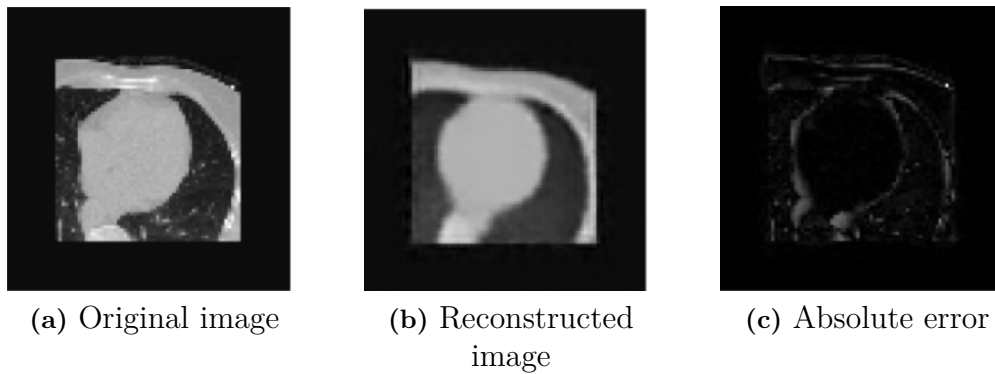


**Figure 4.13:** Original and reconstructed images of outlier G, and the absolute error between the two.

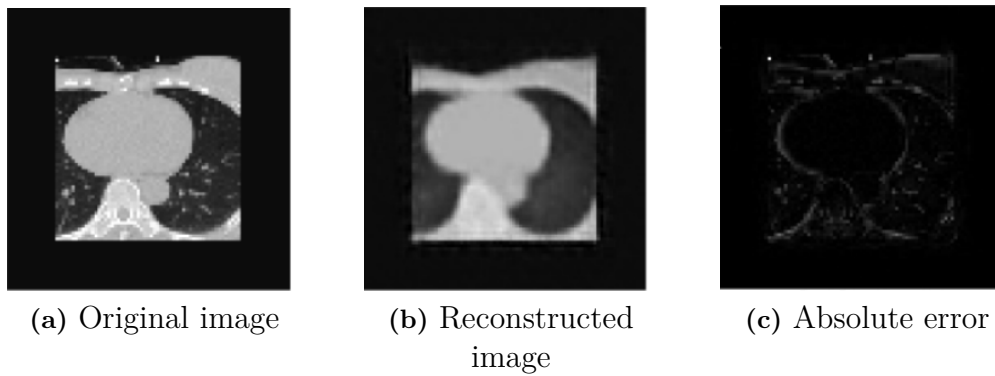
Outliers I, K and N all contain small light dots outside of the body that have not been reconstructed, see Figures 4.15 and 4.16. Noticeable is that outliers I and K have max pixel values around 2000 HU, while outlier N has a max pixel value of 3063 HU. It is possible that outliers I and K contain a contrast tubes, while outlier N contains a cable. More detailed orthogonal slices of outliers I, K and N are found in appendix A. Outliers L and M show no directly visible unusual traits.



**Figure 4.14:** Original and reconstructed images of outlier I, and the absolute error between the two.



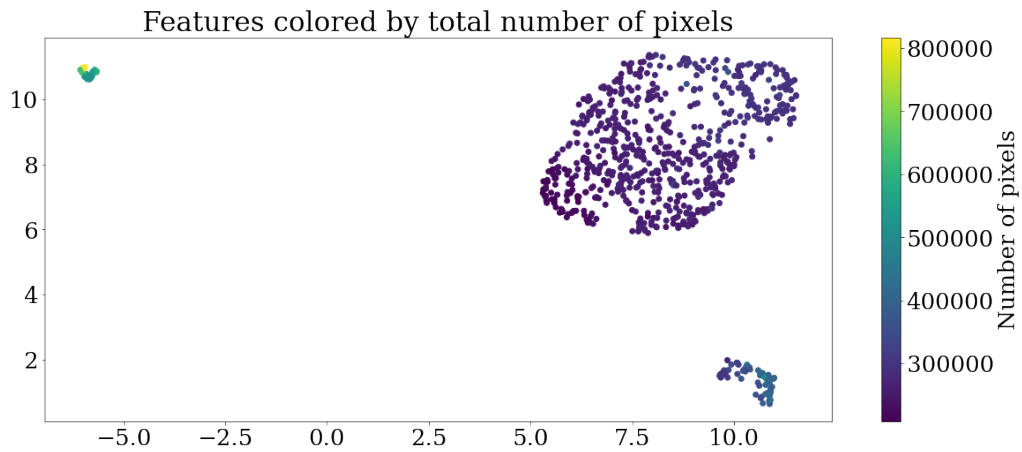
**Figure 4.15:** Original and reconstructed images of outlier K, and the absolute error between the two.



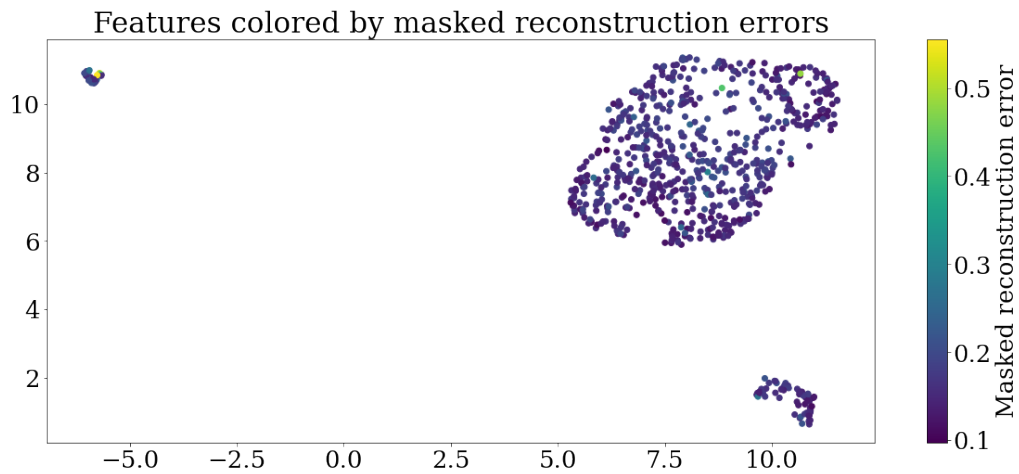
**Figure 4.16:** Original and reconstructed images of outlier N, and the absolute error between the two.

### 4.3 Feature analysis

A UMAP dimension reduction of the testing datasets feature vectors resulted in the 2D feature plots shown in Figures 4.17 and 4.18. The features are clearly clustered into three groups, and as can be seen in Figure 4.17 the groups correlate strongly to the total number of pixels in the images. The groups do not however, correlate to the  $MSE_m$  as can be seen in Figure 4.18.

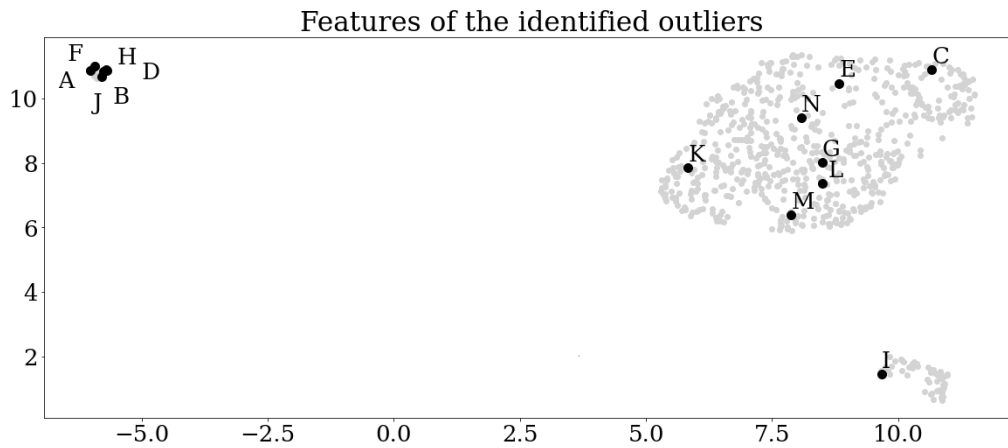


**Figure 4.17:** UMAP dimension reduction of the feature vectors, colored by pixel counts.



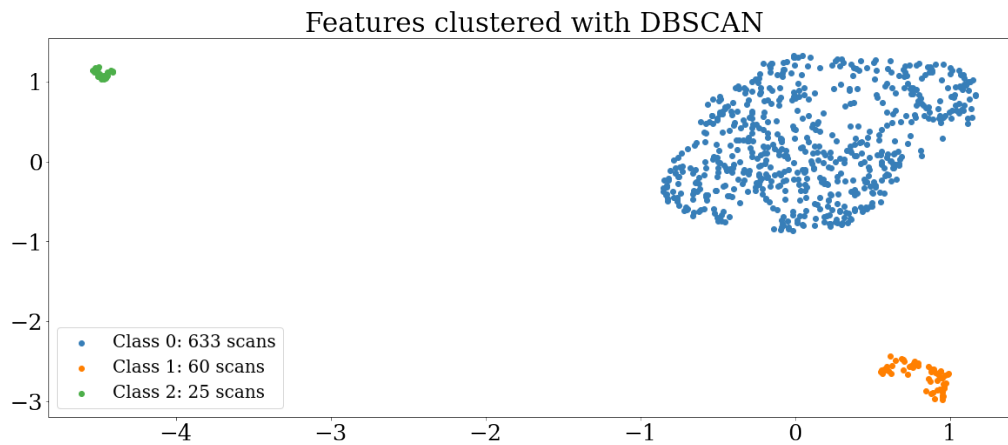
**Figure 4.18:** UMAP dimension reduction of the feature vectors, colored by  $MSE_{ms}$ .

In figure 4.19 the identified outliers are marked in the UMAP features plot. Noticeable is that outliers C, E, G, K, and N are all in the same cluster.



**Figure 4.19:** Identified outliers features marked in the UMAP feature plot.

Clustering of the UMAP feature plot using DBSCAN is presented in Figure 4.20. Normalized histograms of the clusters pixel counts and  $MSE_m$ s are presented in Figures 4.21 and 4.22. There is a clearly visible correlation between clusters and pixel counts with class 2 containing images with the highest number of pixels. The  $MSE_m$  does not have a significant correlation to the clusters. The  $MSE_m$  distributions within the clusters is somewhat similar, although class 0 and class 2 contain images with higher  $MSE_m$ s than class 1.



**Figure 4.20:** UMAP dimension reduction of the feature vectors clustered using DBSCAN.

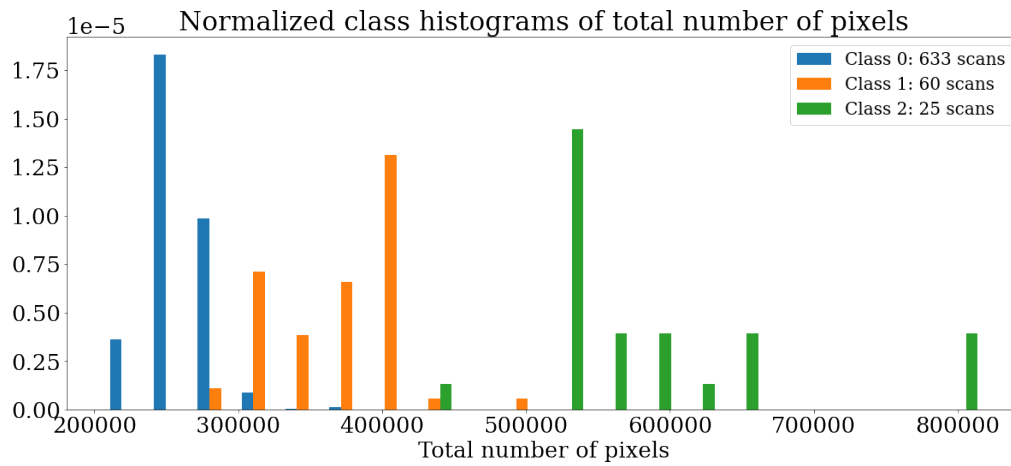


Figure 4.21: Normalized pixel count histograms for all three clusters.

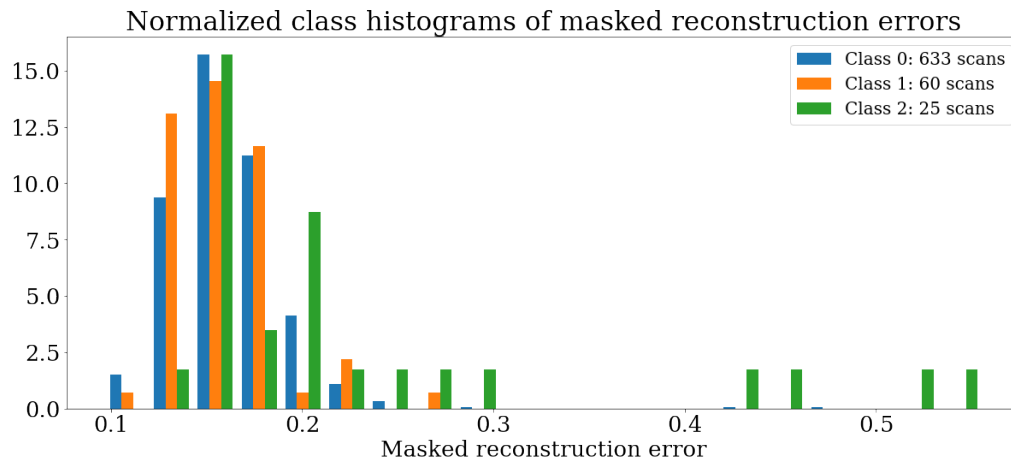


Figure 4.22: Normalized  $MSE_m$  histograms for all three clusters.

# 5

## Discussion

In this section the autoencoder’s reconstruction performance and possible improvements is discussed. This is followed by discussions about the model size bias and the use of masked reconstruction errors. Then the identified outliers are analyzed in image space and in feature space. Lastly possible future work is presented.

### 5.1 Autoencoder reconstruction improvements

The hypothesis that outlier images will be reconstructed worse than inlier images builds on the expectation that the AE can make adequate reconstructions of inlier images. Given the non-detailed reconstruction presented in Figure 4.1, there is obvious room for improvement for the reconstruction performance of the AE. Due to the limited time frame of this project, many methods of improving the network remain, both model oriented and data oriented. Model oriented methods includes trying other more advanced loss functions, or other network architectures, while data oriented methods has to do with improving the training data.

There is the possibility to use a more advanced network architecture such as the Med3D model based on ResNet50 [11], or increasing the feature vector size. Another approach would be to incorporate a discriminator to improve reconstruction resolution as is done in [6], or a cascade VAE to extract features of different detail as in [8]. This would increase the number of parameters in the model, which will require more computational power to train, and may make the model more prone to overfitting. The model has not shown tendencies toward overfitting in this project, so computational power could be the more limiting factor.

Another approach is to improve the training dataset. Since the dataset provided for this project is relatively small for a DL project, increasing the amount of training data would be beneficial. Implementing data augmentation would be a way of increasing the variations within the training data. However, the augmentations done to the data could affect what features the model learns to reconstruct. For instance augmenting the data by adding noise could result in the model classifying images with additive noise as inliers, although it may not be preferred. Therefore, augmentation should be done with knowledge of which image traits are acceptable for inlier images and which are not.

### 5.1.1 Image size bias

As seen in Figure 4.2, the models reconstruction ability is worse for images of larger sizes. One could argue that since the larger images are a minority in the training dataset, it is reasonable that they classify as outliers in the dataset, and according to the hypothesis should therefore have higher reconstruction errors. However, when the  $MSE_m$  was used this correlation between image size and reconstruction error was reduced. This points to the fact that background pixels contribute to the reconstruction error less than non-background pixels, and larger images will as a result automatically have higher reconstruction errors. Therefore, the correlation between larger images and higher reconstruction errors is instead seen as the model having a bias towards smaller images.

The  $MSE_m$  was used instead of the MSE to compensate for the size bias when detecting outliers. However, figure 4.5 shows that the correlation was not completely removed. One way to improve this could be to filter the mask to get a better fit of the mask to the foreground pixels. A better method would be to remove the models size bias, improving the reconstruction performance and removing the need for implementing a  $MSE_m$ .

The size bias is likely due to the fact that the sizes of the images in the dataset is very unbalanced, with only about 3% of the images containing more than 100 slices. One solution to this would be to acquire more images of the larger size, which is not practically feasible. Training the model on only images of the smaller size could improve the reconstruction of the smaller images, but would reduce the reconstruction performance of large images, which would be counter productive. Instead, one solution could be to segment the images into slice sections and train the model to reconstruct these sections. Then outliers could be classified based on the combined reconstruction error of its slice sections, either via a summation or as the mean. This would likely reduce the models bias towards smaller images, while maintaining much of the third dimensional information.

## 5.2 Detected outliers

Due to the dataset being unlabeled, there is no right answer to which images should be identified as outliers. Looking at the images that have been identified as outliers, there is no one common outlier trait that explains why they have a high  $MSE_m$ . Two identified possible outlier traits are high pixel counts and high max pixel values. The outlier trait of having high pixel counts is likely partly due to the remaining size bias in the model, which is likely the reason for outliers H and J being identified. However, size bias does not fully explain why A, B, D, and F have such high  $MSE_m$ .

The outlier trait of having a high pixel value is visibly verifiable as being due to metallic objects in the body. About 5 % of the images have max pixel values corresponding to metal ( $HU > 3000$ ), and this could be a justifiable outlier group of the dataset. However, there was low correlation between reconstruction error and max pixel value. One possible cause could be that several images with high max pixel

values could contain very few high value pixels, such as scan 245 in figure 4.1, which contains only a thin cable or contrast tube, similar to outliers I, K and N. Looking at the error plot of the reconstruction shows that these pixels have a high reconstruction error. However, since they are not very many they likely do not contribute that strongly to the reconstruction error of the image. If the models overall pixel reconstruction performance were to increase, it could be expected that even small features could have a significant contribution to the reconstruction error. It is possible that the model has identified outliers C,E,G and N because these contain the largest metallic objects in the dataset. However, this can not be ensured without looking through the entire dataset, which would essentially be a labeling process. Nonetheless, that the AE identifies several images with large metallic objects is a strong indicator that the model is partially successful at identifying outliers.

In this project 14 outliers were identified based on the validation sets  $MSE_m$  distribution, although as seen in Figure 4.7 there is no obvious gap between outliers and inliers. The threshold was set based on the assumption that the data should approximately fit a normal distribution, however this can be more thoroughly analyzed and there are many possible alternative methods for setting the threshold that can be investigated.

### 5.3 Feature analysis

As seen in section 4.3, the extracted features correlate highly to the sizes of the images, which complies with the discussion in section 5.1.1. This signifies that for the model, image size is the most defining feature for an image, stronger than for instance the max pixel value. This result is partly reasonable, since no image size compensation is made for the feature plot as it is for the  $MSE_m$ . The outliers identified by the  $MSE_m$ s are not distinguishable as outliers in the feature plot in figure 4.19. Outliers C, E, G, K, and N, all with visibly identifiable oddities, are all in the cluster 0, that of small images. The features of these images show no tendency of being outliers to that cluster.

The imperfect reconstructions imply that the AE has not successfully extracted all significant features, and therefore the feature map is not fully representative for the images. If the image size bias was to be reduced, and the reconstruction performance improved, the extracted feature map could be expected to be significantly different. Furthermore, the downsampling done to achieve voxel sizes of  $3 \times 3 \times 3 \text{ mm}^3$  reduced the resolution of many images, discarding much information from the images. To instead use smaller voxel sizes of for example  $1.5 \times 1.5 \times 1.5 \text{ mm}^3$  would provide the model with more information to learn from, and might result in a more informative feature space representation.

## 5.4 Future work

Future work should initially focus on improving the models reconstruction ability and removing size bias. This could be done by retraining the existing model architecture on slice sections as this would provide more, and better balanced data. To improve resolution, the model could be expanded by for instance including a discriminator network.

When the models reconstruction ability has been improved, a new feature space analysis should be done, as the model will be able to extract an improved feature space representation of the images. A feature clustering could then result in a more informative classification of inlier groups and outlier groups or points. Clustering together outliers with similar features could provide more information on different outlier causes. This could be useful to be able to identify specific types of outliers, such as outliers due to metal implants, or outliers due to noise.

# 6

## Conclusion

The autoencoder reconstruction performance is adequate, although with very low resolution and large room for improvement. To improve reconstruction resolution, a discriminator network could be added to the network and data augmentation incorporated to increase the variation in the relatively small dataset used for training. The AE shows a bias toward small images, most likely due to an unbalanced dataset. This bias is apparent when analyzing the features extracted by the encoder. For future work, training the model on slice sections instead of whole images could be a method for reducing size bias, while increasing the amount of training data. Identifying outliers via a  $MSE_m$  compensates for some of the size bias. The  $MSE_m$  distribution clearly shows that some images have significantly larger  $MSE_m$ , although there is no clear threshold between inliers and outliers. Of the identified outliers, 6 out of 14 have visibly identifiable unusual traits. The results strengthen the hypothesis of identifying outliers via an AEs reconstruction errors. Improving the model reconstruction performance and removing the models bias towards small images is expected to significantly affect the reconstruction error distribution, the extracted features, and the outlier detection performance.

# Bibliography

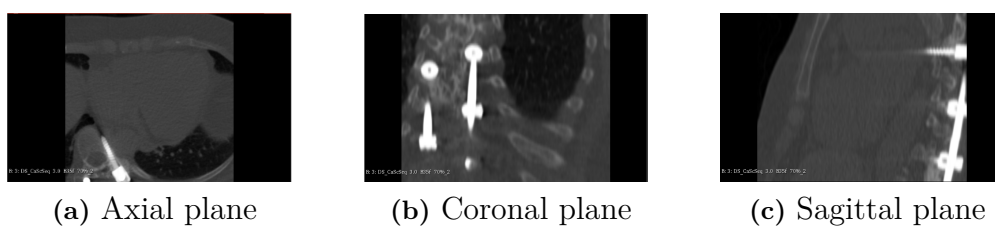
- [1] Daniel Bell and Kyle Greenway. Hounsfield unit. *Radiopaedia.org*, 7 2015. doi: 10.53347/rID-38181.
- [2] Jian Wang, Hengde Zhu, Shui Hua Wang, and Yu Dong Zhang. A review of deep learning on medical image analysis. *Mobile Networks and Applications*, 26:351–380, 2 2021. doi: 10.1007/S11036-020-01672-7.
- [3] Tharindu Fernando, Harshala Gammulle, Simon Denman, Sridha Sridharan, and Clinton Fookes. Deep learning for medical anomaly detection – a survey. *ACM Comput. Surv.*, 54(7), jul 2021. doi: 10.1145/3464423.
- [4] Maximilian E. Tschuchnig and Michael Gadermayr. Anomaly detection in medical imaging – a mini review, 2021. arXiv: 2108.11986 [eess.IV].
- [5] Henrique O. Marques, Ricardo J.G.B. Campello, Jürg Sander, and Arthur Zimek. Internal evaluation of unsupervised outlier detection. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14:47, 6 2020. doi: 10.1145/3394053.
- [6] Takahiro Nakao, Shouhei Hanaoka, Yukihiro Nomura, Masaki Murata, Tomomi Takenaga, Soichiro Miki, Takeyuki Watadani, Takeharu Yoshikawa, Naoto Hayashi, and Osamu Abe. Unsupervised deep anomaly detection in chest radiographs. *Journal of Digital Imaging*, 34:418–427, 4 2021. doi: 10.1007/S10278-020-00413-2.
- [7] Karina Zadorozhny, Patrick Thorat, Paul Elbers, and Giovanni Cinà. Out-of-distribution detection for medical applications: Guidelines for practical evaluation. 2021:1–17, 9 2021. arXiv:2109.14885 [cs.LG].
- [8] Cvad: A generic medical anomaly detector based on cascade vae. 10 2021. arXiv:2110.15811 [eess.IV].
- [9] Haibo Zhang, Wenping Guo, Shiqing Zhang, Hongsheng Lu, and Xiaoming Zhao. Unsupervised deep anomaly detection for medical images using an improved adversarial autoencoder. *Journal of Digital Imaging 2021*, pages 1–9, 1 2022. doi: 10.1007/S10278-021-00558-8.

- 
- [10] Jaime Simarro Viana, Ezequiel de la Rosa, Thijs Vande Vyvere, David Robben, Diana M. Sima, Center Tbi Participants, and Investigators. Unsupervised 3d brain anomaly detection. *Lecture Notes in Computer Science (including sub-series Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 12658 LNCS:133–142, 10 2020. doi: 10.1007/978-3-030-72084-1\_13.
- [11] Sihong Chen, Kai Ma, and Yefeng Zheng. Med3d: Transfer learning for 3d medical image analysis. 4 2019. doi: 10.48550/arxiv.1904.00625. arXiv:1904.00625 [cs.CV].
- [12] Zhen Cheng, En Zhu, Siqi Wang, Pei Zhang, and Wang Li. Unsupervised outlier detection via transformation invariant autoencoder. *IEEE Access*, 9: 43991–44002, 2021. doi: 10.1109/ACCESS.2021.3065838.
- [13] Miguel Romero, Yannet Interian, Timothy Solberg, and Gilmer Valdes. Targeted transfer learning to improve performance in small medical physics datasets. *Medical Physics*, 47:6246–6256, 12 2020. doi: 10.1002/MP.14507.
- [14] Hongzhi Wang, Mohamed Jaward Bah, and Mohamed Hammad. Progress in outlier detection techniques: A survey. *IEEE Access*, 7:107964–108000, 2019. doi: 10.1109/ACCESS.2019.2932769.
- [15] Martin Ester, Hans-Peter Kriegel, Jiirg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. 1996. URL [www.aaai.org](http://www.aaai.org).
- [16] Leland McInnes, John Healy, and James Melville. Umap: Uniform manifold approximation and projection for dimension reduction. 2 2018. doi: 10.48550/arxiv.1802.03426. arXiv:1802.03426 [stat.ML].
- [17] neuronnätverk. URL <http://www.ne.se/uppslagsverk/encyklopedi/lång/neuronnätverk>. Accessed: 2022-05-18.
- [18] Keiron O’Shea and Ryan Nash. An introduction to convolutional neural networks. 11 2015. doi: 10.48550/arxiv.1511.08458. arXiv:1511.08458 [cs.NE].
- [19] datortomografi. URL <http://www.ne.se/uppslagsverk/encyklopedi/lång/datortomografi>. Accessed: 2022-05-08.
- [20] About – the scapis study. URL <https://www.scapis.org/about/>. Accessed: 1-2-2022.
- [21] Snic-sens - swedish national infrastructure for computing. URL <https://www.snic.se/allocations/snic-sens/>. Accessed: 5-2-2022.
- [22] Learn the basics — pytorch tutorials 1.11.0+cu102 documentation. URL <https://pytorch.org/tutorials/beginner/basics/intro.html>. Accessed: 2022-06-02.

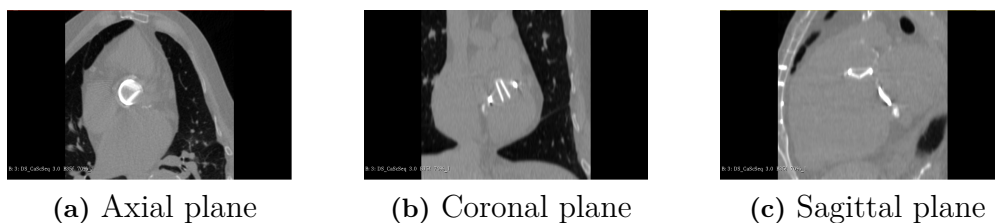
- [23] Github - tencent/medicalnet: Many studies have shown that the performance on deep learning is significantly affected by volume of training data. the medical-net project provides a series of 3d-resnet pre-trained models and relative code. URL <https://github.com/Tencent/MedicalNet>. Accessed: 2022-06-02.

# A

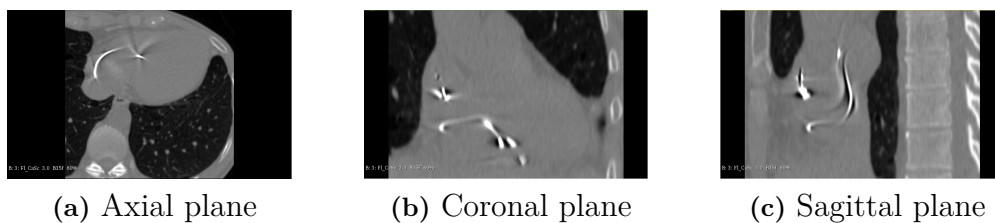
## Appendix A



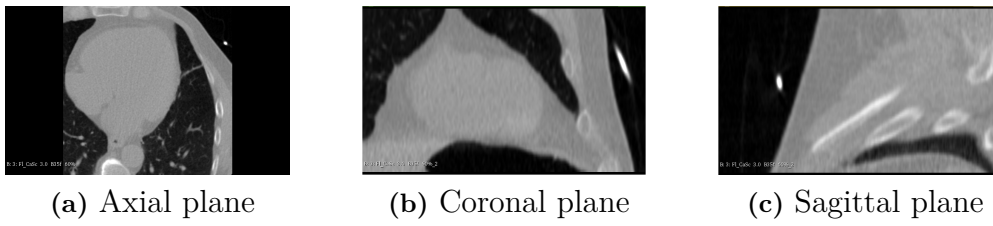
**Figure A.1:** Slices in three orthogonal planes of outlier C.



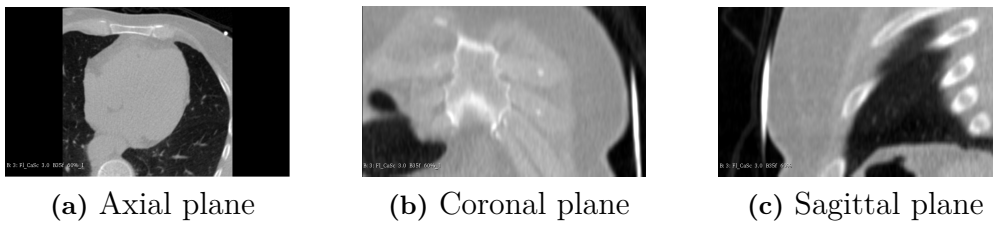
**Figure A.2:** Slices in three orthogonal planes of outlier E.



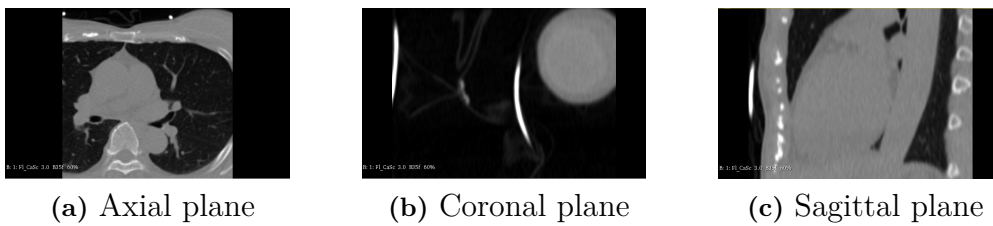
**Figure A.3:** Slices in three orthogonal planes of outlier G.



**Figure A.4:** Slices in three orthogonal planes of outlier I.



**Figure A.5:** Slices in three orthogonal planes of outlier K.



**Figure A.6:** Slices in three orthogonal planes of outlier N.

DEPARTMENT OF ELECTRICAL ENGINEERING  
CHALMERS UNIVERSITY OF TECHNOLOGY  
Gothenburg, Sweden  
[www.chalmers.se](http://www.chalmers.se)



**CHALMERS**  
UNIVERSITY OF TECHNOLOGY