Technische Universität Berlin
Fakultät IV Elektrotechnik und Informatik
Fachgebiet Computer Vision,
Zuse-Institut Berlin

Localization and Classification of Teeth in Cone Beam Computed Tomography using 2D CNNs

in Fulfillment of the Requirements

for the Degree of Msc. Computer Engineering

Mario Johannes Neumann
Matr. Nr. 327468

April 30, 2019

Supervised by:

Prof. Dr. Olaf Hellwich
Dr. Stefan Zachow

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich die vorliegende Arbeit selbstständig und eigenhändig sowie ohne unerlaubte fremde Hilfe und ausschließlich unter Verwendung der aufgeführten Quellen und Hilfsmittel angefertigt habe.

Berlin, den

Zusammenfassung

Computerbasierte Datenverarbeitung in der Medizin ist bereits Bestandteil des täglichen Lebens. In der Zahnmedizin bieten softwaregestützte Visualisierungs- und Planungswerkzeuge Möglichkeiten für vereinfachte und bessere Therapie- und Diagnostikansätze.

Wir stellen ein Verfahren zur automatischen Erkennung von Zahntypen und Positionen in digitaler Volumentomographie (DVT) vor. Durch den Einsatz von modernen Ansätzen des Deep Learnings in Kombination mit Dimensionsreduktion durch Projektion und nicht-planare Umformatierung der Kieferoberfläche können die 3D Daten effektiv verarbeitet und Zähne zuverlässig erkannt werden.

Um gleichzeitig mehrere Zahnpositionen zu bestimmen, betrachen wir Vor- und Nachteile zweier verschiedener Lösungen mittels faltender Neuronaler Netze (Convolutinal Neural Networks). Der beste Ansatz, ausgewertet auf einem medizininschen DVT Datensatz, identifiziert Zähne zu 94% korrekt bei einer örtliche Genauigkeit von weniger als 2mm Abwichung im Vergleich zu von uns manuell gesetzten Landmarken.

Contents

1 Introduction

Computer-assisted analysis of medical images is gaining more and more importance in modern day diagnosis and therapy planning. Recently, deep learning techniques show promising results in various applications. They aid in diagnosis, predicting disease probabilities, guiding surgeons with accurate distance measuring, and detecting relevant anatomical parts in radiographic images.

The remainder of this chapter motivates the application in dental imaging, shows potential challenges, and introduces the task to localize and classify teeth in cone beam computed tomography. This is followed by a summary of our general approach, a review of related work, and lastly an overview of the structure of this.

1.1 Motivation

Many tasks in image analysis comprise not only the classification of a single image but also accurate localization of objects or landmarks indicating anatomical points of interest in the images. Also segmentation techniques that analyze the shape of an object by mapping each pixel in the image to a corresponding object are common in medical image analysis. Throughout the last years, advances in general artificial intelligence (AI) and image recognition were properly adapted to various domains resulting in the recent success in e.g. medical and biological image analysis.

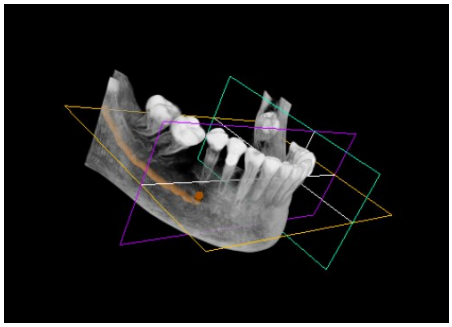


Figure 1: Volumetric rendering of a cone beam CT scan of the lower jaw[1]. The dental anatomy is clearly visible and e.g. the mandibular nerve canal can be properly inspected and marked as done in the image.

In contrast to photographs or x-ray images, computed tomography (CT) or magnetic resonance imagery (MRI) compose volumetric data and empower three di-

[1]Image by Dent3D, license: CC BY 3.0, https://de.wikipedia.org/wiki/Digitale_Volumentomographie#/media/File:Gerendertes_DVT_mit_Nervdarstellung.jpg. Accessed Apr.18th,2019

mensional insight into the human body. Cone beam computed tomography (CBCT) provides volumetric data at short scanning times at low radiation exposure which is why it is a well suited visualization method for the cranio-facial (head and skull) area where many anatomical structures are at risk and need to be protected from high radiation doses. CBCT is widely used in dentistry and for diagnosis of skull, ear-nose-throat, and neck area. Due to good representation of dental roots and the alveolar nerve (Fig. 1), dental CBCT is becoming a standard tool in dentistry and orthodontics used both, for diagnosis and for therapy or implant planning.

The identification of the dentition is an often conducted procedure required for therapy planning, standardized communication between physicians, or even for data acquisition in post-mortem studies [PMS12]. Therefore, a unified notation of the human dentition is needed. Figure 2 and Table 1 illustrate tooth types and the enumeration codes internationally used by dental practitioners. Figure 3 shows a full enumeration of the opened mouth.

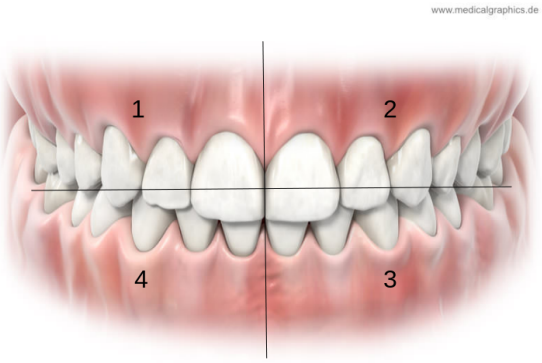


Figure 2: Frontal view onto the human dentition[2]. The sectors are labeled according to the World Dental Foundation (FDI: Férdération Dentaire Internationale). The sides are named from the patient view: 1= upper right, 2 = upper left, 3 = lower left, 4 = lower right.

Table 1: FDI two-digit notation for permanent teeth. The tooth types are enumerated from the middle towards the outside.

1st digit	Quadrant
1	Upper right (UR)
2	Upper left (UL)
3	Lower left (LL)
4	Lower right (UR)
2nd digit	**Tooth Type**
1	1st incisor
2	2nd incisor
3	canine
4	1st premolar
5	2nd premolar
6	1nd molar
7	2nd molar
8	3rd molar

[2]Image by www.MedicalGraphics.de, license: CC BY-NE 4.0, http://www.medicalgraphics.de/en/free-pictures/organs/upper-lower-jaw-teeth-frontal.html. Accessed Apr.18th,2019

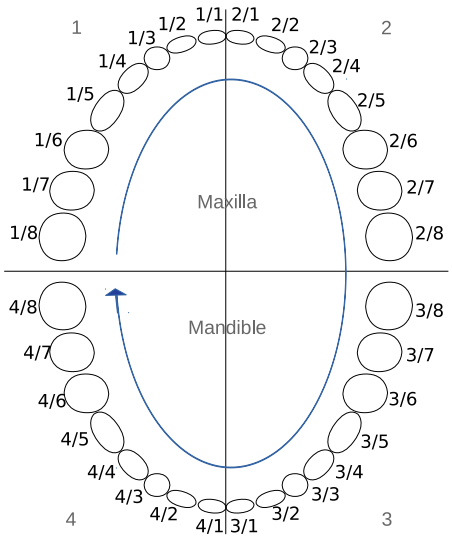
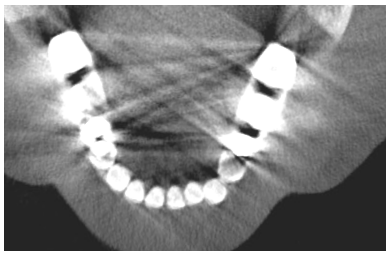


Figure 3: The graphic resembles a dentist view into the opened mouth with full dentition. The teeth are numbered after FDI with sector / tooth number. For computation, we can simply enumerate the teeth by increasing order along the blue arrow.

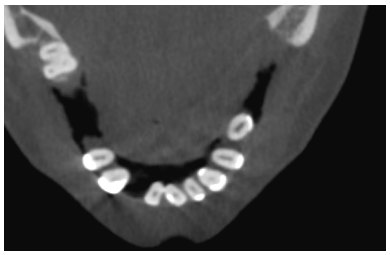
An automatic analysis of tooth positions and types from dental images can help with measuring distances e.g. to find an equidistant spacing in CBCT guided implant planning. On the one hand, dental practitioners would benefit from an automatic classification of the teeth since this can speed up the documentation process and reduce potential human errors. On the other hand, it could be used to evaluate tooth positions and relative distances on a large scale to provide statistical data. Moreover, the predicted locations can be used to automatically extract regions of interest for further image processing.

1.2 Challenges

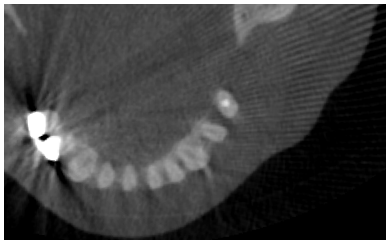
Analyzing medical images comes with a set of challenges: Clinical data may not be accessible publicly, images can contain artifacts, and volumetric data comes with higher computational complexity. In the field of deep learning, the capability of a neural network is highly enforced by numerous and diverse high quality images. For medical images however, this is not guaranteed. Due to privacy protection of the sensitive data, the huge variety in thousands of images that can be used in other domains (see e.g [DDS+09]) may be difficult to obtain and the training data might not contain all anatomical variation that is theoretically possible. Images can be distorted, can have limited resolution or may contain artifacts due to the technical and physical properties of their acquisition.



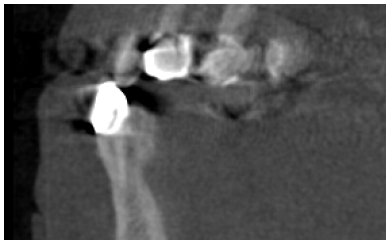
(a) Metallic artifacts due to dental crowns.



(b) Multiple missing teeth and therefore difficult numbering.



(c) Metal occlusions (left) and aliasing pattern (diverging lines).



(d) General low image resolution resulting in low contrast of dental roots (sagittal view).

Figure 4: Artifacts and challenges is dental CBCT data.

A lot of work in image recognition is done on natural images (i.e. photographs) in 2D. Applications in 3D are more difficult simply due to the increased computational complexity caused by the additional dimension. Dimension reduction methods can help to extract relevant features and reduce the problem complexity but even then, those methods have to be applied thoroughly, s.t. no necessary information is lost.

Particular difficulties arise for dental imaging. Figure 4 shows possible challenges that can occur in real-world dental CBCT. Occlusion artifacts, caused by metal implants and movement artifacts both are common and may heavily degrade the image quality. Missing teeth, implants, and anatomical outliers may be hard to categorize. Furthermore, the goal of simultaneously detecting and labeling multiple objects in one image is significantly harder than just classifying an image or detecting objects that are independent from each other.

1.3 Assumptions

In order to consistently perform the classification and localization task and to fit the workload into this thesis, we restrict our approach and its application. Without losing generality of the methodology, we reduce the amount of pre-processing by only regarding the lower jaw and the lower teeth row, respectively. Although

Table 2: Table for tooth entries. For each group of teeth the number of columns corresponds to the number of teeth in the anatomically intact model as shown earlier in Figure 3.

Tooth	Number	x	y
LL molar	0	0.885	0.535
	1	0.788	0.491
	2	-	-
LL premolar	3	0.658	0.508
	4	0.	0.
LL canine	5	0.603	0.467
	⋮	⋮	

the segmentation of the maxilla (upper jaw) is more difficult due to the fine bone structure, the findings of this thesis should be transferable analogously to the teeth of the maxilla.

Despite coping with implants and artifacts in the training data, for this first investigation, we stick with our goal to classify teeth but do not extend this to classify fillings or even different implant types. As long as the shape of the tooth is visible, we classify the tooth regardless of any fillings, crowns, or bridges.

While we indeed want to notate missing teeth, we need to restrict the total number of teeth we can find. Since CNNs have a finite number of outputs, we have to limit number of tooth entries to a fixed maximum. Therefore, we reserve only entries for three lower left molars, two for lower left premolars etc. The entries can be represented in a tabular ordering as in Table 2.

1.4 Objectives

The main goal of this work is to locate and classify teeth in dental CBCT data. In order to reduce the problem complexity we need to extract and process relevant features of the data. We have to find and adapt appropriate CNN architectures and train them to perform the desired task. In this process we strive to:

- Extract relevant features using and existing mandible segmentation
- Investigate deep-learning approaches to identify tooth locations
- Process and transform data into a suitable format for a CNN architecture
- Cope with missing teeth in representation and in learning architecture.

- Review data augmentation methods in order to cope with the size of our training data
- Achieve a practically useful prediction accuracy

1.5 Approach

We address the task of finding and classifying teeth in CBCT data. To reduce the problem complexity, we perform dimension reduction by projection and non-planar reformatting. This allows us to use standard image processing tools and benefit from established 2D CNN architectures.

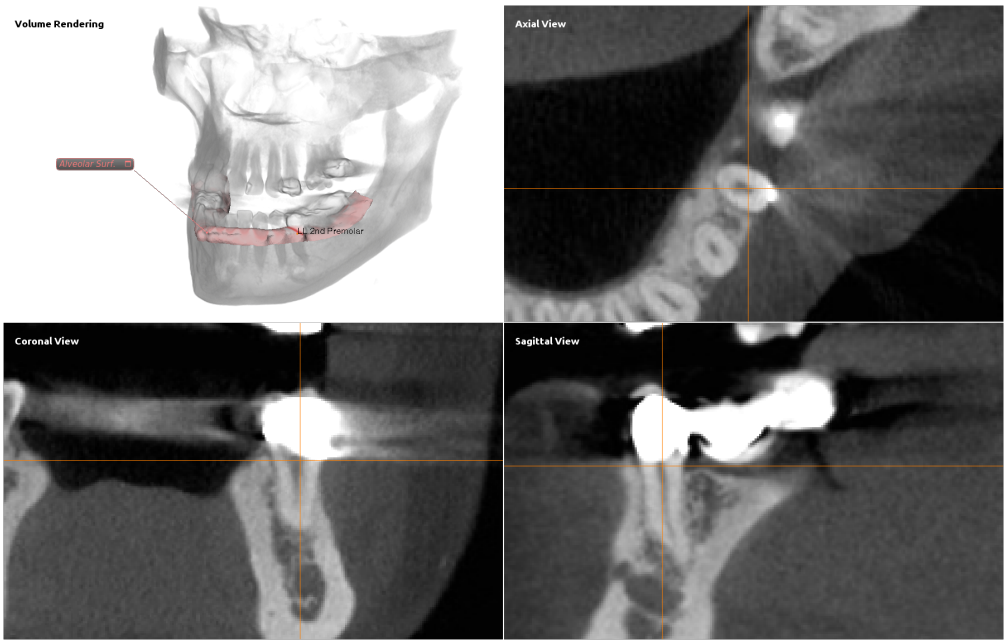


Figure 5: Volume rendering and orthogonal slices visualizing the lower left canine. Although the neighboring teeth are covered by a bridge, the break through interface of the tooth is clearly visible in the axial slice and the root is visible in sagittal an coronal view. The volume rendering shows the surface of the dentition bone interface that we want to extract.

In order to extract significant features, we select a surface that cuts through the teeth at the intersection of teeth and bone surface of the mandible (crest of the alveolar process). We will term this region as tooth bone interface, dentition bone interface, or break through region. Figure 5 shows the location of the used surface and perpendicular slices centered on the lower left canine.

The hypothesis is that the surface of the break through region provides relevant and reliable features. The dental roots may be hard to distinguish from their surrounding due to low difference in material density and the view of the crown as well as of the dentin may be distorted by image artifact caused by fillings or dental bridges. As visible in Figure 5, the shape of the actual tooth is hardly recognizable due to the dental bridge. The axial slice at the height of the break through interface has a significant shape that is usually not occluded by copings or dental bridges.

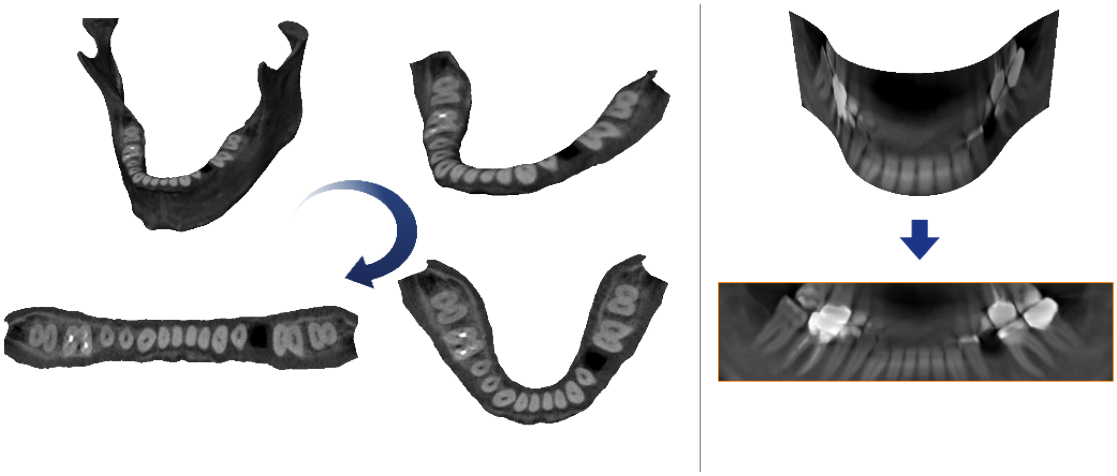


Figure 6: Surface Transformation. The surface containing the tooth-bone interface is extracted and flattened into a planar U-shape that is bent up in the next step (left). Additionally, a panoramic image is generated from a curved slice following the dental arch (right).

The goal is to generate an image of this dentition area. We take the surface of the mandibular bone generated by an existing segmentation method and extract the relevant region. To reformat this into an optimized input image for a 2D CNN, we use surface transformations to flatten the surface and to normalize it into a rectangular frame. Figure 6 visualizes the surfaces. We will term the 2D image of the straight row of the dentition on the surface of the mandible as "Break Through Image" (BTI) in the remainder of this thesis. To further improve the capability of our approach, we investigate the combination of these images with an artificially panoramic radiograph (APR) image that is similar to the commonly used panoramic x-ray images. It is virtually generated from the same CBCT data. The CBCT data, the SSM based segmentation, as well as the synthesized APR were kindly provided to us by 1000Shapes GmbH, a ZIB spin-off company.

Before we can train a CNN in a supervised manner, we need to manually generate ground truth annotations for the tooth positions in the training data. To label the

positions efficiently, we developed a software plug-in for the ZIB-Amira framework to interactively annotate tooth position. The tool aids in the manual labeling process and automatically calculates the landmarks on all surfaces and training images from annotated 3D positions on the segmentation surface of the mandibular bone.

In the next step, we investigate two existing CNN architectures and adapt them to our problem. We train the networks based on our ground truth annotations to predict landmark positions for all teeth in new images and finally track back the 3D positions by a reverse flattening. To evaluate our approach, we compare correct labeling of the teeth, i.e. annotating the correct number, and the positioning accuracy. The whole processing pipeline is visualized in Figure 7.

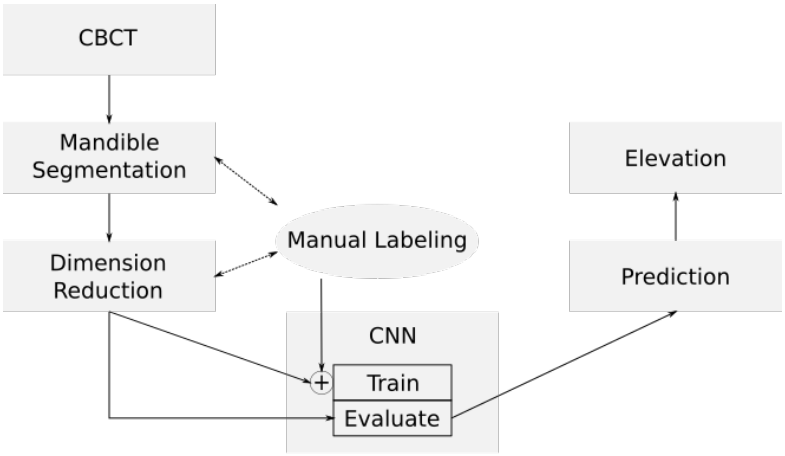


Figure 7: Data flow scheme. From the generated CBCT data, we extract the segmented surface of the mandible, reduce the dimensionality by generating 2D images of the dentition region. The manually set landmarks are used to train a CNN. Then, new data is evaluated on the trained CNN and lastly tracked back the 3D tooth position.

1.6 Related Work

In order to relate our approach to similar studies, this section reviews image analysis with dental application and tooth detection. Although there has been research applied to panoramic radiograph images for e.g. caries detection [ISB], we focus on published works on CBCT data and especially classification or localization of the teeth.

Gao et al. [GC10] segment individual teeth with level set bases methods in CT images. Further classical image analysis approaches aim to classification and num-

bering with appearance and shape based features classified by support vector machines [NALA08], [HAZATFS10], [YSNAA12].

Duy et al. [DLKZ12] separate the dental region into 32 3D neighboring bounding boxes. They select the region of the dentition based on a model segmentation of maxilla and mandible and optimize the separation into sub-regions as a graph problem. The classification is done implicitly by the ordering of boxes but additionally the decision whether a tooth is present within the box is done by a support vector machine with the intensity histogram of the region as input. Using the graph based approach, they are able to realize anatomical ordering constraints and achieve good detection results. However, this only holds if the initial separation is correct. If it is inaccurate, which can likely be the case if multiple teeth are missing or teeth are crowded (touching), the error propagates to other regions and the classification might also fail. Furthermore, they only regard intensity features for classification and neglect the shape information. This makes the approach inadequate to deal with tooth decay or temporary implants like screws or pins that might occur in medical data.

Miki et al. [MMH+17] aim to classify the dentition in axial slices. They manually select the 3D region of interest capturing a set of axial slices of the complete tooth. Thus, they have to do this procedure for training and for every new data again. They classify the teeth in a regional CNN (R-CNN) approach [GDDM14]. That means, they merely focus on the local appearance of the teeth by implementing a sliding window with an AlexNet [KSH12] or VGG [SZ14] network architecture in an improved version of their work.

Recently Macho et al. [MKU+18] as well as Ezhov et al.[EZG18] proposed multi-level, multi-resolution based simultaneous segmentation and localization. They first train a coarse (down-sampled) model with roughly segmented training images to predict regions of interest. In a second step, they train another network on accurately segmented training images to segment cropped regions of the coarse stage. While Macho et al. address tooth segmentation in general, Ezhov et al. also divide the segmentation in 32 channels identifying the different tooth numbers. Since they do not automatically generate dental charts and tooth locations, it is hard to evaluate their approach w.r.t. classification accuracy.

Even when regarding the last approaches where the knowledge about tooth positions is implicit by the segmentation, we know of no such method to reliably localize and classify teeth in volumetric medical imaging data.

1.7 Structure

While this chapter placed our work in the field of medicine and image analysis, the following parts of this thesis are structured as following.

In Chapter 2, we summarize basic knowledge for the later on used methodologies. The main part is split into two chapters. The first is a detailed explanation of our approach (Chapter 3) and the second the evaluation of different experiments (Chapter 4). Chapter 5 discusses our findings and gives an outlook towards future work.

2 Preliminaries

To understand the medical application, the image processing, and machine learning algorithms that will be used later, this chapter covers the necessary background knowledge. The first part explains how volumetric images are generated by CBCT and the second gives an overview of medical image processing methods covering the segmentation method we rely on and deep learning approaches for landmark or object detection.

2.1 Dental Cone Beam Computed Tomography

Dental CBCT is a special kind of CT imaging that can be used to generate 3D images of the maxillo-facial region. It visualizes soft and hard tissues as jaw bones, sinuses, nerve canals, and the nasal cavity. Because the machinery has become smaller and prices have fallen throughout the last decade, CBCT is becoming standard equipment in many dentist, orthodontics, and ENT (ear, nose, throat) facilities.

Panoramic or cephalometric radiographs have been commonly used for diagnosis and therapy planning in dentistry and orthodontics. 3D tomography, however, is a preferred imaging method for surgical planning and navigation, since it provides full volumetric information. CBCT provides 3D imaging with good results at relatively low radiation exposure and fast scanning times. With an average radiation dose of 36.9-50.3 microsievert[μSv] [SFS06], the exposure is still higher than for a panoramic radiograph but significantly less compared to an equivalent CT scan.

2.1.1 Imaging Principle

X-rays are attenuated differently by different materials. Different bone or tissue densities result in intensities that directly represent the accumulated absorption along the radiated beam. By taking multiple projections from different directions, physical properties of the object can be determined by the principles of the radon transform [Rad17] and its inverse, respectively. However, in practice of tomographic reconstruction, highly optimized, discrete, and iterative versions of this theory are applied.

In regular CT, a fan beam is used to construct slices through the patient that are coplanar to the gantry. The radiation source rotates around the patient to generate multiple 1D projection rows which are measured by an also rotating detector. With the projected rows a slice through the person's body can be calculated using a reconstruction algorithm. Since the radiated beams of a fan are not parallel, a

reordering of all beams and attenuation values, such that they are grouped into sets of parallel rays, may be done to use a standard reconstruction method. After each step, the patient is moved in z-direction (along the longitudinal axis) by a fixed increment and the next slice is generated. Doing so, a volumetric scan with configurable number and thickness of the slices can be generated.

The original CBCT reconstruction algorithm was introduced in 1983 by Feldkamp et al. [FDK84]. A 3D volume is reconstructed from a set of 2D projections. CBCT uses a cone shaped beam to generate the 2D radiographic projections and can be seen as natural extension of the fan beam reconstruction. In contrast to regular CT a volumetric reconstruction can be obtained by only one rotation of radiation source and detector around the object. However, by only using one rotation, the back calculation of the volume becomes less accurate at locations that are hit by beams with a higher opening angle of the cone. Although the CBCT reconstruction is only an approximation to the 3D volume at these locations, it introduces only small errors while being computationally efficient.

2.1.2 Apparatus

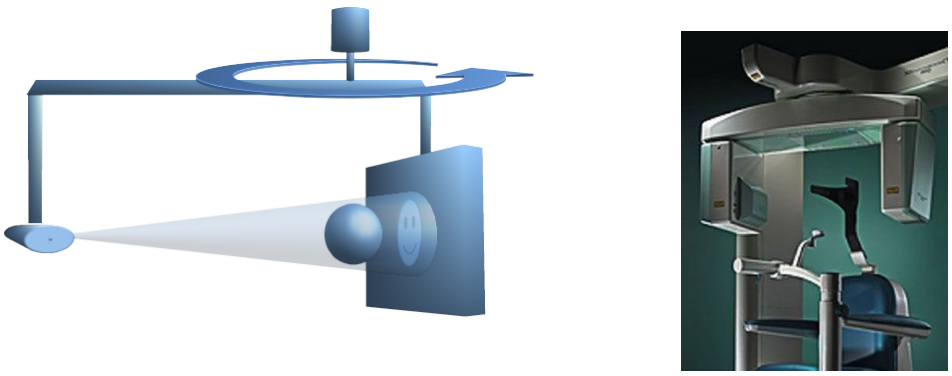


Figure 8: Cone Beam CT apparatus: Simplified scheme (a) and vendor image of the Accuitomo F170[3](b). The beam source and the detector array are fixed on a gantry that rotates with the probe being in the axis of rotation.

For dental CBCT, the patient has to stand or sit upright with the head between the radiation tube and the detector. Figure 8 shows an exemplary CBCT machine and a visualization of the functionality.

Depending on the application, the placing can be done by biting onto a mouthpiece or by fixation of the head. The available devices may have different detectors,

[3]Image by J.Morita Corporation, https://www.morita.com/anz/en/products/diagnostic-and-imaging-equipment/cone-beam-ct-systems/3d-accuitomo-170/?tab=media. Accessed Apr.2nd,2019

radiation tubes, and geometry. The tube voltage and the field of view are usually adjustable resulting in scanning times of 10-70 seconds [SFS06]. Due to the scattering of the beams in a cone shape, the side not facing the radiation source has lower image quality. Especially if the region of interest is only on one side of the scanned object, a full rotation is not necessary and radiation exposure can be reduced by generating fewer projections. In general, CBCT machines can provide isotropic voxel shapes and can provide resolutions in sub mm range. Figure 9 shows a use case of a CBCT scan in a typical dentist software.

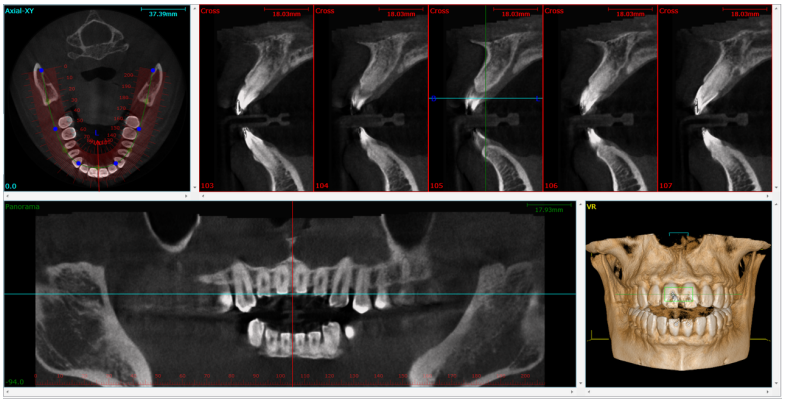


Figure 9: The image shows multiple views and slices through a volumetric CBCT scan including an axial view (upper left), multiple sagittal views (upper right), a synthesized panoramic image along a manually defined curve (lower left) and a volumetric rendering (lower right)[4].

2.2 Image Analysis

Medical image analysis deals with finding keypoints, segmenting anatomy, or predicting probabilities for medical conditions. These analyzes are based on images of the human body that can be obtained by a variety of techniques (e.g. CT, MRI, ultrasound, etc.).

This section briefly discusses segmentation methods, in particular model based segmentation using geometric priors with a statistical shape model (SSM) because we rely on such an approach for the segmentation of the mandible. We give an overview of surface transformations that can be used to embed 3D surfaces into 2D images and finally, we review CNN architectures and their application for landmark and object detection, respectively.

[4]Image by Panda 51, CC BY-SA 4.0, https://en.wikipedia.org/wiki/Cone_beam_computed_tomography#/media/File:CBCT_image_03.png. Accessed Apr.2nd,2019

2.2.1 Medical Image Segmentation

Statistical shape models (SSMs) for 3D medical image analysis have been established since the beginning of this millennium [HM09]. They represent variation of a shape across a training population and can be used to match an image against variations of the modeled shape.

An SSM is constructed by analyzing multiple samples of one object and thereby building a distribution over shapes and appearances. Shape is commonly represented as set of points distributed across a surface in so called Point Distribution Models (PDMs). Therefore, every shape can be represented as vector of points: $\mathbf{x} = (x_0, x_1, \dots x_k)^T, x \in \mathbb{R}^n$. The shape model is constructed by aligning the training samples and identifying the mean shape and the main modes of variation. The modes of variation are determined by Principal Component Analysis (PCA). The PCA yields the Eigenvectors of the PDM's covariance matrix. Then, every shape that lies on the manifold spanned by the training data can be described as linear combination of weighted modes:

$$\mathbf{x}_s = \mu + \sum_{n=0}^{C} w_n \xi_n, \tag{1}$$

with $\mathbf{x}_s$ being any particular shape described by the SSM, μ the mean shape, ξ_n the n'th mode of variation (or Eigenvector in PCA terms) and w the corresponding weight necessary to aggregate the shape instance. Since the modes are sorted by Eigenvalues describing the magnitude of variation, the highest variance of a shape is encoded by the first modes corresponding to the highest Eigenvalues. Summarizing only a set of higher modes and dismissing lower ones reduces the SSM complexity while still representing the main statistical variations of the shape (see e.g. [LZW+06]).

Now, the task is to match the shape to a given image. New images are matched against template shapes generated by variation of the SSM modes. In contrast to matching a single template shape, which may be sufficient to detect ridged shapes, matching a SSM is more robust against natural variation of shape, outliers, and image artifacts which all are common in medical images. Figure 10 shows examples of an SSM for the mandibular bone adapted to CBCT image data.

The predominant approaches to adapt an SSM are active shape and active appearance models [HM09]. The active shape model iteratively deforms the shape within the modes of the SSM and tries to fit local keypoints to the image by a comparison based on image gradients. Active appearance models extend this approach by combining the SSM with a learned gray level appearance of the object and then minimizing a difference between the image and the synthesized model instance.

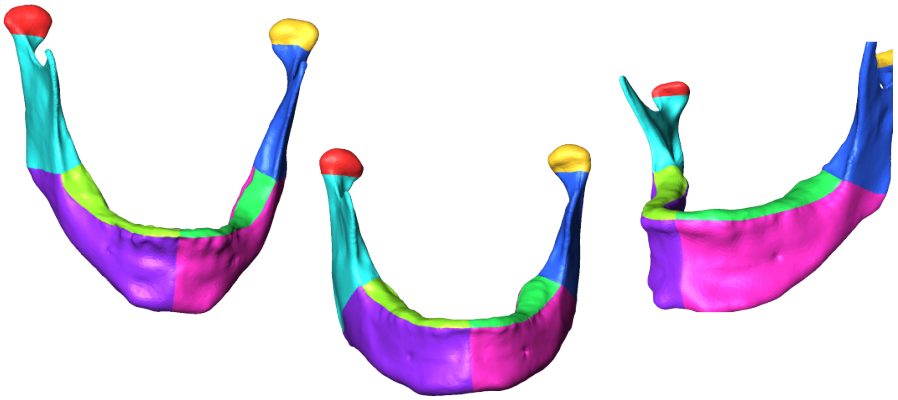


Figure 10: Segmentation of the mandibular bone using a statistical shape model with an approach by Lamecker et al. [LZW+06]. The shape is adapted by transformation and variation of the SSM modes and finding the keypoints of the SSM within the image data. Thus, correspondence of predefined regions of the SSM can be inferred to partition the segmented shape into patches which are indicated by the different colorization.

Throughout the last years, also deep learning based approaches have been greatly successful segmenting medical images. The U-Net architecture by Ronneberger et al. [RFB15] is a predominant architecture used for 2D image segmentation. This architecture will be described in Section 2.2.4 since we use it in the context of landmark detection.

2.2.2 Surface Transformation

In this thesis, we use surface parameterization to generate two dimensional mappings of the dentition surface of the mandibular bone within the CBCT image data. Doing so, we can further process 2D images in an efficient manner while preserving the information of the 3D surface.

Finding a mapping from a 3D surface mesh into a suitable domain for easier processing or vice versa is a common problem in computer graphics, either to reduce the complexity of a 3D mesh or to map a texture onto an arbitrary surface. It can be advantageous to generate two dimensional mappings of 3D image data at that lies on a surface, i.e. to find a surface parameterization.

Depending on the application, these mappings try to minimize distortion of angles (conformal parameterization), the triangle area (equiareal parameterization) or a blending of both. We focus particularly on methods operating on surface triangle meshes that are homeomorphic to a disc. For some methods, a fixed boundary in the planar domain, e.g. a circle or a rectangle, has to be chosen. For others, the

vertices of the boundary are free to move.

Popular approaches are for instance least square conformal maps as introduced by Lévi et al. [LPRM02] or mean value parameterization by Floater [Flo03]. The former minimizes angle distortion while having a free boundary parameterization. The latter generalizes barycentric coordinates to express a vertex in the planar domain as convex combination of its neighboring vertices and a fixed boundary 2D is used.

The method we use was presented by Wang et al. [WLT12]. They derive discrete formulations for the metric distortion of the surface and for the surface curvature (first and second fundamental form). A surface then can be flattened by keeping the first fundamental form fixed and manipulating the second one, i.e. the change of the surface normal. This condition is realized by fixing one normal and prohibit its variation over the surface. This is formulated as a constraint for the transition rotation between neighboring triangles (Fig. 12, defined by their frames (two vectors describing the tangential plane and the on describing the normal).

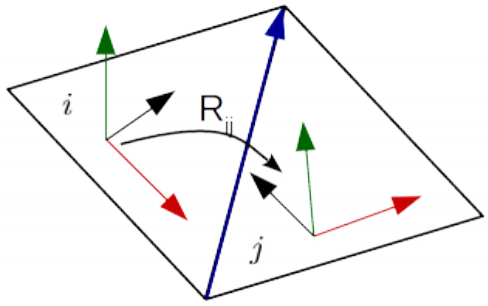


Figure 11: Rotation from one triangle frame to another sharing a common edge. The triangle normals can be aligned by modifying this rotation[5].

A second constraint demands the edge lengths to be preserved as good as possible throughout the mapping. Figure 11 show an exemplary flattening of a facial mesh. Selecting a fixed vertex and a frame from a neighboring triangle, makes this a uniquely solvable linear system. Since this approach transforms the surface while preserving the shape of the triangles as good as possible, we will call it quasi-isometric surface transformation (QIST). The QIST method is appealing to our application since it minimizes the distortion of the triangle and thereby also of the appearance of the tooth contours. This approach was implemented and further adapted by the medical planning group at ZIB (Felix Ambellan).

[5]Image provided by Felix Ambellan (ZIB)
[6]Image provided by Felix Ambellan (ZIB)

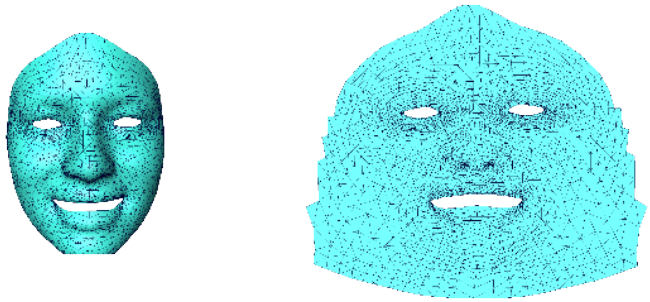


Figure 12: Flattening of a facial surface mesh with the QIST method[6]. The holes for eyes and mouth were meshed and removed again in an intermediate step to flatten a continuous mesh.

2.2.3 Convolutional Neural Networks

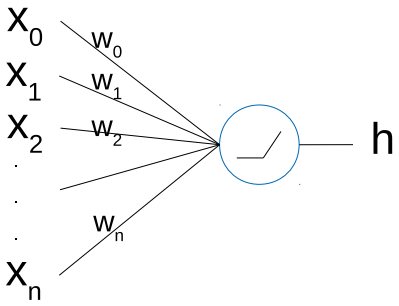


Figure 13: Artificial Neuron. All input signals are weighted and summarized. The result is activated by a non-linear function, typically the ReLU that clamps values smaller than 0.

In deep learning, CNNs are a particular family of Artificial Neural Networks (ANNs) An ANN is constructed by connecting a set of artificial neurons which are inspired by biological neurons in the brain. A model for a single artificial neuron is the perceptron (see Fig. 13). The perceptron computes its output $h \in \mathbb{R}$ by building a weighted sum over the inputs x_i followed by a non-linear activation function Φ, thus output is calculated as

$$h_j = \Phi(\sum_i x_i w_{ij}). \tag{2}$$

By chaining multiple perceptrons with an activation function, highly non-linear mappings from input to output can be realized. In multi-layer perceptrons (MLPs), the artificial neurons are organized in layers as shown in Figure 14. Each neuron

i in layer j gets connected to all neurons $i', j-1$ of the previous layer. For e.g. classification tasks, one neuron is used to build a global output score over all outputs of the last hidden layer. The input data is put into the first layer and usually all consecutive layers take the output of their previous layer as their input.

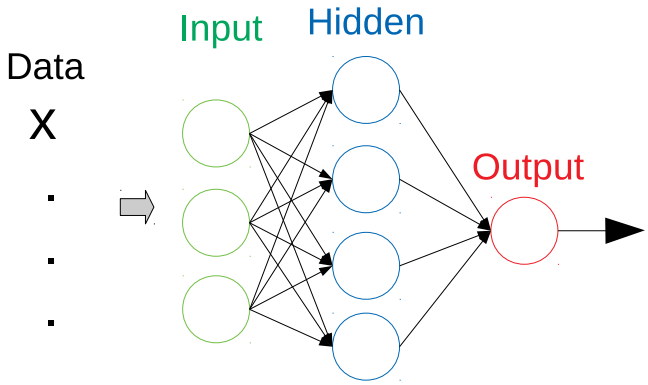


Figure 14: Multi-Layer Perceptron. By combining multiple layers of neurons, arbitrary non-linear functions can be approximated. The input layer converts the data into a higher feature representation. The hidden layer then again transforms the non-linear activation of those features and passes it to one or more output neurons.

To optimize (i.e. to train) these networks, a cost function C which assigns an error value to the output of the network $\hat{y}$ with respect to some ground truth target data y has to be assigned. E.g. for a binary classification $y \in [0, 1]$ and $\hat{y} \in [0, \ldots, 1]$

$$C(y, \hat{y}) = (y - \hat{y})^2 \tag{3}$$

The network is optimized at each time step t by iteratively changing the weights of all neurons, such that the global output error (i.e. the cost C) is minimized. For the output, the weight update can be derived as

$$w_{ij}(t+1) = w_{ij}(t) + \frac{\partial C}{\partial w_{ij}} \tag{4}$$

Backpropagation of the output error is used to calculate the gradients of the weights in previous layers with respect to the overall error.

With the rise of stronger computation power and the use of GPUs to calculate the network operations, those networks could be designed with more and more layers. This is referred to as the depth of the network explaining the name deep neural network.

CNNs as well as the MLPs are organized in layers, each consisting of a set of neurons that are connected to the same input. While in an MLP all neurons in one layer are connected to all outputs of the previous layer, in a CNN only a receptive field of n neurons is taken as input. This process is inspired by the response of a biological neuron to visual stimuli and mathematically similar to a convolution defining the name CNN.

The receptive field of a neuron (i.e. its input) is only a small local patch of the image or feature map of the previous layer. If we arrange and stack the neurons in a 2D shape like an image, each neuron in the next layer gets connected to a window around the respective position of the previous layer. In a convolution sense, the receptive field can then be interpreted as a filter of height h and width w, i.e. $h \times w$ weights. Figure 15 illustrates the striding convolution over a two dimensionally arranged CNN layer. By using the same weights for all neurons, this implements the shifting of a constant filter over the input data which is the same as the result of a discrete convolution of the input with this filter. By combining

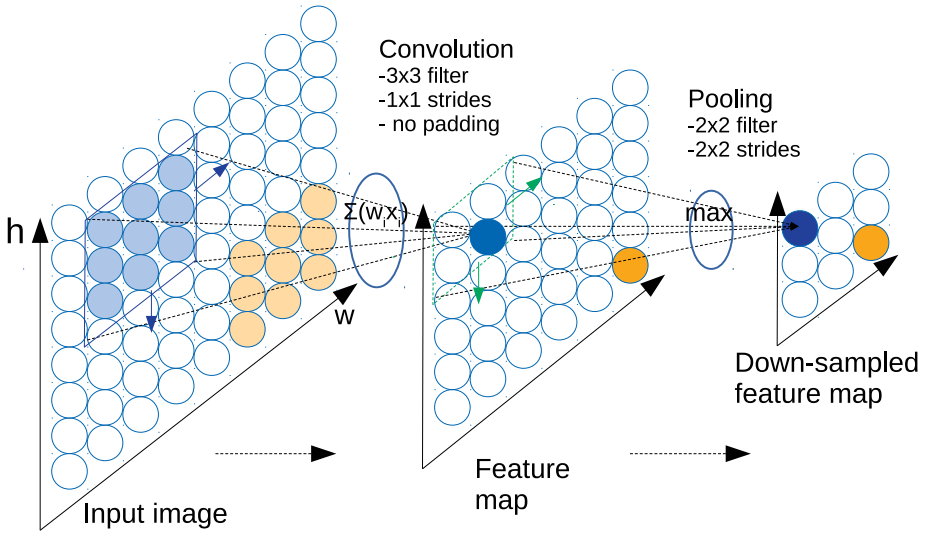


Figure 15: 2D Convolution and Pooling: The filter (blue rectangle) is slit over the input layer. A neuron (blue circle) is activated by the weighted sum all output values within its receptive field (the filter) of the input layer. The filter response is activated by the non-linear function. The resulting feature map is down-sampled, typically by taking the maximum over a window (green, dashed rectangle) to reduce spatial dimensionality. The step width (strides) is usually the same as the size of the filter in order to 'activate' each neuron only once.

the local filter application with maximum-pooling, the feature abstraction becomes more compacted with each layer and the spatial information vanishes. Because a lot of computer memory is saved with weight sharing, more neurons can be used

to implement multiple filters to recognize multiple features which makes CNNs a trainable multi-purpose feature extractor.

The fact that the filter weights are constant empowers the idea that what is a good local feature extractor in one location is also a good one over the whole input. Together with the pooling operation, this gives CNNs a spatial invariance to where certain features are recognized helping to operate on large images and detecting features regardless of their position.

In order to classify images, multiple convolutional layers are concatenated and after the last convolutional layer evaluated with one or more fully connected (FC) layers calculating class probabilities. This initiated the successful path for CNNs. With ongoing improvements of architecture, activation function, and pooling strategies, variants like LeNet [LBBH98] or AlexNet [KSH12] gained attention by winning multiple image recognition challenges like ImageNet[DDS+09]. Figure 16 shows a typical CNN architecture.

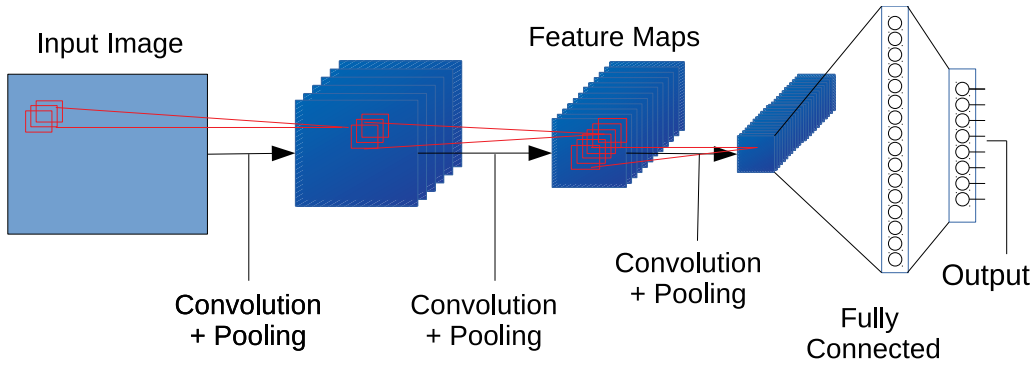


Figure 16: Simple CNN for Classification. The network contains multiple convolution stages. All feature maps at the deepest stage are connected to a FC layer. The output consists of C neurons yielding probability scores for C classes

Since then, CNNs have been improved by various enhancement techniques. For instance, batch normalization [IS15] normalizes the convolution filter responses such that their mean is zero and their standard deviation is one. This can enormously stabilize and speed up the training process because the following layers do not need to adapt to a constantly changing distribution of their inputs.

2.2.4 CNNs for Object and Landmark Detection

Object detection deals not only with 'what' is in an image but also 'where' it is. Applications comprise autonomous driving, robotics, face detection, tracking a ball in a video scene, and many more. Landmark detection is about finding keypoints

in an image. It is usually seen as part of another problem, e.g. matching facial keypoints to a shape model, identifying joints for a pose estimation, or finding measure points in medical images.

For object detection, the desired output usually is a bounding box around the object whereas landmark detection requires merely the center of an object or a keypoint location in the image. For both, classical machine learning and deep learning approaches do exist. While in the classical variants hand-engineered or statistical feature descriptors, e.g. histogram of gradients, are used to match keypoints or object shapes, deep learning algorithms can rely on the CNN based trainable feature detectors.

Popular deep learning algorithms that combine CNN feature extraction with object localization started with R-CNN (Regions with CNN Features) architectures by Girshick et al. [GDDM14]. For each image, a set of independent region proposals is generated heuristically and thereafter classified by a CNN. Additionally the bounding box defined by the proposal may be refined after prediction. Improvements like Fast R-CNN and Faster R-CNN integrate the region proposal into the trainable architecture and make the computation of bounding boxes faster and more efficient. The models Single Shot MultiBox Detector (SSD) [LAE+16] and You Only Look Once (YOLO)[RDGF16] aim at intelligently discretizing the bounding boxes and thus improving prediction speed to real-time but lacking localization accuracy.

Because in object detection, the bounding box is generally described by the coordinates of one corner and the width and height of the box, theoretically all object detection approaches can be transformed to only predict the coordinates which then represent the landmark. However, in those cases where only the location is of interest, often the spatial accuracy is more important than it is for object detection.

For landmark detection, a CNN can be used to predict the landmark locations in image coordinates. Multiple landmarks can be predicted individually with multiple CNNs or simultaneously with multiple output channels of a network. Like in object detection, region proposals can be used to deal with large images. To further refine the coordinate prediction, cascaded networks can be used. They consist of multiple CNNs, each taking a cropped region around the predicted location of the previous stage as input and thereby refining the predicted locations.

Sun et al. [SWT13] propose a method for face recognition to regress coordinates of multiple landmarks. They individually train multiple CNNs in multiple stages (therefore cascade) with each stage refining the landmark prediction. This can be further improved by calculating a global loss function over all the whole cascade and thus training all levels of the cascade together as shown by He et al. [HKZ+17].

Although this approach can achieve accurate localization performance, the prediction of a deeper cascade is only based on the cropped input of its predecessor. This makes it prone to errors made in the early stages of the cascade and hard to learn specific spatial relations and dependencies.

Another approach to predict landmarks is to discretize the possible locations in a grid, e.g. for each pixel. Each location contains a score value how likely it is that there is a landmark at this location. The numerical values can be visualized as image intensities often resembling a similarity to thermal imaging and therefore termed heatmaps. When predicting heatmaps, the peak locations with the highest intensities are identified as landmark locations or region proposals for further processing. E.g a sliding window classifier or random forest voting can be used to give out a classification score for each location and thus to generate a heatmaps.

Figure 17: Heatmap for human joint positions. The image is taken from Pfister et. al [PCZ15], visualizing a network prediction for human joint positions in video sequences. To distinguish the different joints they use different colors for each type instead of a temperature related color mapping.

However, in heatmap regression, a CNN predicts the heatmaps in one shot directly from the input data. Instead of a CNN with FC layers and numerical output, fully convolutional networks (FCNs) are used (see e.g. Thompson et al. [TGJ+15]). The last convolution stage of the CNN is taken as output and the network is trained to generate heatmap images. The predicted output images visualize a spatial probability score for each pixel of containing a landmark. The landmark locations can then be determined by finding the maximum activation in the heatmaps. Figure 17 shows an exemplary heatmap application for human joint detection.

However, often the heatmap cannot directly by used for further data processing but absolute positions have to be determined. This can be done by selecting the position with the highest intensity which can be difficult if the predicted heatmap is of general low quality or distorted. Since CNNs use pooling layers to reduce the

image resolution the heatmaps become coarse and are typically up-sampled and smoothed to achieve better localization.

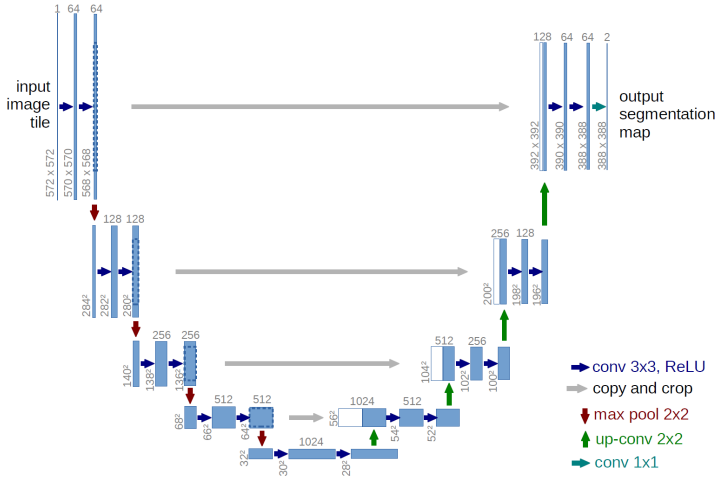


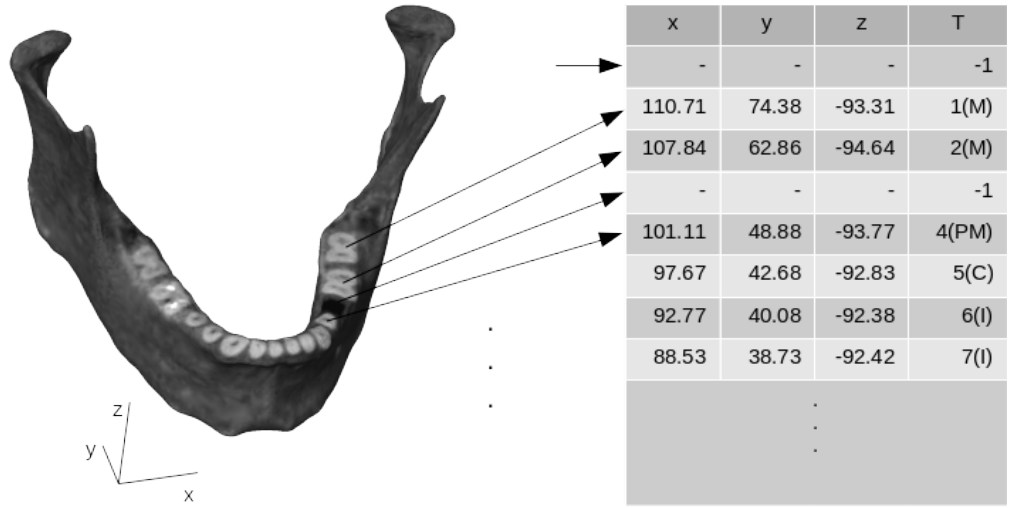
Figure 18: U-Net Architecture. The image from [RFB15] shows how the U-Net combines a down-convolution part with symmetric part where the resolution is restored and high level features are combined with less complex features.

An alternative to the up-sampling is use the U-Net architecture by Ronneberger et al. [RFB15] (see Figure 18). The U-Net was actually developed for image segmentation. It can produce pixel-wise accurate segmentation, and therefore also localization, by combining detected high-level features with the locally more accurate features of the lower convolution stages. This way it can learn to generate the heatmaps with respect to the full original image resolution.

Payer et al. [PSBU16] investigate different heatmap based CNN architectures for hand pose estimation and propose a new architecture that explicitly learns inter landmark dependencies. Their approach is to combine local appearance of landmarks (i.e. feature maps) with probability maps learned from spatial relation of the local appearances. To address the issue of losing spatial resolution due to the pooling operations, they investigate the application of large convolution filters without pooling layers and up-sampling and interpolation of the heatmaps. They also evaluate the performance of the U-Net and of a simple CNN with large filter sizes to cover spatial dependencies. Their architecture achieves better results for a reduced amount of training data. However, they only use simple up-sampling instead of the transpose convolution for all compared architectures including the U-Net thus, not fully exploiting the U-Net potential.

3 Detecting Teeth in CBCT

In this Chapter, we present an approach that is able to predict individual tooth numbers and locations of teeth within a CBCT volume. Figure 19 shows the localization of teeth on the segmented surface of a mandibular bone together with the tabular representation of tooth positions that we use to notate classes and locations in the 3D volume.



x	y	z	T
-	-	-	-1
110.71	74.38	-93.31	1(M)
107.84	62.86	-94.64	2(M)
-	-	-	-1
101.11	48.88	-93.77	4(PM)
97.67	42.68	-92.83	5(C)
92.77	40.08	-92.38	6(I)
88.53	38.73	-92.42	7(I)

Figure 19: Tooth detection using the reconstructed surface of the mandibular bone. We want to detect the positions of the tooth footprints on the surface and thereby position them in 3D. The tooth number and type (M=molar, PM=premolar, C=canine, I=incisor) are given in the rightmost column but also implicitly stored in the table by assigning the coordinates to fixed row entries, i.e. the first row for the lower left molar, etc.

In order to to locate and classify teeth in CBCT data, we divide the problem into three sub-steps. (1) We reduce the data dimension using an existing model based segmentation of the mandible (see [LZW+06]) and transform the tooth region into a planar surface that is further normalized onto an axis. A detailed description how the 2D images are generated follows in Section 3.1. (2) We train a CNN in a supervised manner with manually generated ground truth annotations. The generation of the ground truth is described in Section 3.2. To predict the individual tooth locations, we evaluate two different CNN based approaches. The first one is a down-convolution architecture with a FC output layer and will be described in Section 3.5. Section 3.6 introduces the second approach to detect landmarks via heatmap regression. (3) We use the prediction of the trained CNN and apply triangle correspondence between the 2D tooth region and the 3D segmentation to

find the tooth positions in the volumetric data. Since the 2D images are generated by intensities that are transported via the surface mesh, we can determine in which triangle each predicted position is and calculate its barycentric coordinates w.r.t to the triangle. The 3D position is now given by applying the barycentric coordinate to the corresponding unflattened triangle in the 3D segmentation of the dentition region.

The evaluation of the approach will be described in Chapter 4. Additionally to improve our approach, we explore image augmentation methods as will be explained in Section 3.3.

3.1 Dimension Reduction (3D to 2D)

In order to generate normalized images for the 2D CNNs, we apply surface transformations and sample the volumetric image data to transport the intensities along with the transformations. The following describes the transformations used to generate images of the dentition region.

3.1.1 Flattening the Dentition Surface

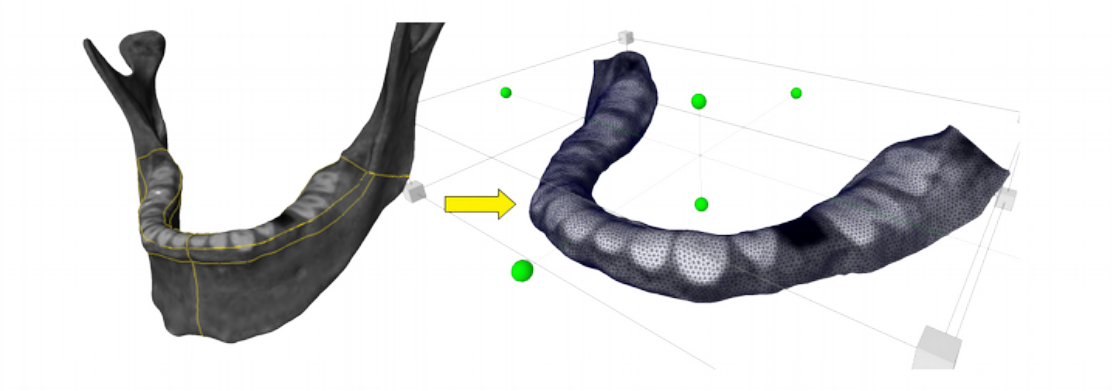


Figure 20: Extraction of the dentition surface. The original dentition patches of the SSM are extended by adding new patches defined by a new manually created path set (yellow paths, left). After the extraction, the surface is remeshed to smooth the contour of the surface and to have a moreover homogeneous mesh.

The surface reconstruction generated by fitting the SSM onto the CBCT scan produces a triangle mesh for the segmentation of the mandibular bone. From that, we select the patch containing the dentition region. This selection is manually defined on the SSM once and thus automatically given by the resulting adaptation

of the SSM fit to the image data. The extracted surface is a downwards opened and cutting through the teeth right above the bone surface as shown in Figure 20. In contrast to a simple axial slice, the 3D segmentation contains the actual curved surface of the bone and thus can cut through all teeth at the same relative height even if the mandible is not aligned with the horizontal plane. Moreover, not all teeth might be visible within one axial slice.

Since we do not want to lose the information that is represented by the curved surface, we cannot simply project the intensity values of the surface into an image. Instead, we flatten the surface with the Quasi-Isometric Surface Transformation (QIST) method by Wang et al. [WLT12] as explained in Section 2.2.2 and then capture an image of the planar dentition region. The Amira QIST module has to be initialized with a fixed triangle. We choose a fixed triangle of the meshed surface which lies in the center of the dental arch between the two central incisors. Again, this is consistent for new data due to the SSM correspondence. All triangles of the surface are now flattened into the plane that is defined by the fixed triangle. We further rotate the flattened surface into the xy-plane and reassure numerical stability by setting the z-coordinate to zero.

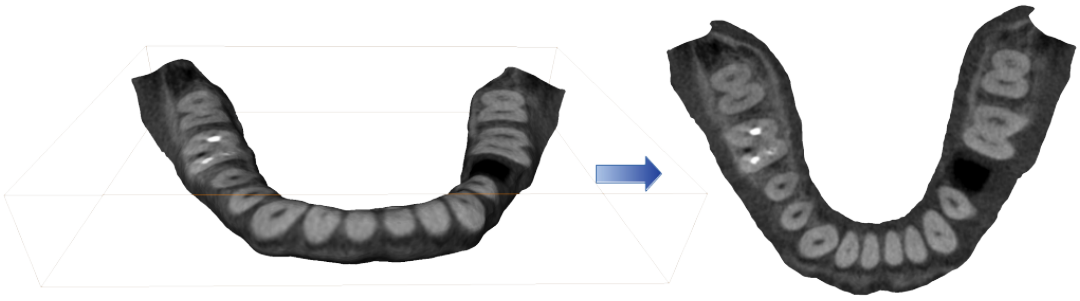


Figure 21: Flattening of the dentition surface. The QIST flattens the extracted surface while preserving the perceived appearance of the tooth contours. The image data is sampled at the 3D points of the triangle mesh and used for visualization of the planar surface.

Since the flattening operates on the triangle mesh it only provides the transformed mesh geometries but not an intensity mapping. For the intensity mapping, we probe and store the image values (intensities) at each vertex coordinate in the CBCT volume and use these values to visualize the flattened surfaces. To not lose information from the CBCT data, we refine both, the original and the flattened surface after the flattening but before sampling the image data. The refinement is

done by inserting a new vertex in the middle of each triangle edge until the mesh is much finer than the voxel size of the CBCT scan.

3.1.2 Normalisation (Bending)

Since an image of the U-shaped view of the bone surface would contain a lot of background pixels which would be fed to the neural network, we further optimize the input size to the network by normalizing the flattened arch into a rectangular form as shown in Figure 22 (right). To realize this normalization, the QIST theory and implementation were further extended within the ZIB by Felix Ambellan to realize a 2D bending operation. In order to bend up the dental arch, we define piecewise rotation fields that rotate a segment as in Figure 22 (left) from the curved arch onto a straight line. The rotation is that of the direction vector following the dental arch for each segment towards a fixed axis. Since the orientation of the head is fixed and reliable along the latitudinal direction, we simply use the x-axis for the normalization.

In order to calculate the rotation fields, we calculate multiple line segments along the center line of the dental arch. However, to do so we first need to determine this center line. We construct two paths as linear interpolations between vertices of the mesh. One through the vertices of the inner (labial) boundary of the SSM's tooth patch and one through the vertices of the outer (bukkal) path. Since the paths have a different length and a different number of vertices that are not necessarily equidistant, we define an explicit number of points that we use to re-sample and interpolate both paths. Now, we can take a pair of points, one of the inner and one of the outer path and then determine their mean. Executed for all points, this yields a middle path which approximately resembles the center line of the dentition area of the mandible. We determine the two closest points on the path for all triangles. Then, the desired rotation for those triangles is the rotation required to align the segment defined by the two points with the x-axis. Given that, again the transformation is done while imposing minimal distortion on the triangles.

3.1.3 Sampling Training Images

For the flattened and bent-up surface, we compute the bounding box which will be the frame of the image. Since the bent-up mandible is relatively long compared to its thickness, we widen the height of the image to fit an aspect ratio of 4:1. Although the image could have lesser height and still fit the whole mandibular surface, we select a slightly increased height, s.t. the image height does not vanish when the pooling of the CNN is applied. To sample the pixel intensities, we

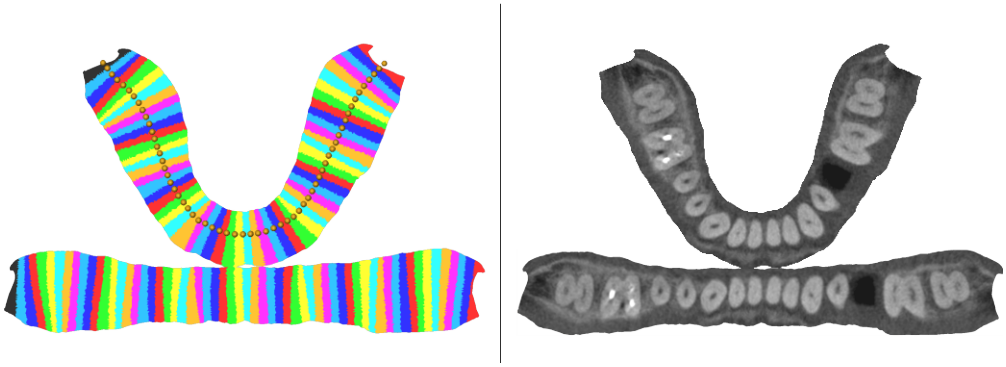


Figure 22: Unbending of planar surface. Left: All triangles of a colored stripe are assigned to the same line segment defined by their two closest landmarks. The QIST bends the surface by rotating each segment onto the x-axis. Right: Mapping of the image data. The tooth shapes are rotated but the local appearance is well preserved.

generate a grid of 1024:256 pixels and sample the center position of each pixel. For each position, we take the intensity value of the closest vertex of the surface mesh.

Although the image data in CBCT can not directly be used for assessment of bone density and quality, they can still be roughly interpreted as Hounsfield units (HU) (see [PJSM15]). We select a range from -1000 to 5000 which may be wider than actually needed to visualize the teeth and surrounding tissues. This range still keeps the break through interfaces of the teeth clearly visible while only thresholding high intensities of metallic implants. This way, we leave it open for the CNN to also draw information from the higher and lower HU values. However, to not lose numerical accuracy in the range of the intensity values of the teeth, we save the images with a high bit depth by normalizing the intensities to a 0-1 range as 32-bit floating point values in TIF format. This results in the BTIs (Break Through Image) we use to train a CNN.

3.1.4 Artificial Panoramic Radiograph

As addition to the image of the dentition interface, we use the APR, an artificially synthesized x-ray image visualizing a panoramic view of the teeth. This view was generated and kindly provided to us by 1000Shapes GmbH. Figure 23 shows the curved surface and the generated image for an example case. The image is generated by taking a ruled surface that stands perpendicular to the surface of dentition patch of the SSM. The x-ray is simulated by accumulating the voxel values in a certain thickness along the surface normal (integral projection). The

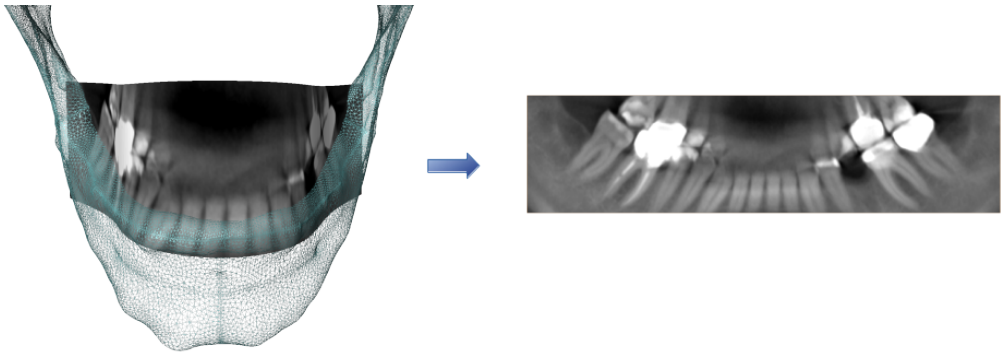


Figure 23: Artificial panoramic radiograph. A ruled surface is created that is placed at the center line of the SSM dentition patch and perpendicular to a plane fit through this patch (left). The x-ray is simulated by sampling the sum of the intensities along the surface normal. Then, the image is generated by unrolling the ruled surface (right).

ruled surface is flattened and thus normalized into a plane to generate the 2D image. This view can be beneficial if the the surface generated by the SSM adaption is not optimal an the break though interface of the teeth are not easy to identify because of metal copings or other artifacts. In this case, the good imaging of the dental roots in the APR can help to recognize the individual teeth (again see Fig. 23).

3.2 Generating Tooth Landmarks for Supervised Learning

Since we do not have existing ground truth data about the location of the individual teeth, it was part of this work to generate the tooth annotation in the CBCT data. We developed an interactive tool in the ZIB-Amira to manually define the position and class of the teeth. Despite the lack of expert knowledge, we were able to confidently identify both location and tooth class according to the numbering scheme in Figure 3.

3.2.1 Visualization

We define the tooth positions on the 3D reconstructed surface from the SSM segmentation of the mandible. In the 3D view,the mandible can be freely rotated and moved which provides a good view onto the dentition. In order to assess the relation between teeth and to correctly identify missing teeth, the curved slice along the dental arch is helpful. We can evaluate the tooth orientation and conclude on pathological conditions. Additionally, Amira provides the option to split the

viewer. We use this to simultaneously display the different views, i.e. curved panoramic slice and the APR, and the flattened dentition interface in U-shape and the BTI. A labeling scenario in which all possible views are displayed is shown in Figure 24.

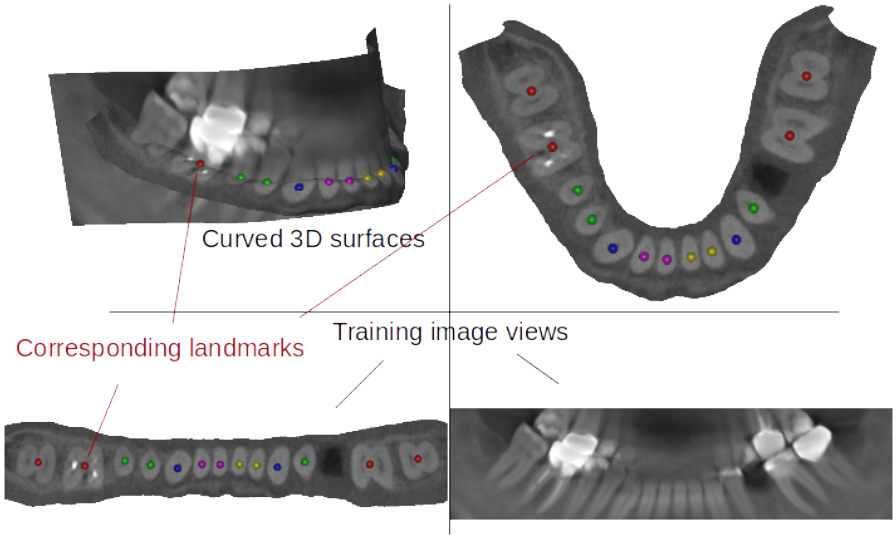


Figure 24: Multiple views of the landmark annotation tool. Upper left: The 3D surface of the dentition region is displayed together with the curved plane for the panoramic radiograph. Upper right: The flattened bone surface is viewed from top-down. Lower left: the normalized bone surface. Lower right: The unrolled panoramic x-ray.

3.2.2 Notation

The teeth are marked by clicking into the tooth islands on the surface of 3D mandible which places a sphere shaped landmark on the picked position. The label, which is the tooth number, can be chosen from an enumeration menu but is also automatically increased after each annotation. Furthermore, hotkeys for increasing, decreasing the tooth number, and for deleting the currently selected marker are provided. All this helps to ease the labeling process and to speed up annotation time. Theoretically, it would be sufficient to annotate the landmarks in the flattened tooth patch to evaluate the prediction but for the benefit of re-usability and visualization, it is an advantage to know the position in all modalities. Therefore, we chose to annotate the landmarks on the 3D surface and to calculate

all corresponding positions. When the examiner places the landmarks in 3D, the positions on the flattened surfaces and in the generated 2D images are calculated and displayed simultaneously.

Table 3: Tooth Location Table. P(T) shows if a tooth is present (1=tooth exists, 0=tooth missing). Coordinates are given in 3D space and as normalized coordinates in the image stripe (2D).

c	p_T	$3Dx$	$3Dy$	$3Dz$	$2Dx$	$2Dy$
0	0	-	-	-	-	-
1	1	110.71	74.38	-93.31	0.885	0.535
2	1	107.83	62.86	-94.64	0.788	0.491
3	0	-	-	-	-	-
4	1	101.10	48.88	-93.77	0.658	0.508
5	1	97.66	42.68	-92.83	0.603	0.467
⋮	⋮		⋮		⋮	
14	1	56.92	71.06	-92.46	0.122	0.506
15	0	-	-	-	-	-

We define a notation of tooth classes and locations with which we store the tooth positions for 3D and 2D in a CSV file. In this scenario, the class equals the row number in the table. For computational purposes, we enumerate the 16 teeth of the mandible from zero to 15. Table 3 shows the representation of a dental chart table to store tooth positions and classes. The locations correspond to the landmarks shown in Figure 24 for the 3D mandible (upper left) and the 2D stripe (lower left).

3.2.3 Reduced Classification

Defining the unique tooth number may sometimes be almost impossible, especially if one of the neighboring teeth of the same tooth type is missing and the remaining teeth have spread to equally close the gap (see Fig. 25). For the cases where we could not explicitly number the teeth but define the tooth type, we derive the class from the type of the tooth and from the sector in that it is located (8 classes).

Now, it is not given in which row of the table the marker should be entered. Since we want to map the output channels of a CNN to the table entries it should be deterministic for every detected tooth into which row it will be sorted. The order can be made explicit again if we sort the entries within one class. Each class is assigned one up to three entries, i.e. the first three entries for the lower left molars, then two for premolars, one canine, two incisor, etc. Within each class, we sort the teeth by their x-coordinate in the image. The result is shown in Table 4. Note,

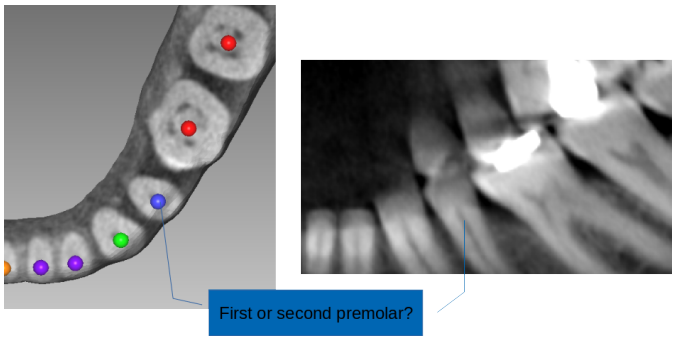


Figure 25: Ambiguous tooth enumeration. The example case only contains one lower left premolar (of two possible). Neither from the dentition image nor from the panoramic x-ray, the tooth numbering becomes evident. Nevertheless, it can still be classified as lower left premolar.

how the two lower left molar entries moved to the top. Even, if we did not know that they are first and second molar, the table would be the same.

3.3 Image Augmentation

Usually, for deep learning approaches, thousands of images are used as training samples. If these data is not existent, image augmentation can improve the performance of a neural network. Data augmentation can be used to model expected variations that are not present or underrepresented in the training data. For medical images, that can be changes in anatomy or varying image properties caused by the configuration of the machinery.

Commonly, image augmentation is used to empower the model's robustness to correctly process rotated or translated views that might theoretically occur. In our case, like in many medical scenarios, the orientation of the objects of interest is rigid in a certain sense. Patients are fixated or have limited movement possibilities within e.g a CT or MRI tube, thus only small translations and rotations are possible. However, noise and intensities depend on the CBCT scanner and its configuration and must be assumed variable.

Because we generate the images based on the SSM fit, freedom in positioning and orientation is further reduced by aligning the image frames to the shape and orientation of the generated surfaces. This is guaranteed also for new data such that rotation or shifting as image augmentation are not necessarily required. However, flipping the sides of the mandible an slight rotation could still be beneficial because they also cause the local appearance of the individual teeth to be changed relatively to the convolution filters.

Table 4: Tooth location table with 8 classes. Entries are sorted by the x value. Missing tooth entries are moved to the last row of their class.

Tooth Type	c	x	y
LL molar	0	0.885	0.535
		0.788	0.491
		-	-
LL premolar	1	0.658	0.508
		0.	0.
LL canine	2	0.603	0.467
	⋮	⋮	
LR molar	7	0.216	0.477
		0.122	0.506
		-	-

3.3.1 Geometric Augmentation

We augment the training data by mirroring all images on the image y-axes such that the sides of the mandible are swapped. This is done for the flattened tooth row as well as for the artificial panoramic radiograph. The flipping of the sides is reasonable because we can assume variations in the tooth shape to be independent on the sides of the mandible. Flipping the tooth row upside down is less plausible, since typical orientations of the tooth shapes would be inverted.

A rotation of the shape of a single tooth in the images would be desirable but is not easily possible because we do not have the segmentation of this shape and rotating an estimated bounding box would heavily introduce artifacts and destroy the general appearance of the image. Instead, we rotate the whole images for two image-wise randomly selected angles between -5 and 5 degree. This rotation is not consistent with any sort of variation in anatomy or in the image generation process but changes the local appearances of the tooth interfaces and thus should make the prediction more robust against rotations of individual teeth. The same rotation is applied to the panoramic x-ray image to also vary the local appearance of the teeth.

3.3.2 Artificial Tooth Loss

A more important augmentation is to cope the combinatorial challenge, that theoretically any tooth can be missing. To address this issue, we artificially simulate missing teeth within the complete dental arch. An example image with artificial

tooth removal and rotation is shown in Figure 26. For each case, how many and which teeth have to be removed is defined by a randomly generated binary series.

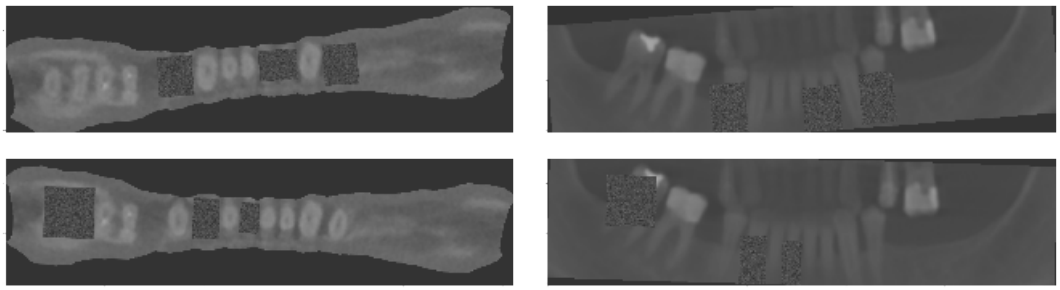


Figure 26: Artificial removal and rotation. We simulate a missing tooth by placing a dark rectangle over the tooth position. Furthermore, both, the BTI and APR are slightly rotated after the placement of the rectangles.

At each position of the designated tooth, we place a rectangle with its center at the tooth location. In order to occlude the image features of the tooth, we use different rectangle sizes for the different tooth types and the two different images. We define an approximate edge length of the rectangle for a molar as 10 percent of the image width. Depending on tooth type ([molar, premolar, canine, incisor]) the width is determined by multiplying the edge length with [1,0.8,0.9,0.7] and the height by multiplying with [1,0.8,0.8,0.6], respectively. We use the same factors for BTI and for APR but further limit the right and left width of the rectangle by half the distance to the landmarks fo the next neighboring tooth. Although this is not close to a perfect method, the bounding boxes seem appropriate for all our training data and the augmentation did not require manual interaction. The filling of the rectangle is generated by taking the average intensity of the image and further distorting it with Gaussian noise. The position in the APR is estimated from the break though position by assuming a parabolic curvature of the tooth row in the panoramic image. We visually determine the parabolic formula $y = 0.85x^2 + 0.35$ to be a good fit.

Although consecutive effects like tooth movement cannot be modeled this way, this augmentation method improves robustness of the prediction against new data that might contain combinations of missing teeth that do not exist in the training data.

3.3.3 Application to Training Data

Note, that for all the above mentioned augmentation methods, also the locations of the ground truth location table have to be changed. We first flip the images, then apply the artificial tooth removal, and apply the rotation last such that the occlusion rectangles are rotated together with the image. Because of implementation reasons, these augmentations are calculated initially once before the training process and appended to the training images. The effective number of pre-rendered training images is then the initial number multiplied by 2 for the mirroring, multiplied by the number of tooth removal iterations, and then multiplied by the number of used rotations. We use one iteration for tooth removal and two rotations which results in 944 images with explicitly numbered teeth and 1264 images that can be used in the 8 class scenario.

3.3.4 Intensity Variation

Since intensities can vary due to different bone density or the CBCT configuration, we further use gamma correction as a non-linear image augmentation method. The scaling is defined as: $I_{out} = I_{in}^{\gamma}$. A commonly used normalization constant (often A) can be neglected because we normalized the image to a 0-1 range for all data processing beforehand. The γ parameter is varied uniformly between 0.75 and 1.25. Figure 27 shows transformed images for the boundary values. Since this method

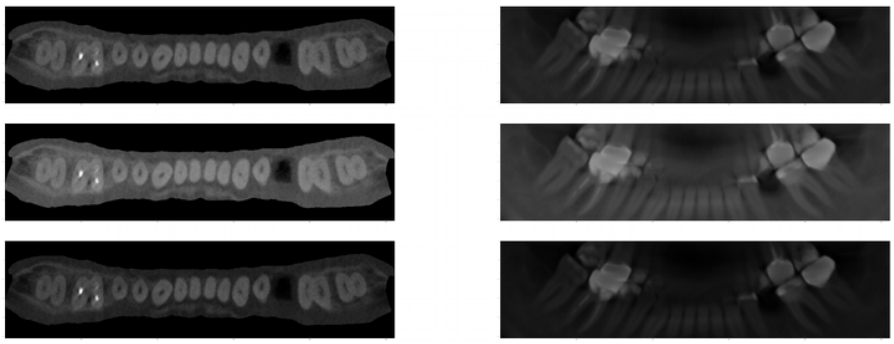


Figure 27: Image Augmentation: Gamma Shift. From top to bottom: Original, gamma=0.75, gamma=1.25

does not require a change of the tooth coordinate in the ground truth data, we can use the on-line image augmentation provided by the Tensorflow-learn framework that applies the modification for every training step again using a parallel process.

3.4 Common CNN Approach

In order to estimate the tooth coordinates, we investigate two CNNs based approaches to predict a tooth existence probability p and (x, y) coordinates in the 2D images.

In the numeric regression approach (see Section 3.5), we predict x, y and p directly from the CNN output channels. Secondly we use a heatmap regression approach (see Section 3.6) to predict location probability maps from which we read the coordinates afterwards.

Both approaches share the same convolutional building block structure that is shown in Figure 28. This sequence is the same as used in the original U-Net ([RFB15]), To extract more powerful and highly non-linear features, it uses two convolutions and ReLU activation combinations followed by dropout. We only add batch normalization (see Section 2.2.3) for better training convergence and generalization.

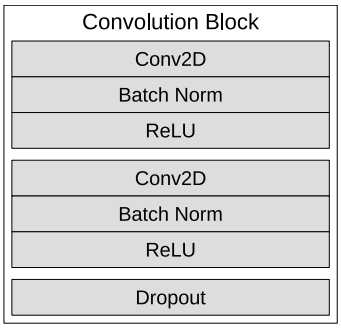


Figure 28: Convolution Block Structure. As proposed in the U-Net architecture, we use two sequential 2D convolutions with ReLU activation followed by dropout. Additionally we use batch normalization restrict the input distribution to the next convolution and thus to stabilize the CNN training.

3.5 Numeric Regression

The first scenario is to regress landmark coordinates directly from the CNN output. We define a CNN architecture and a loss function that is used to train the network.

3.5.1 Loss Function

Since the last layer of the CNN is used to generate a structured output matching to Table 3, we can compare each generated output to its corresponding table entry

of the ground truth. The loss function evaluates the overall difference between the CNN prediction and the ground truth. For the tooth detection, the deviation from the true location as well as miss-prediction for the existence of teeth have to be penalized. From that follows, that besides a distance loss a probabilistic aspect that regards the existence of teeth in the dentition needs to be evaluated. In the case that teeth are missing, the table has empty entries with no defined location. Thus, a dedicated loss for empty entries or teeth that were not detected by the CNN must be calculated.

For the following, the network output for one landmark (tooth) is denoted as $\hat{z}_t$ consisting of $\hat{x}_t, \hat{y}_t \in \mathbb{R}$ and depending on the scenario $\hat{p}_t \in \mathbb{R}$ corresponding to the ground truth entries as in Table 3 denoted as x_t, y_t and p_t. Furthermore the location can be written as $\hat{l}_t = (\hat{x}_t, \hat{y}_t)$ To address the issue how to deal with missing landmarks, in general, there are three possibilities.

1. Setting the x,y entry for missing teeth outside the image, e.g. to the maximum of the data type. Then, the loss is simply the average distance over all $(\hat{x}_t, \hat{y}_t)$ entries. Different numerical values are possible, as long as they are outside the image.

2. Interpreting the existence flag as third coordinate and regressing a 3D function. Each entry is interpreted as one location l_t consisting of all three variables, i.e. $l_z = z_t = (x, y, p)$.

3. Combining a distance loss evaluated only for positively labeled teeth with a probabilistic loss term. This is more complex and will be handled in this section after explaining the distance loss.

Distance Loss For all cases, the distance is the displacement of the predicted location from the ground truth position, evaluated as the distance in image coordinates. We define the distance loss as

$$L_d(z_t, \hat{z}_t) = ||l_t - \hat{l}_t||_2 + ||l_t - \hat{l}_t||_1 \tag{5}$$

$$L_d(z_t, \hat{z}_t) = (x_t - \hat{x}_t)^2 + (y_t - \hat{y}_t)^2 + \sqrt{(x_t - \hat{x}_t)^2} + \sqrt{(y_t - \hat{y}_t)^2}. \tag{6}$$

Equation 6 combines the absolute distance with the quadratic distance. This is beneficial for gradient based optimization, since the quadratic function has significantly greater gradients for large distances and the gradient of the absolute distance does not vanish for small distances. It can often be seen that the network output for the location is activated by a sigmoid function that restricts the coordinates to be in a [0,1] range. Although this would be desired for the tooth coordinates in the normalized image, we found that the training converges faster

without a sigmoid activation. For the evaluation, locations that are outside the image are either clamped to [0,1] or dismissed and interpreted as missing teeth, depending on how missing teeth are encoded (1-3)

Combined Loss For case (3) we add a probabilistic loss (L_p) as the difference between the predicted tooth existence and the ground truth. Then, the loss is defined as

$$L(z_t, \hat{z}_t) = L_p(p_t, \hat{p}_t) + L'_d(z_t, \hat{z}_t) \tag{7}$$

$$L(z_t, \hat{z}_t) = L_p(p_t, \hat{p}_t) + k \underbrace{\frac{p_t \hat{p}_t}{p_t \hat{p}_t + \varepsilon}}_{\text{existence factor}} L_d(l_t, \hat{l}_t). \tag{8}$$

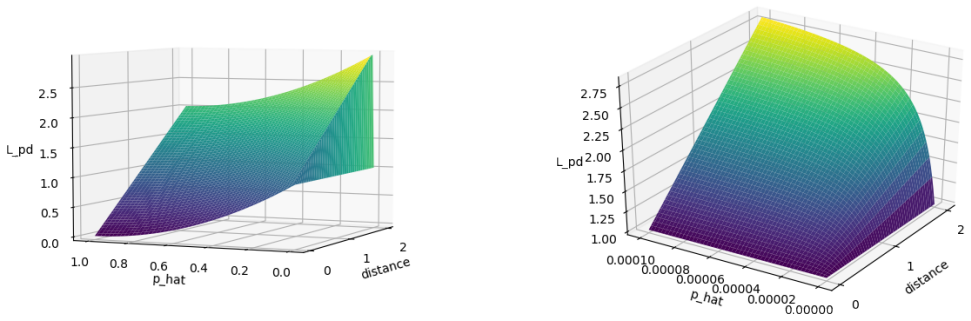
where k can be used as a weighting factor between both losses. Since the class is indicated by the row, a binary classification loss can be used. Therefore, L_p is be defined by the quadratic difference of the first column $L_p(p, \hat{p}) = (p_t - \hat{p}_t)^2$.

However, if we only evaluate the term from Equation 6, the network would learn to regress uncertain existence predictions towards the numerical value that is entered as ground truth location for missing teeth. The existence factor in Equation 8 excludes those values in the distance loss. The distance term is multiplied with the ground truth flag whether the tooth is present or not such that missing teeth that are falsely predicted as existing do not contribute to the distance loss. Analogously, the term is multiplied with the predicted probability $\hat{p}_t$ and divided by $p_t \hat{p}_t$ again to avoid scaling of the distance and inverse dependence of the distance from the probability. By adding a small constant $\varepsilon = 1 \cdot 10^{-8}$, we avoid division by zero. The distance loss becomes zero if either p_t or $\hat{p}_t$ is zero.

Since the ground truth probability is only a binary decision, we can do a case distinction. For the following, we write $L_d(l_t, \hat{l}_t) = d_t$.

Case 1, $p_t = 1$:

$$\begin{aligned} L(z_t, \hat{z}_t) &= L'_d(z_t, \hat{z}_t) + k \cdot L_p(1, \hat{p}_t) && (9)\\ &= \frac{\hat{p}_t}{\hat{p}_{n,t} + \varepsilon} d_t + k(1 - \hat{p}_t)^2 \\ \text{if } \hat{p}_t >> \varepsilon & \\ &\approx d_t + k(1 - \hat{p}_t)^2 && (10)\\ \text{if } \hat{p}_t \approx \varepsilon & \\ &\approx 0.5||l_t - \hat{l}_t|| + k(1 - \hat{p}_t)^2 && (11)\\ \text{if } \hat{p}_t << \varepsilon & \\ &\approx k(1 - \hat{p}_t)^2 && (12) \end{aligned}$$



(a) Loss surface for $0 < \hat{p}_t < 1$.

(b) Close up on $0 < \hat{p}_t < 0.001$.

Figure 29: Plot of the conditional distance loss (L'_d) for $p_t = 1$. The loss function smoothly descends with increasing $\hat{p}_t$ and decreasing distance. Only if the distance is high and $\hat{p}_t \approx \varepsilon$ the loss decreases again.

Case 2, $p_t = 0$:

$$L(z_t, \hat{z}_t) = L_d(z_t, \hat{z}_t) + k \cdot L_p(0, \hat{p}_t) \tag{13}$$

$$= \frac{0 \cdot \hat{p}_t}{0 \cdot \hat{p}_n + \varepsilon} d_n + k(-\hat{p}_t)^2$$

$$= k\hat{p}_t^2 \tag{14}$$

In Equation 11 and 14 the distance is not dependent of the existence term or vice versa, therefore the two different loss types do not scale each other. Only for a complete false negative ($p_t = 1$ and $\hat{p} \to 0$), the distance becomes neglected as visualized in Figure 29. Thus, a stochastic gradient descent optimization should lead to the minimum at $\hat{p}_t = 1$ and $L_d = 0$. This coincides with the logic reasoning that if a tooth is not recognized at all, no distance can be calculated. Numerically, the loss for a true positive (Eq. 11) is greater than for a true negative (Eq. 14), for equal deviation of $\hat{p}_t$. However, the evaluation in Section 4.3 shows that there was no predominance of false negative predictions.

For the weighting factor k in Equation 8 we can reason about the importance of correct prediction on the one hand and localization accuracy on the other. A prediction is only accurate if the existence is correctly predicted and the location lies at least within the shape of the tooth contour. Therefore, a prediction outside the tooth shape is equal to a false $\hat{p}$ value. We measure the radius of an example tooth shape in the normalized image of the tooth row as $r = 0.05$ in normalized image coordinates. A prediction, displaced this far, could be seen as a complete

false prediction and equal to $L_p(0,1) = 1$. Therefore, it follows from $1 = k \cdot 0.05$ that $k = 20$. This is a roughly estimated approximation, but we found that the setting of k does not have great influence on the prediction and $k = 20$ is a good weighting factor.

3.5.2 Reduced Classification

Additional considerations have to be taken if classifying the teeth into less than 16 classes, i.e. if one class can have more than one instance. If comparing the prediction and ground truth for e.g. class 0 (the 3 lower left molars) the order of those entries is not important for classification but it is for the pairwise comparison of the locations. We evaluate two options. 1) We use the existing loss and establish a deterministic order of the entries. Since for the loss function, each pair of ground truth and network prediction entry is compared individually, the ground truth data must be ordered consistently as defined in Table 4. This means, however, that the reordering has to be learned by the network, too. 2) We calculate all permutations of the ground truth entries and only compare the best fit of prediction and ground truth.

For 8-classes at maximum three locations (for left and right molars) must be permuted. For each class, we calculate distances from the ground truth to all permutations of the predicted channels

$$L_{n,c} = \min_{\rho=0}^{P} L_d(\mathbf{z}_{n,c} - \hat{\mathbf{z}}_{n,c,\rho}) + L_p(\mathbf{z}_{n,c} - \hat{\mathbf{z}}_{n,c,\rho}), \tag{15}$$

$$\tag{16}$$

where $\mathbf{z}_c$ and $\hat{\mathbf{z}}_c$ are the ground truth and predictions of all p, x, y values for all table rows t that belong to class c. P is the number of possible permutations for the entries within the class and $\mathbf{z}_{c,\rho}$ is then the ρ'th permutation of network output vector ($\mathbf{z}$).

3.5.3 Network Architecture

We implemented a simple downwards-convolutional CNN architecture as shown in Figure 30. Since the optimization task is not only classification of an image but also a localization of multiple landmarks, the output consists of 3×16 neurons prediction $\hat{p}$ and $\hat{x}$ and $\hat{y}$ coordinates for all possible entries.

Since the pooling operations lead to a loss of spatial information that would be crucial for correct landmark prediction, we evaluated different the use of fewer pooling stages but instead using convolution filters of larger size. The theory is

that the filters at the deepest convolution stage should be large enough to detect the shape we want to recognize.

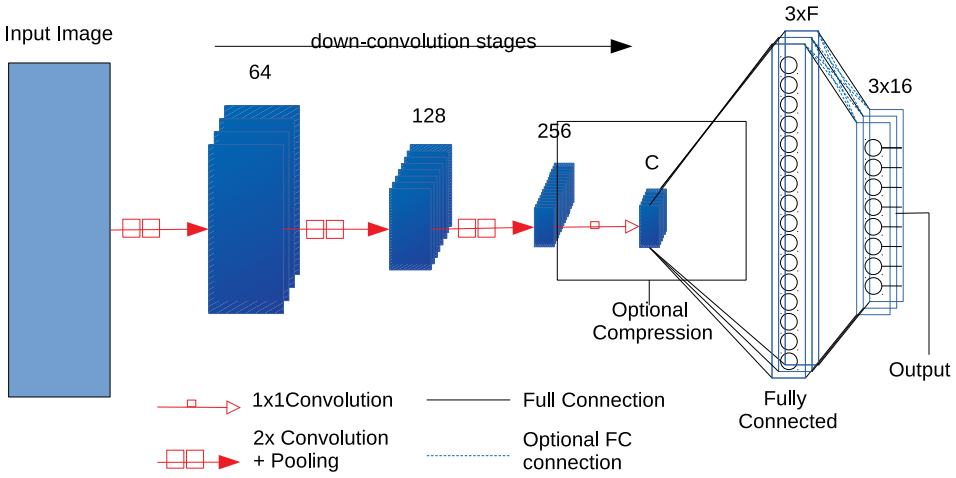


Figure 30: Tooth Detector CNN. The depth of the network is defined by the number of convolution stages S. The number of filters and thus channels is doubled at every stage while the image size is halved by max-pooling. After the last down-convolution stage, a single convolution with a $1 \times 1 \times 16$ filter is inserted. The number of channels or units is annotated above the image or FC layers, respectively. C denotes the number of classes used and F the size of one fully connected group for $\hat{p}_t, \hat{x}, \hat{y}$

In the FC layers, we can either connect the output neurons to all neurons in the previous layer or we can split the network to connect x, y, p to separate FC layers. This is indicated by the blue dashed lines in Figure 30. We connect the first FC layer to the complete feature map. If we then consider each variable separately, the output for each variable can be connected to a individual fully connected hidden layer. Therefore, there are three stacked FC layers in the figure. It can be reasoned by the assumption that the calculation especially of the coordinates ($\hat{x}$ or $\hat{y}$) must not be connected to the calculation of $\hat{p}$. However, although the separation reduces the number of weights, there was no notable speed up in the training but the performance was slightly worse. Although this might have an impact in some scenarios, we did not pursue this investigation any further and chose to connect all FC units due to slightly better performance in a first experiment.

As we will show in Section 4.3, we found that for our case it is beneficial to reduce the number of feature maps before those are connected to the fully connected layer. This is done by applying 16 1x1 convolution filters. Fewer feature maps reduce the number of connections and thus weights for each of the neurons in the FC layer

which we observed to speed up training time and to prevent over-fitting of the training data.

We also evaluated larger networks as the VGG architecture [SZ14] and adapted the output layer to our coordinate regression. Since the VGG architecture was designed for training on a larger number and variety of training images as in e.g. the image-net challenges [DDS+09], it was more difficult to train, more prone to over-fitting, and had worse performance than the shallower networks.

3.6 Heatmap Regression

Another way to address the localization problem is by predicting heatmaps. Instead of generating numerical coordinates in the image, a network is trained to produce heatmap images as described in Section 2.2.4. In the case of object or landmark localization, the heatmaps display the probability over the image space of how likely it is that the pixel contains a landmark.

There are two main advantages of using heatmaps instead of direct coordinate regression. (1) Heatmaps provide a solution for multiple or missing instances. The number of object instances per class does not have to be encoded in the network architecture anymore. Any amount of keypoints can be represented by providing a ground truth heatmap with its peaks representing the probability of an object or keypoint being at this location. (2) The network actually learns an image to image mapping that is potentially less complex but more intuitive because we know that the spatial order of landmarks directly corresponds to the input image.

Despite this, there are also two downsides to this approach. (1) The step of extracting the coordinates is not part of the CNN and cannot be optimized together with the backpropagation of the network. This leads to the challenge of finding a robust procedure that can best determine the predicted location within the heatmaps and also can cope with not perfectly smooth predictions. (2) Convolutions are inherently translation invariant. One advantage of the fully connected layers is that they evaluate all features of the previous feature map at once. This enables them to learn spatial dependencies as e.g. a fixed distance between two joints because of the bone length. In a FCN, the trained filters are convolved over the image and feature activations are calculated independently at each position. In order to still represent the spatial constraints in the heatmap approach, we can try to encode prior knowledge into the network design.

3.6.1 Heatmap Generation and Loss

In order to create the heatmaps, we generate empty (zero valued) images $h_{n,c}$ for every class $c \in (1, \ldots, C)$ and all cases $n \in (1, \ldots, N)$, where N is the number of cases and C the granularity used for the classification problem. Further on, we only regard the heatmaps for a single case and omit the n subscript index. For

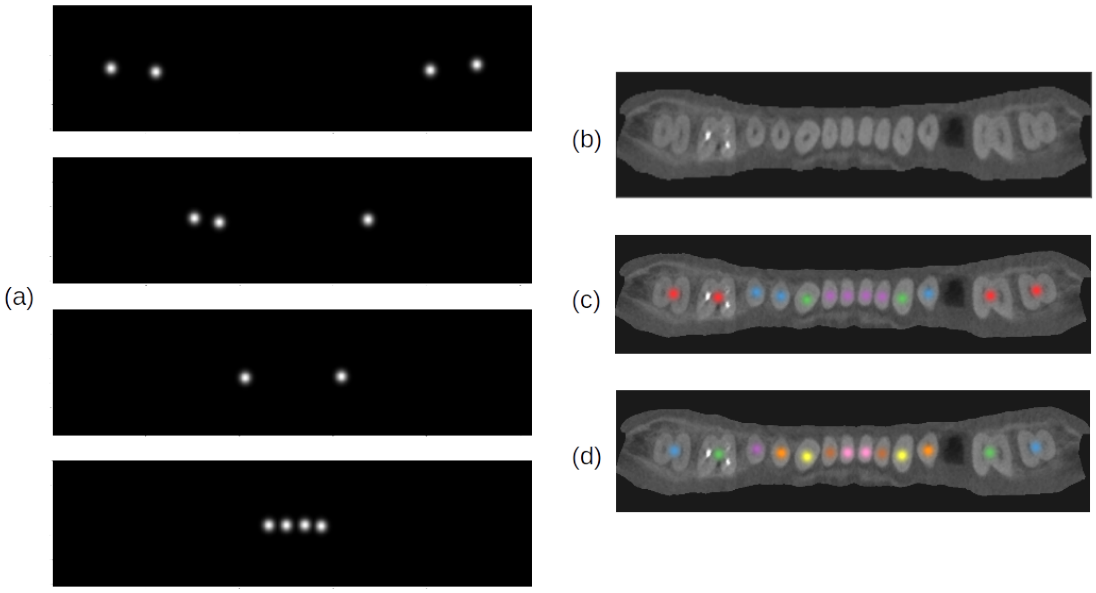


Figure 31: Example Heatmap. Displayed are the heatmaps for a 4 class scenario (a) with positions of the molars, premolars, canines, and incisors (top to bottom). We combine this with the original BTI (b) to create a colored visualization. The corresponding image for 4 classes is shown in (c). Image (d) shows the colored plot for 16 -class heatmaps. The same color is used for corresponding teeth on the left and on the right since the side can be distinguished easily.

every location $l_t = (x_t, y_t)$, we place a Gaussian peak in h_c, where c denotes the class as defined in Table 3 or Table 4.

The Gaussian is defined as

$$G_t(x, y) = exp(-\frac{(x_t - x)^2 + (y_t - y)^2}{2\pi\sigma^2}) \tag{17}$$

with (x, y) as the normalized image coordinates and (x_t, y_t) as the tooth location. Since the heatmaps have a discrete resolution, each pixel gets assigned the distance from the tooth location to the pixel center.

For the case that all teeth are enumerated ($C = 16$), each heatmap contains exactly one location. If $C < 16$, multiple peaks may be placed next to each other resulting

in a possible overlap. Instead of adding the values, we take the pixel wise maximum of the superposition, since the summation would lead to local peaks even where flat areas of the regions overlap.

The heatmap prediction is commonly optimized by minimizing the pixel-wise softmax cross-entropy. To apply the cross-entropy, the feature vector at each pixel along the class dimension has to be a valid probability density. Equation 18 extends the ground truth data, s.t. the pixels of the background layer B_{ij} get assigned the probability of not belonging to a location peak:

$$B_{i,j} = 1 - \sum^{C} h_{i,j}. \tag{18}$$

If two location peaks overlap, the sum over C can be greater than 1, hence we normalize each pixel location in a second step:

$$B_{i,j} = \frac{B_{i,j}}{\sum^{C+1} B_{i,j}}. \tag{19}$$

The network output has the dimension of $H \times W \times (C+1)$ and the values are activated by the softmax function, applied to each pixel along the class dimension. For one pixel, the softmax activation is defined as

$$\hat{h} = \frac{exp(\hat{z})}{\sum_{c=0}^{C+1} exp(\hat{z})} \tag{20}$$

and the cross-entropy as

$$H_{xe}(\hat{h}) = -\sum_{c=0}^{C+1} h_c \log(\hat{h}_c) + (1 - h_c) log(1 - \hat{h}_c). \tag{21}$$

Then, the total loss used for the network optimization is the average cross-entropy over all pixels.

The width of the peaks is the standard deviation (σ) of the Gaussian. Depending on the width W of the input image, we set $\sigma = W/32$ which equals 8 pixels for the selected image resolution of 64×256. Although wider peaks lead to faster convergence of the CNN training, localization accuracy decreases when neighboring peaks overlap. Even if the locations are in different classes, and thus channels, the normalization would falsify the peak shape and the absolute value. On the other hand, if the peaks are to small, predicting zero for the whole heatmap results in a low error when averaged over all pixels which might lead to a local minimum where the optimization can easily get stuck.

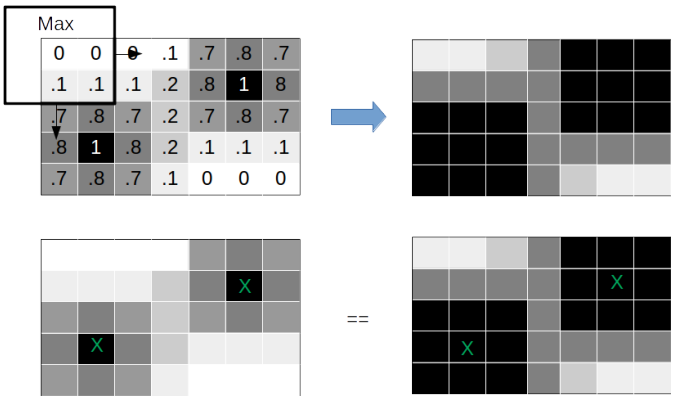


Figure 32: Local peak detection. For simplification, the heatmaps have 5x7 pixels. A maximum filter slides over the image. As a result, each pixel gets assigned the value of the pixel with the highest intensity under the shape of filter. If we compare the resulting image with the original heatmap, a local maximum is found at the position where the pixel values of the result and the original are equal.

One potential drawback is the readout of the peak positions in a heatmap. In many heatmap regression applications, this is done by simply taking the $argmax$. However, in the reduced classification, we may have multiple locations and therefore multiple peaks in one layer. The locations need to be detected by finding multiple local maxima. We detect the local peaks using a maximum filter operation and a pixel wise comparison as shown in Figure 32. For the predicted tooth location heatmaps, we select the same filter size σ as for the Gaussian kernel used to construct the ground truth heatmaps. Furthermore, we dismiss locations that are closer to each other than the filter width σ. Also peaks with a value below 0.5 are not considered. However, if we only regard the pixel index of the maximum, we lose positional accuracy related to the resolution of the heatmap. This is a problem for the typical down-convolution CNN architectures since the resolution of the feature maps is reduced with every down sampling layer but is negligible for the heatmap resolution of 64×256 that can be generated by the U-Net.

3.6.2 Network Architecture

The U-Net architecture is used to regress the heatmaps in the same resolution as the input images. Usually used for segmentation tasks where pixel-wise accuracy is desired, the U-Net is also a state-of-the-art tool to use for landmark prediction [PSBU16].

The expanding part of the U-Net learns how to up-scale the feature maps depending on the previous layer and the feature maps at the same stage. This way,

the feature activations can be propagated smoothly and the locations can get fine tuned based on the lower level features.

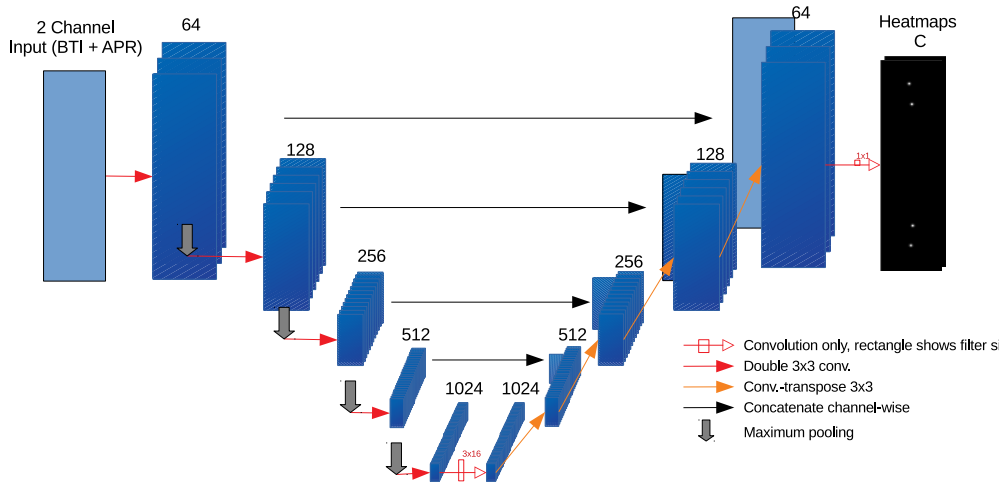


Figure 33: U-Net architecture for tooth detection. Annotated are the number of used feature maps in each stage. Since we use padding to better include the image border, the image size is halved with every downward stags.

We deploy the original U-Net architecture extended by batch normalization (see Section 2.2.3) after each 2D convolution, zero padded convolutions, and a wider convolution filter at the deepest convolution stage. The modified U-Net architecture is shown in Figure 33.

Since in our images the third molars may be very close to the image borders, we use zero padding with every convolution s.t. the filters are applied over the whole input image and the output heatmap has the same size as the input images. Otherwise, initializations, sampling, and filter sizes are equal to the original U-Net.

Payer et al. [PSBU16] argue that despite having very good localization performance, the U-Net architecture can be more prone to outliers than an architecture that is explicitly addressing landmark inter-dependencies. Since we have rigid relations between the tooth positions, we try to encode this knowledge into the network by using a wide convolution filter (kernel) on the deepest convolution stage of the U-Net. We leave the filter height as defined by the network and use a 3x16 convolution filter as shown in Figure 34. Doing so, we exploit the knowledge that the teeth are spread along the x-dimension in the BTI. Therefore, the filter can combine information of all neighboring tooth positions of the next group of teeth

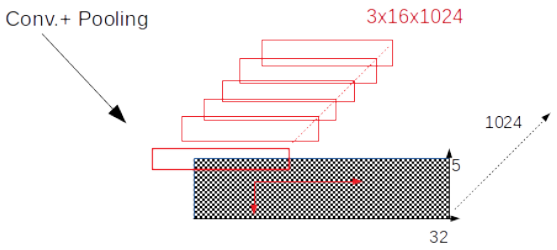


Figure 34: Application of the wide convolution filter. At the deepest convolution stage, a single filter with a width of 16 replaces the 3x3 double convolution. Due to the increased width the filter can directly learn feature dependencies that are spread along the x-axis of the image.

(of one tooth type) in both directions. The direct neighborhood information is crucial for the explicit tooth numbering. Also, it provides additional information for classification if the local appearances of the break through interface and of the frontal view of the tooth are insufficient.

4 Experiments and Results

This chapter analyzes the different learning scenarios targeted by our approach. After describing the hard- and software setup and the data set used for the CNN training, we evaluate the architectures proposed in 3.5 and 3.6. As final evaluation, we take the best performing architecture for each regression model and show the resulting 3D landmarks in Section 4.5. To do this, and to prove the general validity and generalization performance, we perform a 6-fold cross validation over the complete data set. Both approaches are also evaluated for localization only and the 8-class classification scenario in Section 4.6.

4.1 Computational Setup

We implemented the manual tooth annotation tool as a plug-in written in C++for the ZIB version of Amira 6 . To control the Amira QIST module, we wrote Python scripts for the Amira interface. Likewise, we also control the further alignment of surfaces and the sampling of the image data of the 3D surfaces when generating the training images. From there on, all data processing, i.e. data loading, reordering for different classification levels, data augmentation, as well as all machine learning is implemented in Python. For the implementation of the CNNs, we use the Tensorflow-1.10.0 [AAB+15] backend with cuda-9.0 [NBGS08]. All models are trained using Intel(R) Xeon(R) CPU E5-2640 v4 and a Nvidia Tesla P100 SXM2 GPU with 16 GB memory.

4.2 Training

From a set of 5000 clinical CBCT scans with a voxel size of $0.25^3 mm$ and an intensity range from -1000 to 7190, 158 segmentations were generated by 1000Shapes on which we annotated the tooth positions. Of those 158 data sets, 146 uniquely belong to distinct persons and 12 cases are pairs of two different scans belonging two 6 persons. The images that are different cases but belong to the same person usually are quite similar but have slight differences in intensity, orientation, or show different stages of surgical procedures, therefore we do not remove them from the data set but make sure no data belonging to the same individual is used in both, training and validation set.

For 118 cases, we could confidently annotate the full enumeration of the teeth. As explained in Section 3.2, big gaps or moved teeth sometimes make it difficult to identify the tooth number, s.t. it is impossible for us to enumerate the teeth without knowledge of the patients pathological history. For the remaining 40

scans where we could not enumerate all teeth, we annotated the tooth type and the sector, s.t. they are applicable for the 8 class classification.

We chose a set of 20 scans that is not used for training, any preprocessing or hyper-parameter selection as dedicated validation set. This set consists of the first 20 of the enumerated cases. However, we manually confirmed that it does not randomly consist of particular easy or difficult cases. Due to memory limitations and because we want to be able to directly compare all approaches, we use a relatively small but fixed batch size of 8.

For optimization, we use the Adam variant of stochastic gradient descent proposed by Kingma and Ba [KB14] which is implemented by the Tensorflow-learn framework. It generally requires less fine-tuning of parameters like learning rate, decay or momentum, but provides fast convergence and robustness against local minima. We leave two additionally introduced parameters that control the decay of the adaptive momentum at their proposed default [KB14] as also implemented within the framework. Except for the learning rate, no further configuration is necessary compared to a stochastic gradient descent with momentum term for which the optimal parameter configuration differed in each of our scenarios and especially the numeric regression proved susceptible to local minima, when varying learning rate or momentum.

4.3 Numeric Regression Experiments

We compare the loss functions explained in Section 3.5. To evaluate the losses, we look at the classification accuracy, outliers and the distances separately. For setting the coordinates (xy) to a fixed value λ, we write '$fix\lambda$'. The second loss regressing 3D coordinates is denoted as $d3D$, and the combination of distance and probabilistic loss as L_{pd}. Figure 35 shows the development of accuracy and mean distance during the training process evaluated on the validation set. The values are calculated once after each epoch. For the comparison, the CNN from Section 3.5.3 with 5 stages and one FC layer with 3×265 neurons is used. The training was stopped after 400 epochs, where most of the scenarios showed no further improvement. We did not observe over-fitting for any of the scenarios. As argued in Section 3.5, for the third loss, we chose $k = 20$ and $\varepsilon = 10^{-8}$. For the calculation of accuracy and distance, hard classification thresholds are used s.t. either locations outside the image or $\hat{p}_t < 0.5$ are interpreted as 'predicted missing'. The distance is only evaluated for true positives. Therefore, we transform the (x, y) prediction of the CNN trained with the first loss ($fix\lambda$) by interpreting coordinates >1, or <0.1 if zero is used as fix-point ($\lambda = 0$), respectively, as predicted negatives.

Moving the coordinates for missing teeth to a fixed point outside the image gen-

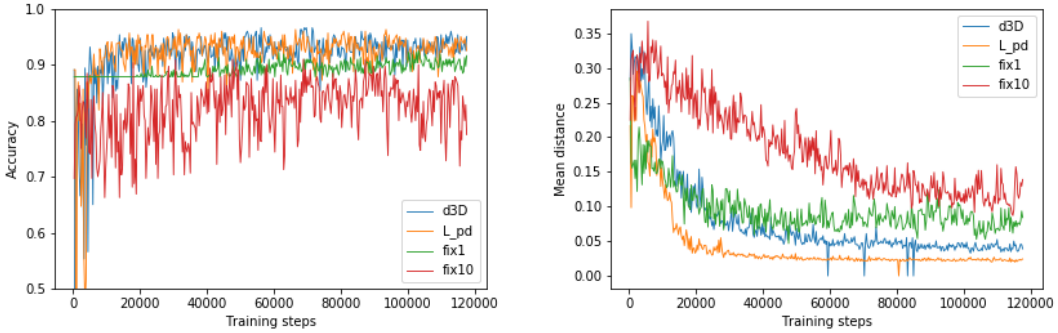


Figure 35: Percentage of correctly predicted teeth (left) and mean distance (right) for different loss functions. Note, that distances are calculated only over true positives. Only a small batch of 8 randomly chosen cases was used for validation causing the jitter.

erally performs worse than the other losses. The higher the numerical value the harder it becomes for the network to approximate the existing positions properly. The graph for a fixed value of 10^8 is outside the image frame in Figure 35 and does not converge towards a comparable range. Therefore, it is not regarded further.

Setting missing entries to zero, with and without interpreting the $\hat{p}$ value as third coordinate seems to have almost the same effect, both achieve the highest accuracy. The proposed loss in Equation 8 performs best in both, localization and classification. A visual representation of the effects of the loss function is show in Figure 36.

While the selection of the correct loss function proved crucial to the prediction performance, we also evaluated different network architectures. Besides changing the connection of the FC layers, we investigated different numbers of convolution stages and different FC and filter sizes. For this, the loss function L_{pd} is used. As visible in Table 5, the results are similar for all different network architectures but some differences are significant. The same network with 5 layers but without the compression performs worse than with the compression stage by a margin in distance and outliers. One could suspect that the compression improves the results because it introduces another convolution but in contrary, when replacing the 16 1x1 filter by another convolution, i.e. in a CNN with 6 stages that does not reduce the number of feature channels, the performance is worse. However, since all numerical models converge slowly and might not have reached their minimum, it might be that the high number of weights that is required without the reduction of feature channels only slows down the convergence but might eventually find a better minimum after longer training.

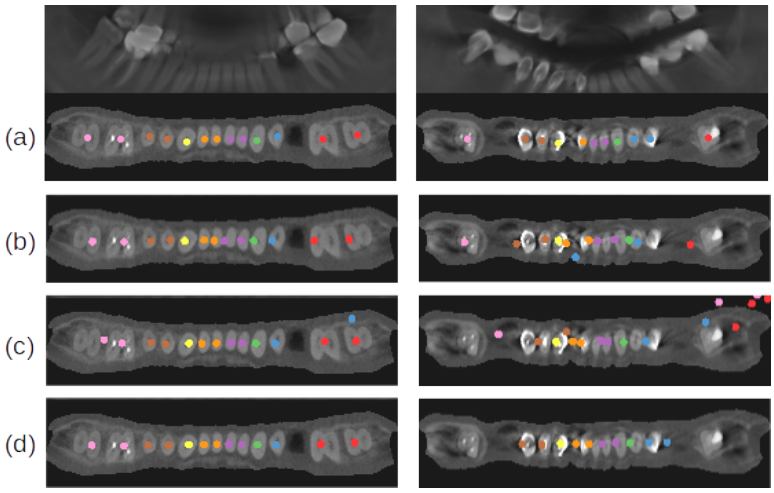


Figure 36: Results for numeric regression losses. (a) ground truth, (b) d3D, (c) fix1, (d) L_{pd}. Landmark colors are group-wise. The left case is predicted correctly except one false positive premolar (b). The right case is more difficult, landmarks with less certainty move towards the fixed point (c) and (b).

Using fewer stages but larger convolution filters performs almost equally good. A downside of using larger filters is that they require more time per training step. Furthermore, we could not find that using larger filters improves the classification by learning better representation of inter-landmark dependencies. This information can apparently be transported by deeper networks with multiple pooling stages, too.

In summary, the details of the CNN architecture do not have a high impact on the performance. For the following comparisons, we employ the network with 5 stages, compression, 3x3 filter, and 3x256 FC units since it has good performance and is fast to train. Considering the localization, none of the models yields satisfactory results compared to a human level. This is already visible in Figure 36 (d). We would expect more accurate placement of the landmarks, especially for clearly visible tooth contours.

4.4 Heatmap Regression Experiments

For the U-Net architecture, the effect of the wide filter was evaluated. We tested learning rates in different magnitudes from $\eta = [0.01, \ldots, 0.00001]$ where $\eta = 0.0001$ was the best performing.

A comparison between the original U-Net and the adaption with a wide filter is shown in Table 6. After 50 epochs, in all cases, training and validation loss reach a

Table 5: Comparison CNN architectures. Different numbers of convolution stages with or without feature compression are evaluated (Arch) in combination with different convolution filter widths (Conv.) and different numbers of FC sizes (FC). All training scenarios where stopped after 400 epochs. Given are the combined loss L_{pd}, accuracy, and the percentage of outliers >0.05 (O>0.05) and the distance in normalized image coordinates with one standard deviation.

Arch.	Conv.	FC	Loss(L_{pd})	Accuracy	O(>0.05)[%]	Dist.
5+c	3	3x64	1.946	0.941	4.62	0.023±0.010
5+c	3	3x256	1.538	0.962	2.49	0.018±0.009
5+c	3	3x1024	1.610	0.953	4.98	0.024±0.011
5,	3	3x256	2.015	0.947	8.89	0.028±0.011
6,	3	3x256	2.345	0.931	18.86	0.054±0.014
4+c	5	3x256	1.463	0.959	2.135	0.018±0.008
5+c	5	3x256	1.661	0.938	2.135	0.019±0.009
2+c	16	3x256	1.619	0.094	4.27	0.019±0.009
2+c	16	3x1024	1.628	0.953	7.11	0.024±0.010

plateau and the training was stopped. When comparing the accuracy, the heatmap approach performs similar to the numeric regression but it outperforms the numeric regression with a twice as good localization and fewer outliers. Figure 37 shows the prediction for three CBCT cases of the adapted U-Net. The heatmaps have smooth peaks and thus the localization is accurate. There are no cases where the heatmap contains a visible location that was not found by the maximum search.

Although the difference is only marginal, the adapted U-Net has slightly better classification accuracy. The higher number of outliers is reasonable, since there are more true positive predictions that can be considered for the distance loss.

Table 6: Heatmap regression results. The original U-Net with 5 stages, 3x3 filters, but used with padding compared to the modifies U-Net with single (+W) or double (+2W) convolution with a wider convolution filter in the lowest stage. The difference is evaluated in terms of accuracy, percentage of outliers with a distance greater than 0.05 in normalized image coordinates (O(>0.05)) and the average distance in normalized image coordinates (Dist.).

Arch.	Loss (H_{xe})	Accuracy	O(>0.05)[%]	Dist.
U-Net+W	0.088	0.953	0.71	0.011±0.006
U-Net	0.107	0.872	1.07	0.011±0.006
U-Net+2W	0.089	0.947	0.71	0.011±0.006

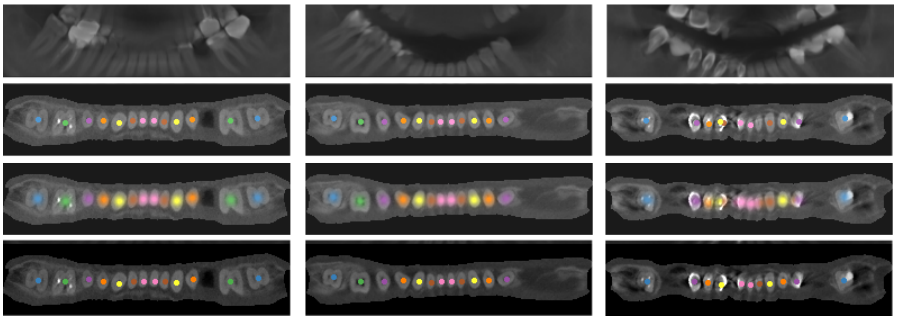


Figure 37: Prediction of the adapted U-Net after 50 epochs. The columns show three different example cases. The rows depict the APR, BTI with predicted locations, the predicted heatmaps, and finally the BTI with ground truth locations. The colors for left and right sector are the same the full 16 classes were predicted and no left-right confusion was detected. All predictions are accurate except for the third (right) case where the narrow incisors and multiple missing teeth increase the prediction difficulty. The mixed colors in the blurry heatmap can be seen as indication of a higher uncertainty.

Furthermore, these are probably difficult cases that are also hard to localize.

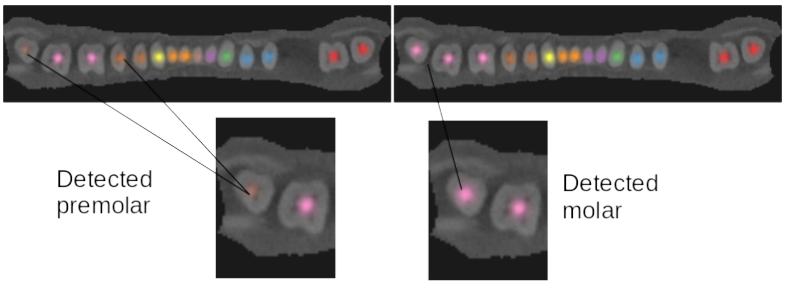


Figure 38: Heatmap prediction at epoch 40 of the standard U-Net (left) and the modified U-Net with wider filter (right). Here, the color mapping distinguishes left from right sector and tooth types but not the explicit number. The falsely classified tooth by the U-Net does not have the typical larger appearance of the third molar but rather one of a premolar. At the same number of training steps, the adapted U-Net correctly classifies this tooth despite its degenerate appearance.

Figure 38 shows an early stage prediction comparing the original U-Net and our adapted version. This is a good example showing where global dependencies between teeth are more important than the local appearance of the tooth.

4.5 Results (3D)

Until now, all comparisons were based on the normalized distances in the BTI. In the following, we evaluate the reverse projected predictions of the tooth locations by taking the 3D Euclidean distance to the ground truth annotations.

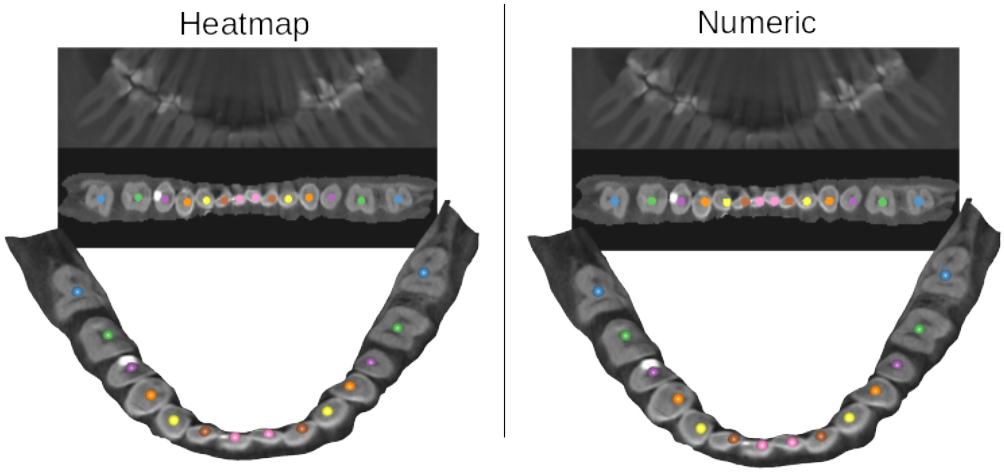


Figure 39: Prediction in 2D and 3D. For the selected case, both models classify all teeth correctly. the better localization by the heatmap approach can be particularly seen on the patient's right canine and premolars. However, the displacement in the 2D images corresponds to the 3D deviation.

In order to validate the generalization of our approach, we perform a 6-fold cross-validation. The cross validation is done for both, the numeric and the heatmap regression in the 16-class scenario. The heatmap model with the adapted U-Net was trained for 50 epochs. The numeric model had 5 stages, the compression layer, and 3×256 neurons in one FC layer and was trained for 400 epochs. The size of the validation set remains 20, such that for the last set of the 118 cases the first two cases are repeated. The training is conducted 6 times with a different set left out for validation and the results are computed as the average prediction scores over the test sets. As shown in Figure 39, we can see that the transport from 2D to 3D and vice versa works seamlessly.

If setting a localization accuracy of 2mm as a requirement, we can calculate an effective prediction accuracy. In table 7, we show the final results for the cross validation for heatmap and numeric regression. Figure 40 visualizes the distance distribution and the effective classification. Regarding a qualitative assessment, the box plot in Figure 40 is more informative showing the actual distance distribution and not just the mean distance which is not robust against outliers. Further scores, evaluated for each tooth type, and further example predictions are given in the

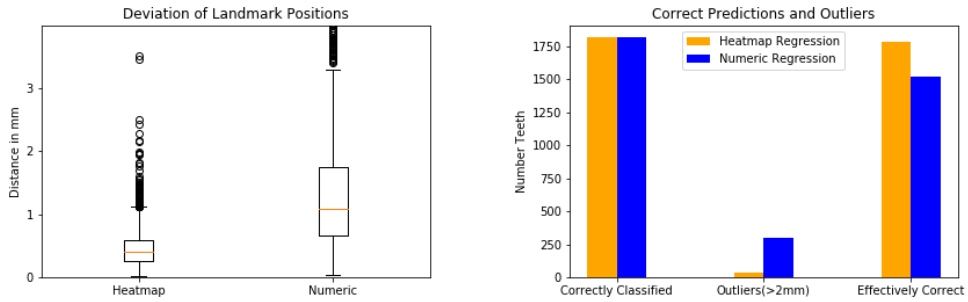


Figure 40: Comparison of 3D prediction for heatmap and numeric coordinate regression. Depending on the threshold that is selected to interpret a landmark as outlier, the numeric regression becomes significantly worse which is shown by the 'Effectively Correct' entries that subtract the outliers from the number of correct predictions.

appendix (Sec. 6).

Table 7: Cross validation results for numeric and heatmap regression evaluated in 3D. Given are accuracy, outliers greater than 2mm (O(>2mm)), average distance, and fraction of correct predictions with a distance less than 2mm (Effective). The higher average distance of the heatmap approach comes from a few heavy outliers that are actually false predictions. Nevertheless, the total number of outliers is significantly smaller for the heatmap approach than for the numeric regression. Thus, resulting in a better effective classification accuracy.

Arch.	Accuracy	O(>2mm)[%]	Dist.[mm]	Effective
Num 3D	0.965	17.61	1.40	0.788
Heat 3D	0.964	1.89	2.38	0.945

4.6 Reduced Classification

The following shows an evaluation for a coarser classification model categorizing teeth into type and sector (8 classes) or not at all for a general tooth localization without classification (1 class). For the reduced classification, we use all 158 annotated cases, again with 20 cases reserved for testing. In this set, however, there are two cases belonging to the same person would have been in training and validation set, respectively. To avoid this, we simply swap the case of the training set into the validation set and remove the next validation case that has no corresponding case belonging the same person.

For the 8 class scenario, the ground truth entries for the numeric regression are sorted as proposed in Table 4. If only a localization is required, the entries are sorted by their x-coordinate. Table 8 shows the evaluation results for the numeric and the heatmap regression approaches trained to predict 8 or 1 classes.

Table 8: Results for coarser classification into 8 or 1 class. Results are shown for heatmap regression (Heat) and numeric coordinate regression (Num). For the 1 class scenario for the numeric regression, we had to apply a sigmoid function to the coordinate output, since the high initial distances lead to numerical overflow making the optimization infeasible. The sigmoid, however, has a severe impact on the localization accuracy producing far more outliers and thus incorrect predictions.

Model	Classes	Loss	Accuracy	O(>0.05)[%]	Dist.
Heat	8	0.042	0.972	1.149	0.010
Num	8	1.695	0.946	3.831	0.021
Heat	1	0.042	0.972	1.149	0.010
Num+sig	1	2.514	0.934	40.23	0.51

The performance of the 8-class model is equal to the 16-class scenario. The CNN can learn the sorting of the output channels in a certain degree but for the prediction of locations only (1-class) which should actually be an easier task, the additionally induced challenge of the reordering is more severe and causes numerical problems when applying the model used for the 16-class scenario. Using the sigmoid to bind the numerical output stabilizes the training but performs far worse in terms of localization.

Permuting Entries of One Group

As side experiment, we evaluated the performance of the permuted loss presented in Equation 16. We apply the same CNN architecture with 5 convolution layers,

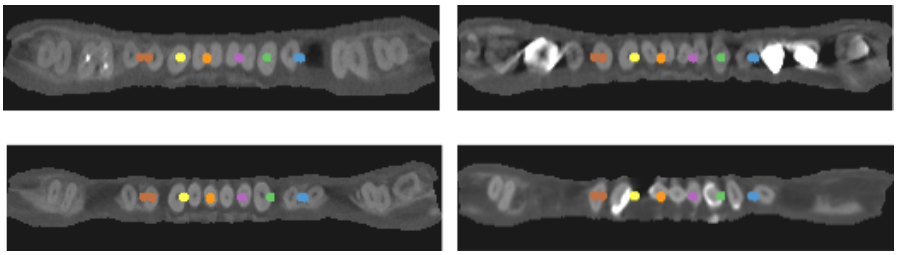


Figure 41: Prediction of a CNN trained with arbitrary order of landmarks within one class for the 8 class model (LM=red, LP=blue, LC=green, LI=purple, RI=orange, RC=yellow, RP=brown RM=pink). Predictions of the individual teeth within one group converge to the center location.

compression, and one FC layer with 3×256 neurons. As visible in Figure 41, the loss function has the opposite of the desired effect. Even with the small learning rate of 0.0001 the training gets stuck at a local minimum with a loss of $L \approx 6.5$. We can observe that due to the freedom of which channel to assign to which entry, the network does not learn the actual ordering of teeth. All output channels predict the middle position of the sub-group, s.t. the loss is almost independent of how the channels are swapped. Molars are always predicted missing which is plausible because the standard deviation of the molar-group centers would result in a higher loss than the miss-classification. However, the prediction indeed starts in the group centers but slowly diverges for the different teeth. Although the time for further investigations was too short, this approach could prove valid with larger data sets, data augmentation or transfer learning.

4.7 Data Augmentation and Feature Importance

We evaluated the benefit of the methods proposed in Section 3.3, and furthermore, how much information comes from the APR and from the BTI. The augmentation methods were foremost developed to improve the numerical regression because of the observed tendency to predict the statistical mean of all tooth positions. Thus, the following compares the data augmentation methods for the best numeric regression model as well as for the heatmap approach.

To evaluate the effect of the methods rotation, flip, and tooth removal, the considered augmentation method is left out and the data in the training set is duplicated to match the number of training steps within one epoch. The effect of the additional input channel with the synthesized APR is evaluated by only using the BTI as input training data. Table 9 shows results for models trained with reduced data augmentation or with reduced input channels.

Table 9: Evaluation of data augmentation methods. For each test, one method is left out and the data is repeated instead to maintain an equal number of training samples. Training was aborted after 50 epochs for heatmap regression and after 400 for numeric regression. The given loss for heatmap regression is H_{xe} and for numeric regression L_{pd}. Average distance and threshold for outliers (O>0.05) are still in normalized image coordinates.

Augmentation	Loss	Accuracy	O(>0.05)%	Dist.
Full Heatmap	0.088	0.953	0.712	0.011
No gamma	0.084	0.959	1.432	0.012
No flip l/r	0.088	0.944	0.712	0.013
No remove	0.084	0.944	0.0	0.011
No rotation	0.089	0.941	0.712	0.012
No pan.	0.086	0.944	0.356	0.010
No APR	0.103	0.837	0.0	0.012
Full Numeric	1.538	0.962	2.491	0.018
No gamma	1.344	0.966	2.491	0.018
No flip l/r	1.855	0.947	3.203	0.021
No remove	2.066	0.938	7.117	0.026
No rotation	1.595	0.953	3.559	0.022
No APR	2.015	0.944	5.338	0.024
No aug.	2.130	0.931	9.609	0.025

Leaving out a single data augmentation method does not show a notable drop in performance but in some cases even an improvement. Not using gamma augmentation even results in a slightly better performance for both regression types. However, this is reasonable because also the validation data comes from the same CBCT scanner. We still hold this to be a valuable augmentation method, since we want our approach to be generalizable as much as possible. It is interesting to see that the augmentation methods that are actually modifying the landmark positions (flipping, rotation, or removing) have a slightly bigger impact on the numerical regression and almost none on the heatmap regression. A likely reason for this is that the heatmap model directly transforms the local appearances into the landmark representation whereas the fully connected layers in the numeric model can fit a bias towards the statistical mean of the landmark positions. Missing the APR as additional training input still performs surprisingly good considering that for a human the panoramic image provides essential information. Only when omitting all augmentation methods (APR is still used), the reduced accuracy and higher number of outliers become eminent. Thus, showing that the overall application of our data augmentation indeed boosts the prediction performance.

5 Conclusion

In this chapter, we summarize the results obtained in Chapter 4. Section 5.1 refers to the set objectives and discusses the found results. Furthermore, in Section 5.2, we give recommendations on possible improvements and point out approaches for future work.

5.1 Discussion

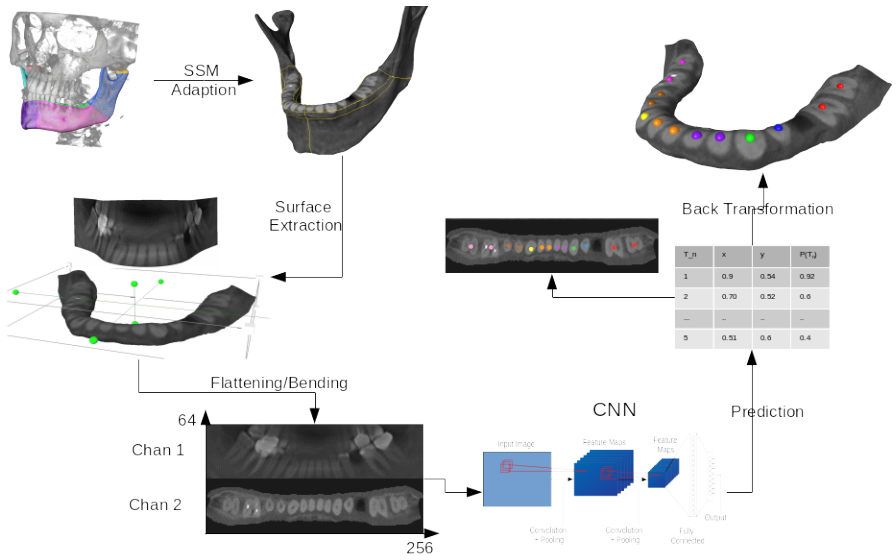
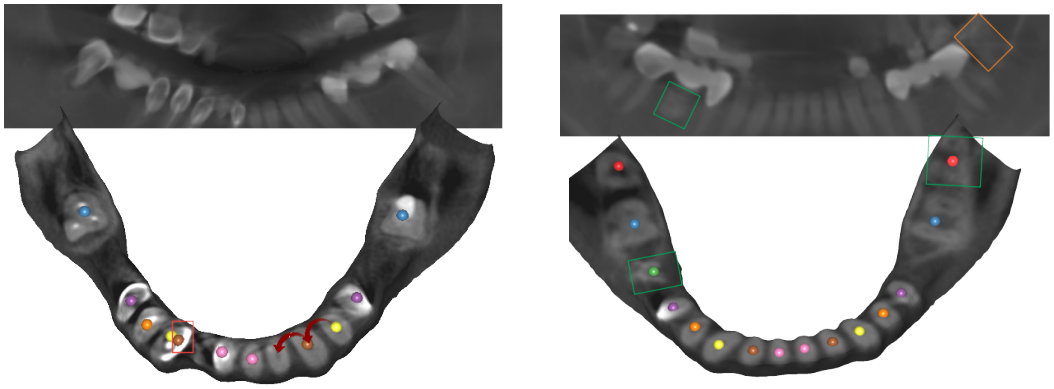


Figure 42: Tooth detection pipeline. From volumetric data, we generate the surface reconstruction. The flattened region of the dentition and a virtually generated panoramic radiograph are used as CNN input. We use the trained CNN to predict the unknown tooth locations in the images. The 3D coordinate are calculated by mesh correspondence.

We presented a sound approach to localize and classify teeth in 3D dental CBCT data. It can cope with missing teeth, dental fillings, implants, and artifacts while providing accurate localization. Full tooth enumeration or partial classification into tooth types and sectors can be learned and can be employed depending on the use case.

We efficiently combine data pre-processing, dimension reduction and deep learning techniques. The data dimensionality is reduced by extracting images of the dentition interface on the mandibular bone surface (BTI) and by artificially generated panoramic x-ray images (APR). For pattern recognition, we made use of supervised learning techniques with CNN based architectures to automatically predict

tooth locations and classes. An overview of the data processing pipeline as applied to generate the tooth locations for new data is depicted in Figure 42.



(a) (b)

Figure 43: Prediction (adapted U-Net) for difficult cases with full tooth enumeration. a) The two lateral incisors (brown) are both predicted incorrectly and shifted to the outside resulting in consecutive false classifications of the lower left canine and first premolar. b) Good prediction despite metal artifacts and gaps. The lower left third molar was detected although hardly visible in the APR (orange rectangle). The missing lower right first molar is predicted correctly although there is no clear contour of the break through interface on the surface (left green rectangle). However, the tooth is visible in the APR.

Figure 43 shows two difficult cases with tooth loss and dental bridges. Although the miss-classification in (a) is severe, the propagation of the error is plausible and the localization the other teeth still accurate. Coping with missing incisors may be difficult for the trained model since this is a rare condition in our training data (see. number of teeth in Appendix, e.g. Fig. 48).

Our approach specifically addresses tooth loss by the designed loss function for numeric regression or using multiple layers in heatmap regression. The heatmap regression achieves a good prediction accuracy of above 94 % of theeth that are within a 2mm range of our manual annotation. This is an accurate localization that is visually not distinguishable from human annotations.

Although the numeric regression does not achieve satisfactory results, there exist techniques for potential improvements that could be applied (see [SWT13] and [HKZ+17]). In particular, the approach of He et al. [HKZ+17] would be interesting, since the cropped image is combined with the feature maps of an intermediate layer that is pooled and thus has the same dimension as the cropped region which could potentially empower the network to learn a global landmark dependencies.

The heatmap regression with the U-Net achieves the good results by directly transforming the input images to a location probability map. With the adaption of a wider convolution filter, we showed a reasonable improvement that potentially applies to any scenario where a global spatial configuration should be considered. Covering more of the spatial dimension with one convolution filter could also be achieved by using more downwards convolution stages in the U-Net leading to a smaller feature where the information can be processed by the 3x3 filters. However, we chose to incorporate the knowledge that the teeth in the BTI are only spaced out along the x-dimension by increasing the width of the convolution filter effectively improving the prediction quality.

Regarding potential applications, our algorithm could be used for automated filling of dental charts and also for statistical studies evaluating tooth positions or relative distances. The current state could be used as an initialization and would need further human correction since also teeth are predicted falsely that are easily recognized by a human examiner.

Feature Selection Revised

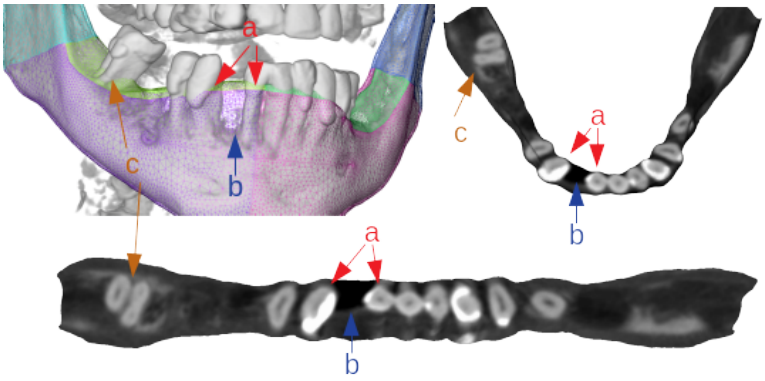


Figure 44: Feature extraction. Although the mandibular bone has partially degraded (b), the surface is smoothly connected between the two neighboring teeth. Instead of following the bone which would result in the surface cutting through the root, the surface cuts through the teeth too high (a), resulting in the inclusion of metal copings and a hardly visible pulp chamber. Towards the molar (c), the surface fits the degraded bone which is correct but results in separated root contours.

The flattened view of the break through interfaces in the bony surface of the mandible (BTI) proved to contain good features for localization, and the frontal panoramic view onto the teeth (APR) could further enhance the capability of our model. By visualizing the intensities on the surface, we can see the tooth interfaces

as light isles with a darker dot in the middle where the pulp chamber is located resulting in reliable features that could be used for a CNN training and prediction. Due to inaccuracies in the fitting of the SSM to the actual bone, the surface may be cutting the teeth higher above the bone surface or lower towards the roots of the teeth as shown in Figure 44. Nevertheless, all teeth are clearly visible in on the extracted surface, the flattened mandible, and the BTI.

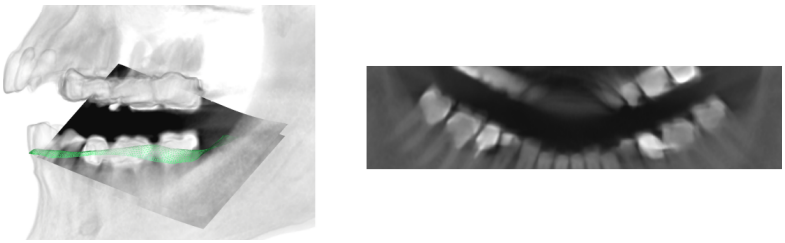


Figure 45: Bad case of surface orientation for generation of the APR. The ruled surface that is used to sample the artificial x-ray image is not aligned with the teeth due to its perpendicular orientation to the SSM patch.

Although the APR indeed contributes to the prediction performance, establishing exact spatial correspondence between teeth in the BTI and the APR should be beneficial. Furthermore, the plane fit through the SSM patch that defines the orientation of the ruled surface for the radiograph generation is not optimal, leading to sometimes inward tilted surfaces as in Figure 45. The resulting projection directions can lead to partial cut-offs of the incisors and to curvature of the tooth row towards the sides of the image although the teeth should optimally be on a line.

5.2 Outlook

Because this thesis is a first investigation of deep learning approaches for tooth detection in CBCT there are a lot of opportunities how to improve it directly or to extend this approach using the gained information in combination with other approaches.

Improvements

In general, our approach would highly benefit from more and also more versatile training data. E.g. cross CBCT machine applicability is not guaranteed since we only train on data from one machine. For deployment in medical applications, this is mandatory but also creating more segmentation from the same CBCT data set and labeling them would be beneficial. The manual effort, to annotate tooth positions is relatively small and with our implemented CNN prediction pipeline, initial estimates for new ground truth data could be generated. Although there are actually more CBCT data sets available by 1000Shapes, the adaption of the SSM can be tedious if the CBCT image data has low quality. Therefore, generating segmentation data from the remaining CBCT scans would require improvements of the segmentation algorithm or further manual interaction.

Besides increasing the training data, correcting the orientation of the APR is a promising enhancement that could be applied directly to our approach. Using the fixed orientation of the head in the CBCT volume and aligning the surface along one axis with a potential offset would likely result in a better projection. By using the same center line for the ruled surface of the APR as used to normalize the U-shaped surface onto a line in the BTI, both surfaces could be normalized together preserving the spatial correspondence between the training images.

To further enhance the model, the data augmentation, i.e. especially the artificial tooth loss could be improved. Editing the images in with a sophisticated graphic software or scripting more advanced image processing techniques to modify the images would lead to better augmented images but would require a lot of hand crafted algorithms. An alternative is to use Generative Adversarial Networks (GANs) as introduced by Goodfellow et al. [GPAM+14] to produce realistic looking images. However, encoding a specific combination of which teeth are present, may not be trivial using GANs but combining the procedural modification with adversarial learning could be an interesting research topic.

Extensions and Applications

The most obvious extension is to use the found locations to extract single-tooth regions of interest. Those regions can then be used to perform a tooth-wise segmentation. E.g. another SSM, but one for every tooth, could be used to place templates on the found positions adapting them to the image data, and thus generating the segmentation. However, this approach would again result in an error propagation if the region was not correctly detected.

It could also be interesting to improve a 3D segmentation approach, e.g. by Ezhov et al. [EZG18] using our results as regularization or as a bias for voxel-wise class probabilities.

Another extension could be to utilize the properties of the QIST surface transformation in 3D. The bending application can also be used to normalize the whole volumetric segmentation of the mandibular bone onto a line. Then, by registration of the image data of the original and the morphed surface, the resulting 3D image can be used for new visualization techniques or more effective 3D convolution approaches. It would also be possible to use this volumetric rendering to generate panoramic radiographs, possibly from multiple angles that could all be passed to the CNN. This would solve the issue of the tilted panoramic surfaces and provide additional features.

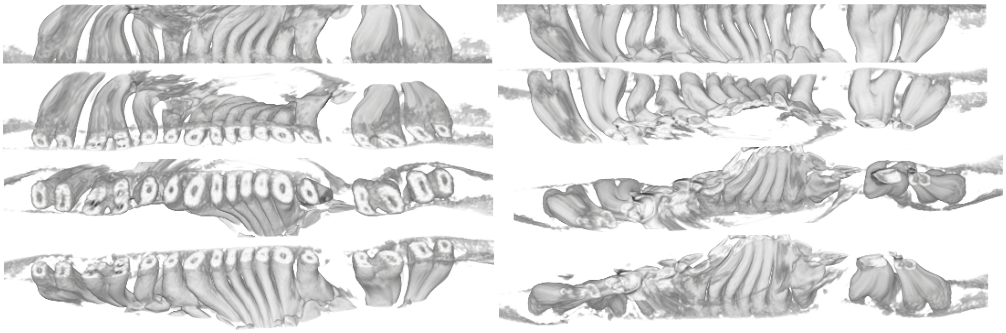


Figure 46: Volumetric rendering of the tooth region. The volume is rotated in 45° steps starting with a frontal view, first rotating the bottom upwards to the front (left) then, further pointing the top of the teeth towards the viewer (right).

We generated an example for one case. Figure 46 shows a volumetric rendering of the tooth region from different angles. Although the teeth are skewed by the deformation, the different angles enhance the understanding of the anatomical structure. The distortions are due to the fact that we only provide correspondence

for the segmented mandible surface and not for the actual teeth region which are above the surface. Furthermore, transforming the whole mandibular bone including the posterior bone parts (condyle, ramus, angle) is actually not necessary for the tooth region and might be a reason for the distortions, too.

6 Appendix

6.1 Numeric Regression Results

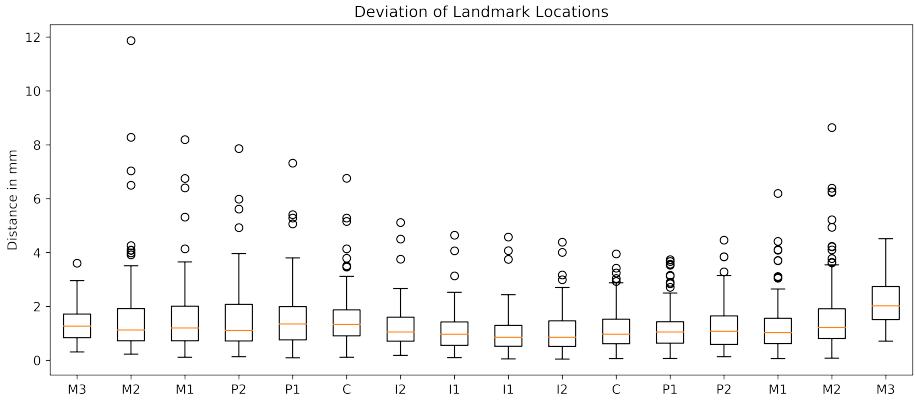


Figure 47: Distance distribution for 6-fold cross validation of the numeric regression model trained for 400 epochs. The number of outliers increases with the tooth size. For the third molar, however not enough true positive predictions might be given for statistical validity.

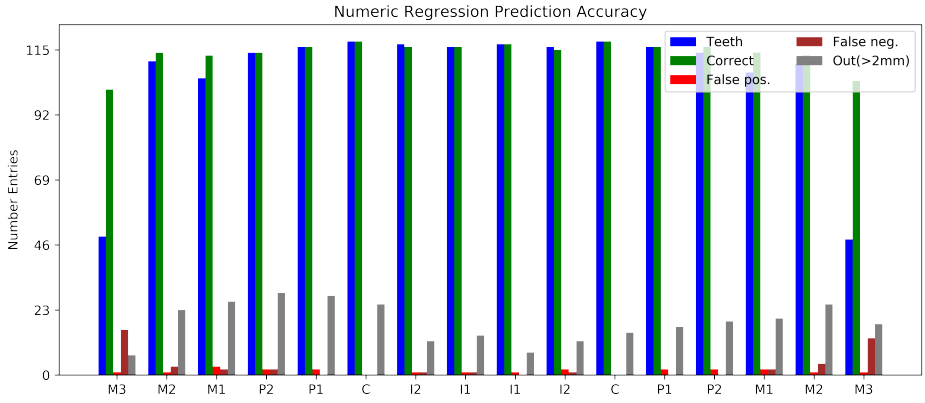


Figure 48: Results for 6-fold cross validation of the numeric regression model trained for 400 epochs. False negatives are particularly high for the third molars. The bias be caused by the high number of missing third molars in the training data.

Figure 49: Example predictions (3D).

6.2 Heatmap Regression Results

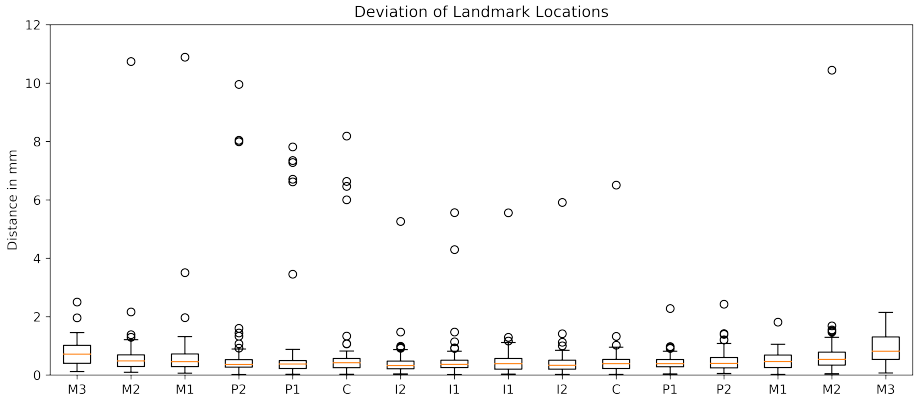


Figure 50: Distance distribution for 6-fold cross validation of the heatmap regression model trained for 50 epochs. Some outliers that can be interpreted as false predictions are present but overall localization is accurate.

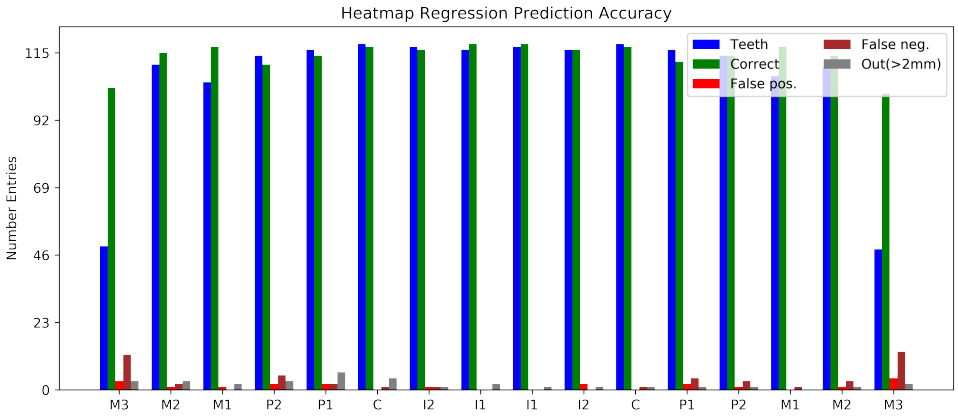


Figure 51: Results for 6-fold cross validation of the heatmap regression model trained for 50 epochs.

Figure 52: Example predictions (3D).

Acronyms

ANN Artificial Neural Network

APR Artificial Panoramic Radiograph

BTI Break Through Image - Referring to the normalized image of the flattened dentition surface

CBCT Cone Beam Computed Tomography

CNN Convolutional Neural Network

CT Computed-Tomography

DCBCT Dental Cone Beam Computed Tomography

DNN Deep Neural Network

FC Fully Connected (layer)

FCN Fully Convolutional Network

FDI Fédération Dentaire Internationale - World Dental Foundation

MRI Magnetic Resonance Imaging

PDM Point Distribution Model

QIST Quasi Isometric Surface Transformation

R-CNN Regions with CNN Features

ReLU Rectified Linear Unit

SSM Statistical Shape Model

ZIB Zuse Institute Berlin

References

[AAB+15] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.

[DDS+09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[DLKZ12] Nguyen The Duy, Hans Lamecker, Dagmar Kainmueller, and Stefan Zachow. Automatic detection and classification of teeth in ct data. In Nicholas Ayache, Hervé Delingette, Polina Golland, and Kensaku Mori, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2012*, pages 609–616, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg.

[EZG18] Matvey Ezhov, Adel Zakirov, and Maxim Gusarev. Coarse-to-fine volumetric segmentation of teeth in cone-beam CT. *CoRR*, abs/1810.10293, 2018.

[FDK84] L. A. Feldkamp, L. C. Davis, and J. W. Kress. Practical cone-beam algorithm. *J. Opt. Soc. Am. A*, 1(6):612–619, Jun 1984.

[Flo03] Michael S. Floater. Mean value coordinates. *Comput. Aided Geom. Des.*, 20(1):19–27, March 2003.

[GC10] Hui Gao and Oksam Chae. Individual tooth segmentation from ct images using level set method with shape and intensity prior. *Pattern Recognition*, 43(7):2406 – 2417, 2010.

[GDDM14] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *2014 IEEE Conference on Computer Vision and Pattern Recognition*, pages 580–587, June 2014.

[GPAM+14] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.

[HAZATFS10] Mohammad Hosntalab, Reza Aghaeizadeh Zoroofi, Ali Abbaspour Tehrani-Fard, and Gholamreza Shirani. Classification and numbering of teeth in multi-slice ct images using wavelet-fourier descriptor. *International Journal of Computer Assisted Radiology and Surgery*, 5(3):237–249, May 2010.

[HKZ+17] Z. He, M. Kan, J. Zhang, X. Chen, and S. Shan. A fully end-to-end cascaded cnn for facial landmark detection. In *2017 12th IEEE International Conference on Automatic Face Gesture Recognition (FG 2017)*, pages 200–207, May 2017.

[HM09] Tobias Heimann and Hans-Peter Meinzer. Statistical shape models for 3d medical image segmentation: A review. *Medical Image Analysis*, 13(4):543 – 563, 2009.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *Proceedings of the 32Nd International Conference on International Conference on Machine Learning - Volume 37*, ICML'15, pages 448–456. JMLR.org, 2015.

[ISB] Grand Challenges in Dental X-ray Image Analysis challenge #2 computer-automated detection of caries in bitewing radiography. http://www-o.ntust.edu.tw/~cweiwang/ISBI2015/challenge2/index.html. Accessed: 2018-01-29.

[KB14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[LAE+16] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. *Lecture Notes in Computer Science*, pages 21–37, 2016.

[LBBH98] Yann LeCun, Leon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. In *PROCEEDINGS OF THE IEEE*, pages 2278–2324. IEEE, 1998.

[LPRM02] Bruno Lévy, Sylvain Petitjean, Nicolas Ray, and Jérome Maillot. Least squares conformal maps for automatic texture atlas generation. In *Proceedings of the 29th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '02, pages 362–371, New York, NY, USA, 2002. ACM.

[LZW+06] Hans Lamecker, Stefan Zachow, Antonia Wittmers, Britta Weber, Hans-Christian Hege, Barbara Elsholtz, and Michael Stiller. Automatic segmentation of mandibles in low-dose ct-data. *Int. J. Computer Assisted Radiology and Surgery*, 1(1):393 – 395, 2006.

[MKU+18] Philipp Marten Macho, Nadja Kurz, Adrian Ulges, Robert Brylka, Thomas Gietzen, and Ulrich Schwanecke. Segmenting Teeth from Volumetric CT Data with a Hierarchical CNN-based Approach. In Gary K. L. Tam and Franck Vidal, editors, *Computer Graphics and Visual Computing (CGVC)*. The Eurographics Association, 2018.

[MMH+17] Yuma Miki, Chisako Muramatsu, Tatsuro Hayashi, Xiangrong Zhou, Takeshi Hara, Akitoshi Katsumata, and Hiroshi Fujita. Classification of teeth in cone-beam ct using deep convolutional neural network. *Computers in Biology and Medicine*, 80:24 – 29, 2017.

[NALA08] D. E. Nassar, A. Abaza, Xin Li, and H. Ammar. Automatic construction of dental charts for postmortem identification. *Trans. Info. For. Sec.*, 3(2):234–246, June 2008.

[NBGS08] John Nickolls, Ian Buck, Michael Garland, and Kevin Skadron. Scalable parallel programming with cuda. *Queue*, 6(2):40–53, March 2008.

[PCZ15] Tomas Pfister, James Charles, and Andrew Zisserman. Flowing convnets for human pose estimation in videos. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1913–1921, 2015.

[PJSM15] R Pauwels, R Jacobs, S R Singer, and M Mupparapu. Cbct-based bone quality assessment: are hounsfield units applicable? *Dentomaxillofacial Radiology*, 44(1):20140238, 2015. PMID: 25315442.

[PMS12] Jahagirdar B Pramod, Anand Marya, and Vidhii Sharma. Role of forensic odontologist in post mortem person identification. *Dental research journal*, 9(5):522, 2012.

[PSBU16] Christian Payer, Darko Stern, Horst Bischof, and Martin Urschler. *Regressing Heatmaps for Multiple Landmark Localization Using CNNs*, volume 9901 of *Lecture Notes in Computer Science*, pages 230–238. Springer International Publishing AG, Switzerland, 10 2016.

[Rad17] J. Radon. Über die Bestimmung von Funktionen durch ihre Integralwerte längs gewisser Mannigfaltigkeiten. *Berichte über die Verhandlungen der Königlich-Sächsischen Akademie der Wissenschaften zu Leipzig, Mathematisch-Physische Klasse*, 69:262–277, 1917.

[RDGF16] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597, 2015.

[SFS06] William C. Scarfe, Allan G. Farman, and Predag Sukovic. Clinical applications of cone-beam computed tomography in dental practice. *Journal of the Canadian Dental Association*, 72/1, 2006.

[SWH05] Detlev Stalling, Malte Westerhoff, and Hans-Christian Hege. Amira: a highly interactive system for visual data analysis, 2005.

[SWT13] Yi Sun, Xiaogang Wang, and Xiaoou Tang. Deep convolutional network cascade for facial point detection. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2013.

[SZ14] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *CoRR*, abs/1409.1556, 2014.

[TGJ+15] Jonathan Tompson, Ross Goroshin, Arjun Jain, Yann LeCun, and Christoph Bregler. Efficient object localization using convolutional networks. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun 2015.

[WLT12] Y. Wang, B. Liu, and Y. Tong. Linear surface reconstruction from discrete fundamental forms on triangle meshes, 2012.

[YSNAA12] Anny Yuniarti, Anindhita Sigit Nugroho, Bilqis Amaliah, and Agus Zainal Arifin. Classification and numbering of dental radiographs for an automated human identification system. *TELKOMNIKA (Telecommunication Computing Electronics and Control)*, 10, 03 2012.