university of groningen / faculty of science and engineering

Automatic teeth thresholding in cone beam CT with convolutional neural networks and tooth segmentation with the watershed transform

Bachelor's Project Thesis

Ruben Cöp, s2703122, r.cop@student.rug.nl
Supervisors: Dr M. A. Wiering & Dr W. J. van der Meer

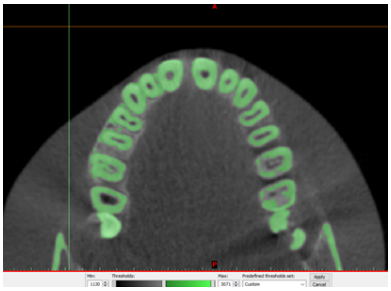
Abstract: The segmentation of a tooth from a Cone Beam Computed Tomography (CBCT) scan is a time consuming process. This study focuses on the automatic segmentation of teeth from CBCT data. It proposes a method for automatic thresholding of teeth in a CT scan using convolutional neural networks, as well as a method for the automatic segmentation of a tooth using the watershed transform. The study shows that the models used in this study suffer from underfitting and are therefore not suitable for automatic thresholding in CBCT scans. This study also finds that the watershed transform is able to segment a tooth slice by slice with reasonable accuracy.

1 Introduction

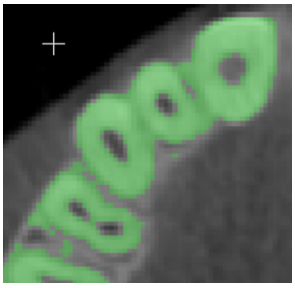
The interest of using machine learning techniques in medicine has increased over the last few years. Cone Beam Computed Tomography (CBCT) is increasingly being used for diagnosis and treatment planning in orthodontics and oral surgery. For elaborate treatment plans, the teeth have to be segmented from the CBCT datasets as individual 3D models in special software (Mimics, Materialise, Belgium).

At the moment the process of segmenting is done manually. The segmentation process includes setting a threshold value (given in Hounsfield Units, HU) for all the hard tissues of the teeth (enamel, dentin and cementum), see Figure 1.1a, and manually segmenting a specific tooth from the structures included by the threshold value by removing parts that do not contain data related to the teeth. When a 3D model of a single tooth is needed, a CBCT scan has to be analyzed, and contact areas with adjacent teeth are removed, see Figure 1.1b. This arduous labor can easily take up to 2 hours of work for one dataset. Finally, the segmented tooth can be reconstructed using a region growing tool, after which a 3D model of the tooth is available. Automating this process by applying machine learning and image processing techniques may drastically decrease the time needed to construct a 3D model of a tooth.

To segment a structure from a CBCT scan, Statistical Shape Models (SSM) often perform well. SSMs have been used for 2D data since the '90s, but SSMs for 3D structures have only been used for the last decade (Heimann and Meinzer, 2009). A Statistical Shape Model uses a template 3D model for the structure that needs to be segmented. However, since teeth widely differ in shape, such a template 3D model will probably not work for segmenting teeth since it will be hard to make a uniform SSM that represents all teeth. Successful tooth segmentation for the upper jaw has been accomplished by first segmenting the maxilla using an SSM in order to find out where the teeth would approximately be. Subsequently, 15 planes were fitted between the teeth of the upper jaw (Lamecker, Kainmueller, Zachow, et al., 2012). This study achieved reasonable results in segmenting teeth. Moreover, this technique was also able to classify areas where teeth were missing. However, this study only focused on segmenting teeth from the upper jaw. This technique also assumes that the teeth are more or less in a perfect arch. This technique will not perform well on CBCT datasets of patients with an orthodontic anomaly. Dispositioned teeth are common in the



(a) A threshold has been set to capture all teeth in this axial slice



(b) An example of teeth touching each other in an axial slice

Figure 1.1: Examples of an axial slice for which a threshold has been set to capture teeth. This was done with the 3D medical image processing software of Mimics 10.01

field of orthodontics, which makes the approach described above not very useful in orthodontics. Moreover, this approach requires a CBCT scan of the complete upper jaw. A scan that only contains part of the jaw, or a scan that is of poor quality, will be useless for this approach.

Convolutional Neural Networks (CNN) have proven to perform well on various image classification tasks. For example tasks on character classification (Ciresan, Meier, Gambardella, and Schmidhuber, 2011b), (Ciresan, Meier, Masci, Maria Gambardella, and Schmidhuber, 2011a). A CNN is a type of artificial neural network, where input features are processed through a multilayered network, defined by a network of weights and biases, to produce a non-linear function mapping the input to the output. CNNs operate on input with regular structure, (Yong, Tan, McLaughlin, Chee, and Liew, 2017). In recent years there is an upcoming interest in using image classification with CNNs on 3D volumes like CBCT scans. These networks can, for example, be used in Computer Aided Diagnosis (CAD) systems. A recent study by Alakwaa et al. focused on the detection of lung cancer nodules in CBCT scans of lungs (Alakwaa, Nassef, and Badr, 2017). In the latter case, the CNN learns features associated with lung cancer nodules. To learn specific structures in 3D data, like in the latter example, one may want to use a 3D-CNN instead of a 2D-CNN. Using a 3D-CNN yields better results since it takes into account the full 3D context of a structure. A 3D-CNN was used in the lung cancer detection study as well in a study by Fechter et al. which focused on the automatic segmentation of the Esophagus from a CT scan (Fechter, Adebahr, Baltas, Ben Ayed, Desrosiers, and Dolz, 2017). In this study a Fully-3D-CNN was used, to generate label probabilities for all labels at once. This technique enabled the construction of a probability map of the location of the esophagus. This probability map then guided a Random Walk algorithm to segment the esophagus from other tissue.

There are, however, a few problems when applying the approach of Fechter et al on teeth. First of all, the esophagus has more or less the same shape over all axial slices. Teeth, on the other hand, differ widely in shape per axial slice. Hence, a lot more data is needed to train a network to learn the features of teeth. Moreover, the pre-processing of the data in positive (i.e. data that contains a tooth) and negative examples (i.e. data that doesn't contain a tooth) is more complex since teeth grow in a crescent arrangement in two opposing arches. Since the network needs to learn the features of a separate tooth, one would need to segment all teeth by hand, which is arduous labor. Another problem is that, if the CNN were able to create a probability map of a tooth, the network would create multiple probability blobs when applied to a CBCT scan, because there are multiple teeth in a CBCT scan. Since teeth are in close contact to each other, the probability maps of the teeth would probably overlap, which would leave us with a probability map of teeth that are still touching each other. This outcome would still require some sort of segmentation.

One widely used segmentation technique which is used frequently in image processing is the watershed transform. The idea behind the watershed transform is that an image can be seen as a landscape with hills and valleys (w.r.t pixel intensity).

The watershed transform then rises virtual water levels from local minima in an image. At the points where different water basins touch each other, a line is drawn. This line then segments the two watersheds from each other. This technique has proven to perform well on segmenting biological structures from images. The watershed transform has for example been used to segment clustered nuclei from bone marrow in fluorscence microscopy-based analytical cytology (Malpica, de Solorzano, Vaquero, Santos, Vallcorba, García-Sagredo, and Del Pozo, 1997). An accuracy of 90% was achieved in correctly segmenting the nuclei clusters from the surrounding environment. Such techniques may dramatically decrease medical analysis time. A downside of the classical watershed, as described in (Gonzalez and Woods, 2002), is that it often oversamples a structure that needs segmentation. This is due to the fact that it rises water levels from every local minimum. To prevent this, a marker based watershed transform can be used. This technique has for example been used in a study on automatically segmenting nuclei from cancer cells, in order to perform a time-lapse analysis on the progression of cancer (Yang, Li, and Zhou, 2006). Such a marker is a connected component of an object, meaning that each object has its own marker area which belongs to that object specifically. In marker-based watershed, the image is flooded from the marker areas instead from the local minima in the image.

While object detection with CNNs, in the sense of detecting where teeth in a CBCT dataset occur, may not be feasible, CNNs can still come in handy for predicting the threshold value set by a clinician. Since an object detection approach would be more of a classification type of problem (pixel wise classification, namely tooth or no tooth), threshold prediction would be a regression problem. To our knowledge, regression with CNNs has not yet been used in the field of orthodontics. But it has been used in other medical fields, e.g. in a study on the automatic segmentation on left ventricular MR cine images. In this study a CNN was trained on 2D images to determine the radial distance of the left ventricular myocardium from the left ventricular centerpoint (Tan, Liew, Lim, and McLaughlin, 2017). This distance is on a continuous scale, which requires a regression approach. Another study focused on automatically segmenting a blood vessel (Yong et al., 2017). Segmenting blood vessels from intravascular optical coherence therapy images is needed for the assessment of coronary artery diseases. In this study a CNN was trained to learn the radial distances of 100 points drawn on the blood vessels wall. This method was able to get a good performance on segmentation. The aforementioned study showed that convolutional neural networks are able to learn regression on visual data. Therefore the regression approach was also used in our study.

The study described in this paper proposes a technique to fit a Convolutional Neural Network to learn the right HU threshold value to capture teeth in a CT scan. After setting a threshold value, a tooth is segmented slice by slice to create a 3D model. This slice by slice segmentation process is done by using a watershed transform on an axial slice of teeth. The aim of this study is to determine whether CNNs are able to detect a decent threshold value for teeth in a CBCT dataset, which has maximum mean deviation of 20 HU from the gold standard (see Section 2.1 for an explanation on Hounsfield Units). Also, this study will establish whether the watershed transform is able to segment a tooth from a CBCT-scan by measuring how well it performs in segmenting a tooth from axial CBCT slices.

2 Methods

2.1 Data

The data used in this study consists of CBCT-scans of 60 patients. These scans have been made with different CBCT scanners on different days. Unlike CT machines, CBCT machines are non-calibrated systems and do therefore often show variance in their HU values not only between different machines, but also in datasets generated by the same machine. The image size of the axial slices differs per CBCT-scan, as well as the number of slices per scan. The slice resolution is either 512×512 pixels, or 1024×1024 pixels. The number of slices per scan is between 254 slices and 1001 slices. This difference in amount of slices is mainly due to the resolution of the scans w.r.t. the amount of slices per mm. This amount varies from 2 up to 8 slices per mm. Another reason that the amount of slices varied in this study is that a larger field-of-view

(FOV) was chosen for one scan, while other scans used a smaller FOV. The pixel values of a CBCT scan are expressed in Hounsfield Units (HU), which is a quantative scale for expressing radiodensity. On the HU scale the radiodensity of air is defined as -1000 HU, while the radiodensity of water is 0 HU. Tooth enamel has a value around 1550 HU for adults and 2050 HU for children, according to the predefined thresholds by the software used for thresholding (Materialise Mimics, Belgium). These values already differ 500 HU, and thus age is of significant influence on the Hounsfield Unit of the enamel. A tooth does not only consist of enamel, but also of dentin, pulp and cementum, which have lower HU values. The threshold values for the scans used in this study ranged from 400 HU up to 1753 HU.

In the scans a threshold value was set manually. The anonymized CBCT datasets were supplied by the Department of Orthodontics of the University Medical Center Groningen (UMCG).

2.2 Convolutional neural networks

2.2.1 Pre-processing of data

First of all, the 2D-CNN is trained on 2D data and the 3D-CNN is trained on 3D data volumes. The networks are trained on data from axial slices that contain teeth. In case of the 2D-CNN the separate slices are chopped into sub-images of 100×100. CBCT scans often contain a lot of air (e.g. the areas around the head and the oral cavity). Some scans also capture large parts of the skull. Sub-images from these regions are not put into the final dataset as these sub-images do not consist of data representative for the threshold value, so they are excluded from the final data set. This is done by checking what percentage of the pixels in the sub-image exceeds the threshold for that specific scan. If this is above 20%, the sub image is included in the final data set, otherwise it is excluded. The threshold of 20% was determined by performing a parameter sweep. Eventually, for a random sub sample (i.e. 20 images) from each patient scan it was checked which images were included in the final data set. Among the included images were some images that only contained jaw bone, or random noise caused by reflections of metal objects (e.g. a brace). These images were hard to exclude automatically. Excluding them by hand would be arduous labor. Since they made up only a small part of the data, it was decided to leave these images inside the dataset. Data for the 3D-CNN is cut in chunks of $100 \times 100 \times 10$, in which 10 is the depth dimension. For the latter, 10 subsequent axial slices are taken. Again, redundant chunks are filtered out by a strategy similar to that applied on 2D data. However, now the 20% of the pixels of the whole chunk must exceed the CBCT scan specific threshold. Each sub image (for the 2D data, as well as the axial sub-images of the 3D volumes) is resized from a 100×100 image to a 50×50 image. This is done to decrease the volume of the final dataset. Besides that, the resolution of the reshaped images is still high enough that it does not suffer from a problematic loss of data.

The pre-processing pipleline as described above resulted in a total dataset size of 182,265 2D images for the 2D-CNN and 19,893 3D volumes for the 3D-CNN. Before feeding the data to the networks, the dataset is split up into a training set and a test set. The data is first randomly mixed per patient, to prevent the system training on data from a specific area of a scan only. Then 80% of the data of a patient is added to the training data, and 20% is added to the test set. This approach makes sure that the network trains on data from every patient, and that there is no overlap between the train data and test data. Besides that, the model does not train on data from only one region of a CBCT scan (e.g. the upper jaw), and test on another region, but the data is mixed such that the model trains and tests on data from every region in a CBCT scan.

2.2.2 2D-CNN and 3D-CNN layout

Since the study described in this paper is of an exploratory type, there was no general architecture of a CNN from another study from which a model could be used as a starting point to solve the problem of threshold learning on visual data. Therefore, a CNN had to be constructed from scratch to learn a threshold value with regression. Often, CNNs are used for classification problems. For example in a study in Imagenet classification, where images had to be assigned to one of 1000 possible classes (Krizhevsky, Sutskever, and Hinton, 2012). However, for threshold finding a different approach is needed. The threshold is on a continu-

ous scale, whereas a classification problem is on a binary scale; 0 if the example does not belong to the class and 1 if the example belongs to the class. The big difference between classification problems and threshold prediction lies in the use of the loss-function. In a classification problem a softmax with cross entropy loss function between the predicted output and the correct output is often used. This will not work for a regression problem. Here a loss function that computes the Mean Squared Error (MSE) between the predicted value and the correct value, suits better. The function for the MSE is:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(\hat{X}_i - X_i)^2 \tag{2.1}$$

where $\hat{X}_i$ denotes the values of n predictions and X_i represents n the number of correct values. The lower the loss, the better the model is fitted to the train data. An Adam Optimizer is used to minimize the MSE (Kingma and Ba, 2014). A commonly used activation function is the ReLU activation function. However, since the ReLU is a discontinuous function, it sometimes returned very large activation output. As a result, the loss could not be computed and returned NaN. Using the tanh activation solved this problem as this is a continuous function. Where tanh activation was used can be seen in Figure 2.1. The fully connected layer consists of 512 nodes.

The general architecture for both the 2D-CNN and 3D-CNN is found in Figure 2.1. The main difference between the networks is that the 2D network uses a 2D convolution in the first convolutional layer, whereas the 3D network uses a 3D convolution in all convolutional layers. The model in Figure 2.1 resulted from tests with simpler models. Testing started with a simple network which consisted of only one convolutional layer. This appeared to be a too simple model to learn regression on the dataset. The model was made more complex to reduce the size of the input image, and to summarize the data from the image in fewer nodes.

The target value that the network has to learn is the threshold value for a specific patient. However, to train the network with a decent learning rate, the threshold values have to be normalized between 0 and 1. Otherwise, the learning rate would be too high, since the threshold values are between 400 HU and 1753 HU, which are really large values. These large values will cause the model to converge too

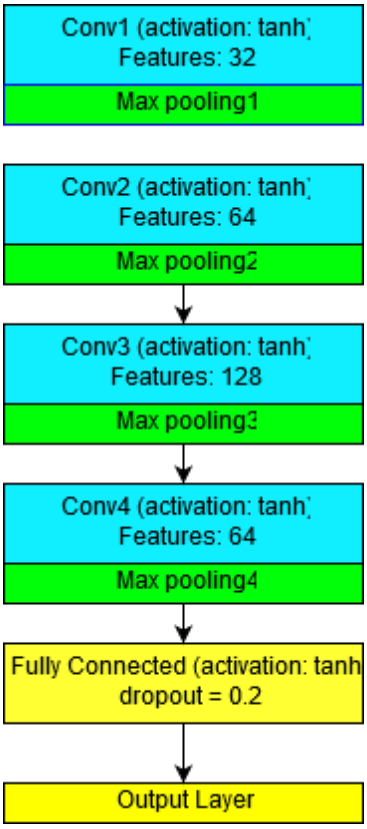


Figure 2.1: Overall architecture of both the 2D- and 3D-networks.

fast. Normalization of the threshold values is done by the following formula:

$$z_i = \frac{x_i - min(x)}{max(x) - min(x)} \tag{2.2}$$

in which z_i is the normalized i-th threshold value, x_i is the original i-th threshold value, and $min(x)$ and $max(x)$ are the minimum and maximum possible threshold values respectively. After training, the right predicted threshold value for a validation example can be obtained by using the inverse of this formula on the predicted output value of the network.

For all convolutional layers a kernel size of 5×5 was used. Larger kernel sizes (9, 15) have also been tried, but this did not have a significant influence on the final performance of the network.

2.2.3 Validation of the model

The model performance was tested on the data in the test set. The model had to predict the threshold value for each patient. In the pre-processing step it was made sure that 20% of the data from every patient was included in the test set. The final threshold prediction for a patient was the mean prediction over all test set examples of a patient.

2.2.4 Program and training

The network was programmed in Python using Tensorflow. Training of the networks was done on Peregrine, the High Performance Computing Cluster (HPC) of the Center of Information Technology of the University of Groningen. Here the networks were trained on a node containing 2 Intel Xeon E5 2680v3 CPUs, 128GB of RAM, 2 Nvidia K-40 GPU cards with 12GB of memory each. On this node only half of the RAM was requested, as well as only one GPU. The reason for this is that when using only half of the capacity, someone else could use the other half of the node, which decreased the waiting time in the SLURM queue of the Peregrine cluster. In the final configuration it took around one hour to train the network. In this situation, the network was trained for 50 epochs, which was enough for convergance of the loss function.

2.3 Segmentation

Once a threshold value is determined for a specific data set, a tooth defined by the user is segmented from the CBCT scan. This is done by using the Watershed algorithm (Vincent and Soille, 1991).

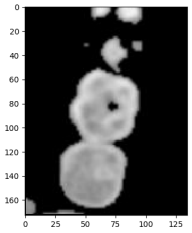
2.3.1 Pre-processing

To segment a tooth, a scan was first pre-processed. A HU threshold was applied to the scan, removing all tissue below the threshold. Here, the HU threshold set by the expert for a specific scan was used, not the threshold predicted by the system. Then an interactive program asked a user to cut out a section in an axial slice, making sure the tooth that had to be segmented was in the middle of the image that was cut out. Examples of such images can be found in Figure 2.2a and 2.3a. The user also had to specify which range of axial slices included the tooth. From this range a 3D object of slices was created that included the tooth.

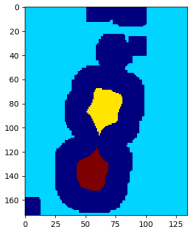
2.3.2 Watershed transform

The watershed transform is an extremely useful tool for segmenting grayscale images. The watershed transform is inspired by the field of topography. The idea is that grayscale images can be interpreted as topographic reliefs. By letting 'water drops' fall on on the image, different catchment basins (i.e. local minima) will fill, and eventually the water levels will flow into each other. At this point, a line is drawn between the water basins to separate them from each other. This line is interpreted as the segmentation line. More details on the watershed transform can be found in the paper of Vincand and Soille (Vincent and Soille, 1991).

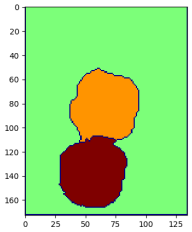
Since rising a water level from every local minimum in the image would yield a lot of segmented areas, in this study a label-based watershed technique has been used. Here it is specified which area in an image is certainly part of a tooth, which areas are certainly not a part of a tooth, and which areas are uncertain. The areas that are certainly part of a tooth are regarded as the local minima in the image, which are the middle areas of a tooth. For the pixels labeled as uncertain it needs to be determined to which watershed it belongs. Figure 2.2b and figure 2.3a are examples of labeled areas.



(a) Original thresholded axial image of molars.

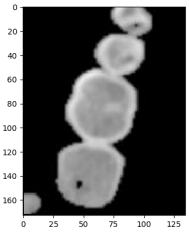


(b) Labeled image. Dark blue is labeled as uncertain. Yellow and orange areas represent local minima.

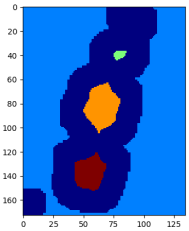


(c) Segmented result after watershed transform.

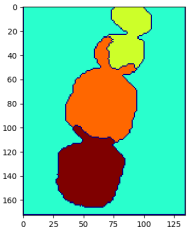
Figure 2.2: Examples of a good tooth segmentation



(a) Original thresholded axial image of molars.



(b) Labeled image. Dark blue is labeled as uncertain. Green, orange and red areas represent local minima.



(c) Segmented result after watershed transform.

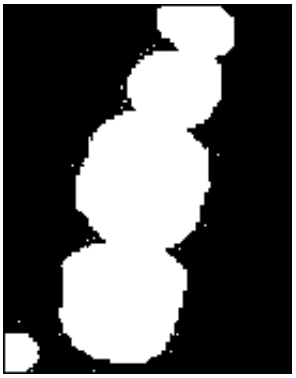
Figure 2.3: Examples of a non-optimal tooth segmentation

Here the uncertain area is marked in blue and the local minima are visible in the center of the teeth.

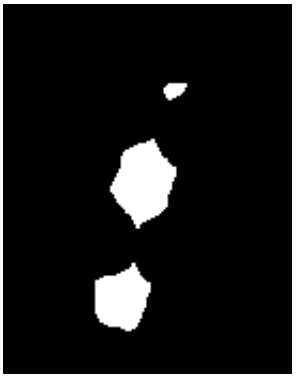
The images that are processed with watershed consist of a part of an axial slice. It is assumed that the tooth that needs to be segmented is approximately in the middle of the image. Since the axial slices of a CBCT scan are 3-channel images, they are first converted to a 1-channel gray scale image. After that, the image is pre-processed with flood filling, to ensure that possible holes in the middle of teeth, for example in the middle tooth of Figure 2.2a, are filled. After that, the image is converted to a binary image, and an erosion algorithm is applied to the image to find the centers of the tooth. The erosion algorithm works with a distance transform, which computes for every non-zero pixel the shortest distance to a 0 pixel. Next, a pixel is set to zero if it is closer than $x \times max(d)$ from a zero pixel, in which d is an array containing all distances for non-zero pixels. This calculation is applied three times to an image with for the first round $x = 0.1$, for the second round $x = 0.2$ and for the fourth round $x = 0.4$. This calculation is performed in three steps, rather than in one, because one step would be too rigorous and would lead to an erosion that is too severe. The values for x were found by conducting some experiments on molars. It should be noted that this erosion approach does not work too well on front teeth, since these are a lot thinner than molars. Example images of the erosion process are shown in Figure 2.4. The separate blobs in the resulting image are given different labels. As a final step, the watershed transform is applied to this labelled image, resulting in a segmented image as in Figure 2.2c and Figure 2.3c.

Since it is assumed that the tooth that needs to be segmented is approximately in the middle of the image, this tooth can easily be picked out by counting which label occurs most in the pixels of an area around the center of the image. In the experiments for this study an area around the center of an image was used of $0.2 \times height(image)$ for the y-axis of the image and $0.2 \times width(image)$ for the x-axis of the image. This simple approach proved to be sufficient as it almost always selected the right segmented blob from the image.

One problem that arose is shown in Figure 2.3c. In few axial slices 'foothills' into another teeth showed up. This is explained by the relief profile of the grayscale image. In case of Figure 2.3c, the



(a) Binary image of tooth before erosion. The white blobs represent the contours of teeth.



(b) Binary image of tooth after erosion

Figure 2.4: Examples of a good tooth segmentation. The white blobs represent the centers of the teeth.

intensity of the outer layers of the orange and green teeth are the same (see Figure 2.3a). Therefore, when the virtual water levels rise, part of the green tooth would be assigned to the orange tooth. A solution to this problem is to compute the centroids of the labelled local minima, and then assign each pixel classified with the watershed transform to the nearest centroid. However, this technique is quite crude and performed very poor in areas where there were also blobs from bone tissue (e.g. yaw bone). Besides that it would sometimes chop big chunks from a tooth and assign it to the adjacent tooth. Since the amount of cases to which the problem depicted in Figure 2.3c applies was limited, this study did not apply the distance to centroid solution.

2.3.3 Program

The program for the watershed segmentation as described above was written in the Python programming language.

2.3.4 Validation of segmentation

The performance of the segmentation is done by taking the intersection over union (IoU, also known as the Jaccard Index) of a tooth slice segmented by an expert and a tooth slice segmented by the system. The IoU is an evaluation metric used for measuring the accuracy of a segmentation. The IoU is for example used as a loss function in foreground and background classification tasks of object in an image (Rahman and Wang, 2016). The formula for the IoU is quite simple:

$$IoU(A, B) = \frac{|A \wedge B|}{|A \vee B|} \tag{2.3}$$

in which A and B are binary axial images of a tooth segmented by an expert and a tooth segmented by the system, respectively. The outcome yields an accuracy of overlap. It must be stated that the IoU penalizes just a slight shift in overlap quite heavily. A good overlap has an IoU > 0.6, and an excellent overlap has an IoU of > 0.9. A good example of two tooth slices over which the IoU is computed can be found in Figure 2.5. These slices fit well and will lead to a high IoU accuracy.

2.3.5 Experiments

The only way to extract a tooth segmented by an expert slice by slice was to convert the slices in which a particular molar occured to an image format (.png). Since the whole image was in gray scale, with exception of the tooth, the tooth could be extracted slice by slice from the images. After that the tooth was converted to a binary format. As a last step the IoU accuracy was computed for an expert slice and a slice segmented by the system.

Experiments were conducted on 11 teeth, which also had been segmented by an expert. These teeth were all molars. There were no segmented front teeth available. Some patient scans had 2 segmented teeth (see Table 3.1 in Section 3).

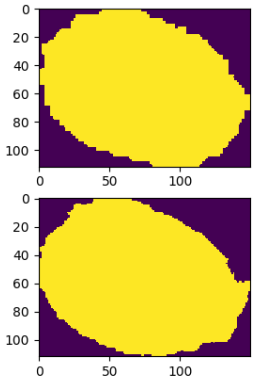


Figure 2.5: Example of an axial tooth slice segmented by an expert (top) and an axial tooth slice segmented by the system (bottom).

During testing of the system it appeared that the IoU performed a lot worse for slices on the top of a tooth and for slices containing the root of a tooth. Therefore the system was tested on both whole tooth and the middle section of a tooth only. This middle section was the whole tooth with the root and top of the tooth (uneven top surface of a molar) excluded. A visual representation of the middle tooth can be found in Figure 2.6.

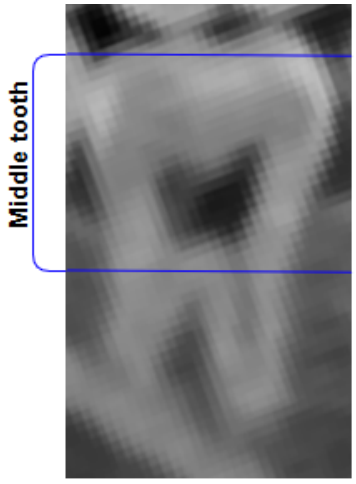


Figure 2.6: Posterior-Anterior cross-section of a molar. Middle tooth section is marked between blue lines.

3 Results

3.1 Convolutional neural networks

In Figure 3.1 the results of the threshold predictions of the 2D-CNN (blue) and 3D-CNN (red) for each patient are shown, as well as the correct threshold values (black). The 2D-CNN needed only 5 epochs to converge to a stable loss, whereas the 3D-CNN needed 50 epochs to converge to a stable loss value. This is explained by the fact that the amount of training examples is a lot larger for the 2D-CNN (see Section 2.2.1), so the model was trained on more data per epoch. Besides the dataset for the 3D-CNN being smaller, a 3D-CNN also takes longer to converge since a 3D network consists of more parameters.

3.2 Watershed segmentation

In Table 3.1 the results are found of the obtained IoU scores of the system. In the 3rd column the IoU score of the system on whole molars is found, while in the 4th column the IoU score on only the middle section of the same molars is found, as described in Section 2.3.4.

There seems to be a significant difference between the score on whole tooth segmentation and middle tooth segmentation. To test whether this difference in score is significant, a two-tailed t-test over the 3rd (whole tooth scores) and 4th (middle tooth scores) column of Table 3.1 is performed. This t-test yields a p-value of 0.002. From this it is concluded that there is a significant difference in scores between whole tooth and middle tooth segmentation.

Some scores seem a little low. However, this has a reason. It must be noted that, tooth 1 of patient 4 had a severely skewed root. Tooth 1 of patient 11 had a very uneven surface at the top of the crown, as well as a brace attached to the tooth. Brace parts have a high return value in HU and will therefore also be regarded as part of the tooth. Tooth 1 of patient 14 also suffered from noise caused by a brace.

4 Discussion and conclusion

4.1 CNNs

The results in Section 3 show that the CNNs seem to learn the mean threshold value of the whole dataset. The cause for this is that the trained model underfits the data. There are various reasons that may cause this problem. First, the pre-processing of the data may not be optimal. One wants to extract sub samples of teeth from a CBCT-scan. However, this has to be done automatically since doing this manually is arduous labour. However, by automatically segmenting the teeth based on their intensity, some bone structures (from the maxilla and mandible bone) are also included in the dataset. Moreover, the data of one patient is cut into small sub-samples. The network is trained on these sub-samples. The final predicted threshold is composed of the mean prediction of all sub-samples from one patient. This may be problematic, since the sub-samples of a patient that are characteristic w.r.t. the threshold value, and thus are given a prediction by the model that is close to the correct prediction, are cancelled out by the predictions of other sub-samples that are less informative for the threshold value. A first step to tackle this problem may be to feed the model with complete axial slices. However, since the resolution of these slices is at least 512×512, performing convolutional operations on such large images requires a lot of computational power. Downsampling these images may result in a loss of valuable information, and can therefore not be considered.

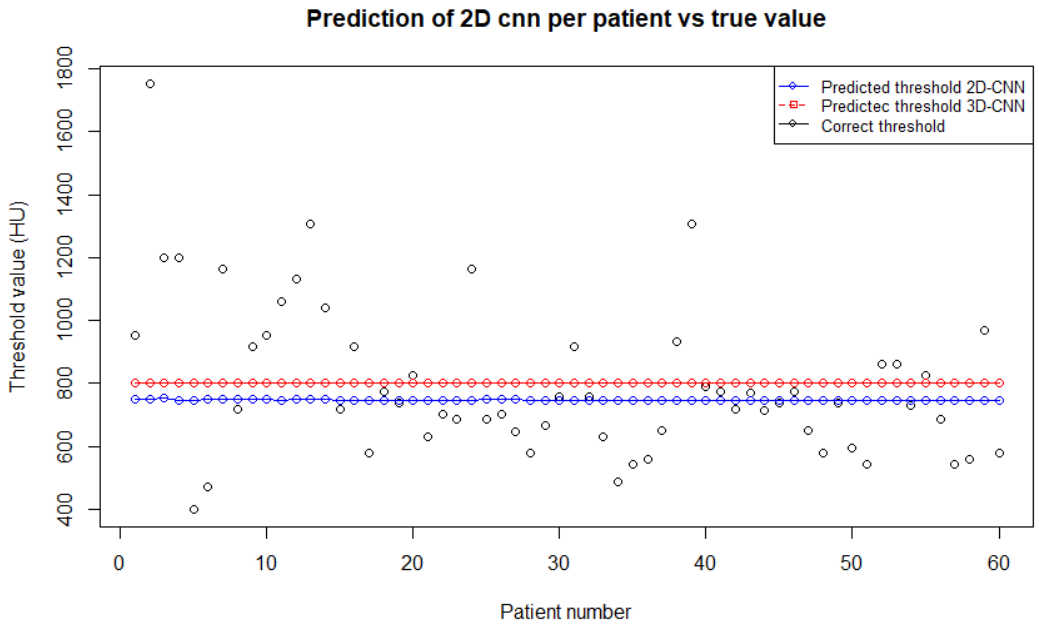


Figure 3.1: Threshold predictions of the 2D-CNN (blue) and 3D-CNN (red) per patient. The predictions are not the same values for each patient. They vary a little, which may be hard to see in the graph.

A second reason for the underfitting of the model is that the model might be too small for a regression problem. However, simpler models have been tried on the dataset (even models with only 1 convolutional layer). Also models with fewer features in the convolutional layers have been tested. These models models showed underfitting issues. These models also converged to mean threshold values very similar to those of the bigger model.

Lastly, fitting a regression function to the data may require a regression function that is too complex for a standard convolutional neural network to learn. Therefore, convolutional neural networks (at least the one described in this paper) may not be the solution to automatically thresholding in CBCT scan imagery. The study in this paper is of an exploratory type, therefore it was and is not certain whether the approach used in this paper will work or not. However, for future work we recommend more accurate pre-processing of the data, such that the train data only includes less non-teeth data. Moreover, training the model on CBCT datasets of more patients may lead to better results. Lastly, other, possibly larger models may be used to learn regression on CBCT datasets.

4.2 Segmentation with watershed

This study shows that watershed segmentation is able to segment a tooth from a CBCT scan quite well. Especially from the middle section of a tooth (including the part where teeth are in close contact). The technique used in this paper performs significantly better in the middle tooth region than on the whole tooth region. The mean whole tooth IoU score is 0.67, whereas the mean middle tooth IoU score is 0.79.

The results show that the segmentation approach, as described in Section 2.3, does not perform very well in areas that contain multiple blobs belonging to the same tooth. These are the areas at the top of the crown (as in the top part of Figure 2.2a), and the radix of a tooth. Moreover, segmenting the root of the tooth is even more challenging because of the root entering the jaw bone, and the similarity in radiodensity between dentin and bone.

Table 3.1: IoU score per tooth for whole tooth and for middle tooth

Patient number	Tooth number	IoU score whole tooth	IoU score middle tooth
3	1	0.69	0.78
3	2	0.65	0.72
4	3	0.56	0.79
8	4	0.71	0.82
9	5	0.74	0.88
11	6	0.49	0.73
12	7	0.77	0.74
12	8	0.76	0.92
14	9	0.60	0.76
15	10	0.72	0.72
15	11	0.68	0.83
Total score		**0.67**	**0.79**

Bone would touch the dentin in some cases causing the segmentation approach described in this study having difficulty deciding which blobs belonged to which tooth, and which blobs (i.e. bone) did not belong to a tooth at all.

A solution to such problems would be to introduce a method that takes into account the 3D structure of a tooth. Such a method could for example consist of an approach that examines previously segmented slices. When the bulbs in the slice above (or below) overlap with a previously segmented slice, the bulbs are regarded as the same tooth. However, this would require a system that works its way in cranial and caudal directions from slices in the middle of the tooth crown, since the intersection area of the tooth is the biggest here.

Also, a more robust way of finding the centers of the teeth (which are used for the marker driven watershed transform) is needed in the future. In this study, an erosion algorithm is used. In doing so, parameters have to be adjusted when dealing with molars or front teeth. If one were to use the parameters used for molars on front teeth, chances are that the whole front tooth would be gone after the erosion algorithm.

Future studies should focus on ways of automatically thresholding teeth in CBCT scans, perhaps using different approaches as convolutional neural networks. Another expansion on automatically segmenting teeth from a CBCT scan would be to train a CNN to detect the locations of specific teeth in a thresholded CBCT scan, specified by their dental notation, and then automatically segment that tooth. For such a system, only entering the dental code of a tooth would suffice to get a 3D model of a tooth. However, this would require a model that learns tooth specific features (like location and shape). This approach requires a large dataset of CBCTs of separate teeth and constructing such a dataset will take a lot of effort in the pre-processing step. In conclusion, this study shows that watershed is a technique that performs well in segmenting teeth from a CBCT scan.

References

Wafaa Alakwaa, Mohammad Nassef, and Amr Badr. Lung cancer detection and classification with 3d convolutional neural network (3D-CNN). *Lung Cancer*, 8(8), 2017.

Dan C Ciresan, Ueli Meier, Jonathan Masci, Luca Maria Gambardella, and Jürgen Schmidhuber. Flexible, high performance convolutional neural networks for image classification. In *IJCAI Proceedings-International Joint Conference on Artificial Intelligence*, volume 22, page 1237. Barcelona, Spain, 2011a.

Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jurgen Schmidhuber. Convolutional neural network committees for handwritten character classification. In *Document Analysis and Recognition (ICDAR), 2011 International Conference on*, pages 1135–1139. IEEE, 2011b.

Tobias Fechter, Sonja Adebahr, Dimos Baltas, Ismail Ben Ayed, Christian Desrosiers, and Jose Dolz. Esophagus segmentation in CT via 3d fully convolutional neural network and random walk. *Medical physics*, 44(12):6341–6352, 2017.

Rafael C Gonzalez and Richard E Woods. Digital image processing second edition. *Beijing: Publishing House of Electronics Industry*, 455, 2002.

Tobias Heimann and Hans-Peter Meinzer. Statistical shape models for 3d medical image segmentation: a review. *Medical image analysis*, 13(4): 543–563, 2009.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

Hans Lamecker, Dagmar Kainmueller, Stefan Zachow, et al. Automatic detection and classification of teeth in CT data. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 609–616. Springer, 2012.

Norberto Malpica, Carlos Ortiz de Solorzano, Juan José Vaquero, Andrés Santos, Isabel Vallcorba, José Miguel García-Sagredo, and Francisco Del Pozo. Applying watershed algorithms to the segmentation of clustered nuclei. *Cytometry*, 28(4):289–297, 1997.

Md Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International Symposium on Visual Computing*, pages 234–244. Springer, 2016.

Li Kuo Tan, Yih Miin Liew, Einly Lim, and Robert A McLaughlin. Convolutional neural network regression for short-axis left ventricle segmentation in cardiac cine MR sequences. *Medical image analysis*, 39:78–86, 2017.

Luc Vincent and Pierre Soille. Watersheds in digital spaces: an efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6):583–598, 1991.

Xiaodong Yang, Houqiang Li, and Xiaobo Zhou. Nuclei segmentation using marker-controlled watershed, tracking using mean-shift, and Kalman filter in time-lapse microscopy. *IEEE Transactions on Circuits and Systems I: Regular Papers*, 53(11):2405–2414, 2006.

Yan Ling Yong, Li Kuo Tan, Robert A McLaughlin, Kok Han Chee, and Yih Miin Liew. Linear-regression convolutional neural network for fully automated coronary lumen segmentation in intravascular optical coherence tomography. *Journal of biomedical optics*, 22(12):126005, 2017.