Stefan Eggenreich

Uncertainty Evaluation in Age Estimation using Deep Neural Networks

MASTER'S THESIS

to achieve the university degree of
Diplom-Ingenieur

Master's degree programme
Biomedical Engineering

submitted to

Graz University of Technology

Supervisor

Univ.-Prof. Dipl.-Ing. Dr.techn. Horst Bischof
Institute of Computer Graphics and Vision, Graz University of Technology

Dipl.-Ing. Dr.techn. Darko Štern
Department of Biophysics, Medical University of Graz

Graz, Austria, April 2020

Affidavit

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used.

The text document uploaded to TUGRAZonline is identical to the present master's thesis dissertation.

Date	Signature

Acknowledgments

First and foremost, I would like to thank Dr. Martin Urschler for giving me the opportunity to do my master thesis in this group. Furthermore, I would like to thank Dr. Darko Štern for his guidance, for the provided constructive feedback and the review of this thesis. I would also like to thank the members of the research group, especially DI. Christian Payer and DI. Franz Thaler for giving me a lot of support and helpful suggestions throughout this work. Lastly, I thank my family and my girlfriend for their moral support during my whole studies.

Contents

List of Figures

List of Tables

Abstract

Age estimation of humans has become an essential task in today's Clinical and Legal Medicine. Current Machine Learning (ML)-based multi-factorial age estimation methods show an accuracy, which is in line with established manual age estimation methods. However, the age estimation task underlies a substantial uncertainty caused by different biological development of bone structures in humans. In this thesis, we evaluated multiple Deep Learning (DL)-based methods regarding their ability to extract the label uncertainty, which arises from the biological variation in bone development.

We investigated the *epistemic uncertainty* by applying Ensembles, Monte Carlo (MC)-Dropout and Bayesian Convolutional Neural Networks (BCNNs) to our baseline and the multi-factorial age estimation task on 2D and 3D data. To extract the label uncertainty, we modified the model output. With our baseline experiments, we captured the theorized behavior of the *epistemic uncertainty* and the *aleatoric uncertainty* in literature. We were able to approximate a function, which describes behavior of the age information present in our age estimation data. The compared methods transfer the same behavior to the uncertainties in the age estimation task. The expressed uncertainty strongly depends on the dimensionality of the used data. The Ensemble can estimate the expected label uncertainty the most accurately and, at the same time, achieve the highest accuracy in the age estimation task.

We were able to show the theoretically described behavior of the uncertainties in our baseline experiment, which was also present in our age estimation experiments. In our baseline results, all methods, except the *MC-Dropout* method, were able to capture the expected uncertainty. The results in this thesis demonstrate that all methods can fit the data with an accuracy, comparable to current automated methods. We show that Ensembles are suited best for estimating the underlying label uncertainty in the investigated multi-factorial age estimation task. However, further investigation of the underlying problem is recommended in order to fully utilize the capabilities of the Bayesian framework.

Kurzfassung

Die Altersschätzung von Menschen ist eine essenzielle Aufgabe in der heutigen klinischen und rechtlichen Medizin. Derzeitig verwendete *ML*-basierende multifaktorielle Methoden der Altersschätzung weisen eine vergleichbare Genauigkeit zu etablierten manuellen Methoden auf. Der Altersschätzung unterliegt eine wesentliche Fehlerquelle, die durch die unterschiedliche Entwicklung von Knochenstrukturen in Menschen begründet ist. Im Rahmen dieser Diplomarbeit evaluierten wir verschiedene *DL*-basierende Methoden bezüglich deren Fähigkeit diese Unsicherheit, basierend auf der variierenden Entwicklung von Knochenstrukturen, zu schätzen. Wir untersuchten die epistemische Unsicherheit durch die Anwendung von Ensembles, *MC*-Dropout und *BCNNs* auf Baseline-Beispiele sowie 2D und 3D Daten zur Altersschätzung. Die aleatorische Unsicherheit wurde durch die Modifikation vom Output der Modelle gewonnen. In unseren Baseline-Experimenten konnte das theoretisierte Verhalten der epistemischen und aleatorischen Unsicherheit visualisiert werden, welches mit der Literatur übereinstimmt. Zusätzlich konnten wir die Funktion ermitteln, die das Verhalten der Altersinformation in unseren Daten zur Altersschätzung widerspiegelt.

In dieser Diplomarbeit konnten wir das theoretisch beschriebene Verhalten der Unsicherheiten durch unsere Baseline-Experimente aufzeigen, welches auch in der Altersschätzung sichtbar ist. Unsere Baseline-Experimente zeigten, dass alle Methoden, mit Ausnahme von *MC-Dropout*, die erwarteten Variationen in den Daten wiedergeben zu können. Unseren Resultaten ist zu entnehmen, dass alle Methoden in der Lage waren die Daten zur Altersschätzung zu lernen und Prognosen zu treffen, welche in deren Genauigkeit mit derzeitigen automatisierten Methoden übereinstimmen. Aus dieser Arbeit geht hervor, dass Ensembles die erwartete Label-Unsicherheit am besten wiedergeben kann und gleichzeitig die höchste Genauigkeit bezüglich der prognostizierten Labels besitzt. Um das Potential vom Bayesian Framework auszunutzen, werden jedoch noch weitere Untersuchungen auf diesem Gebiet notwendig sein.

1 Introduction

Contents

1.1 Age estimation

The estimation of age in living humans from radiological data has become an important topic in recent years. In clinical medicine, forensic anthropology, and legal medicine, age estimation is used to assess the unknown age of young subjects or discriminate adults from minors in cases where official personal documents are not available. The clinical medicine focuses on the stage of bone development, which is measured by the Biologial Age (BA), in children to assess an optimal time point of intervention to overcome endocrinological diseases [36] or plan orthopedic interventions. While the *BA* is of interest in the clinical medicine, in the legal medicine, the focus lies on the estimation of the age till birth, namely the Chronological Age (CA), based on the bone development in humans. The investigation of the *BA* development is mostly enabled by the ossification of epiphyseal gaps of human long bones [3]. The accuracy of the *CA* estimation, based on the *BA*, is limited by the biological variation of subjects of the same *CA*. The biological variation limits the accuracy of the estimation with an error of approximately one year [52]. The biological variability causes the estimation of the *CA* to be a difficult task for radiologists, in which the resulting estimate of the *CA*, which tend to be uncertain.

Established radiological methods for the estimation of *BA* are based on visual examination of the stage of bone development in terms of the ossification of hand bones from X-ray images [15]. The most commonly used way for manual X-ray based age estimation is

provided by the Greulich-Pyle Atlas method (GP) [15] or the Tanner-Whitehouse (TW2) approach [55]. The *GP* relies on atlases, which represent the bone development of subjects in discretized age intervals in terms of a mean or a median image. By visually identifying the accordance between the subject's radiological image and this atlas, an estimate of the *CA* is provided. Although the *GP* is fast and easy to use for the age estimation on the whole hand, it provides a low accuracy and high inter- and intra-rater variability. An improvement over the *GP* is provided by the *TW2* method. Instead of relying on atlases, the *TW2* bases the estimation of the *CA* on a scoring based system. In this system, scores are assigned to multiple defined maturity indicators in discrete stages of the individual hand bones. The single estimates are then combined in a weighted manner to calculate the final age estimate, which represents the age prediction with the highest probability [6].

While manual methods provide a good accuracy, they can be a tedious task for human experts. BoneXpert [57], an automated age estimation method, which is able to mimic the *TW2* scoring procedure, has been applied successfully.

In the process of the examination, the clinicians have to take the strong non-linear behavior of the ossification of different bones into account. Commonly, more distal bones, such as hand bones, finish growth earlier than proximal bones. In the case of hand bones, the development is finished at around 18 years. With the finished development of hand bones, the available age information saturates, which restricts the use of only hand bones in the age range from 13 to 18 years in clinical medicine. To extend this age range up to 25 years, which is highly relevant in legal applications, the Study Group on Forensic Age Diagnostics (AGFAD) [46] recommended a multifactorial approach by combining age-related information from hand bones in X-ray images with clavicle bones in chest Computed Tomography (CT) images and teeth in dental panoramic X-ray images.

The major drawback of using the established manual and automated X-ray based atlas methods for age estimation is the exposure of the subject of interest to ionizing radiation. Without a diagnostic purpose, the exposure of a human subject to ionizing radiation is not justifiable and is nowadays prohibited in most European countries. In legal applications, the interest mostly lies in the investigation of the age of healthy subjects, such as young asylum seekers, without proper identification papers. Therefore, research has recently focused on exchanging X-ray based techniques with Magnetic Resonance Imaging (MRI) as a non-invasive imaging modality for forensic age estimation [9, 18, 54, 56]. An additional advantage is that the acquired Magnetic Resonance (MR) images are usually volumetric, and multiple sites can be acquired within one imaging session. Volumetric data has the potential to hold much more information relevant for the age estimation, as compared to the 2D images acquired with X-ray imaging.

To eliminate the need to define discrete staging schemes for individual anatomical sites and subjective schemes for their fusion into a single age estimate, automatic, Deep Learning (DL)-based age estimation methods have already been developed [29, 49, 53, 57]. *DL* is an automated way of extracting features and learning representations from data, which as shown great success on various tasks in research fields like Computer Vision [47],

Physics [2] and Biology [16]. A recently proposed method by Štern et al. [52] for multi-factorial age estimation, based on Convolutional Neural Networks (CNNs), provides a solution for the automatic fusion of developmental information from 3D *MRI* scans of hand, clavicle and teeth into a *CA* prediction.

1.1.1 Types of uncertainties and their sources

Especially in the medical domain, where the lack of big datasets can negatively influence the performance of developed *DL*-based methods, it seems unjustifiable to treat the prediction of such a *DL* model without a measure of its predictive quality, or uncertainty. To be able to decide, based on the quality of the prediction, if certain samples need to be further examined by human experts, can help identifying problematic cases, reduce the workload on humans and can potentially reduce the overall error in the age estimation pipeline.

In current research, regarding the estimation of the predictive quality, the uncertainty is split into two main parts, namely the *aleatoric uncertainty* and the *epistemic uncertainty* [5, 7, 12]. The *epistemic uncertainty* gives insight into the quality of the model parameters regarding the data, which can be improved by increasing the number of data samples. This uncertainty is also often referred to as the *model uncertainty* [7, 21, 48].

The *aleatoric uncertainty*, also called *data uncertainty*, captures the noise of the information content, present in the data. It represents the deviation of the information present in a sample from the total population of samples. This uncertainty can only be reduced by improving the quality of the data itself. The *aleatoric uncertainty* can further be split into *homoscedastic* and *heteroscedastic uncertainty*. The *homoscedastic uncertainty* assumes a constant observation noise and stays constant, independent of the data. The *heteroscedastic uncertainty* depends on the noise in the input data and can vary according to the data [21].

In the task of the age estimation, the deviation of the *CA* from the *BA* can be interpreted as the label uncertainty present in the data. This label uncertainty falls into the category of the heteroscedastic *aleatoric uncertainty*.

1.1.2 Estimation of the Uncertainty with CNNs

Although automatic age estimation methods show favorable accuracy compared to human performance, the predictions of conventional *CNN*-based methods lack a measure of the predictive quality of the estimate. Therefore, the outlier in the data can lead to predictions that are wrong but indicate a high confidence, which can negatively influence the overall performance of the task [12, 37]. Due to the state-of-the-art results for various tasks, recent research has focussed on exploiting properties of *CNNs* to estimate the predictive quality with them.

While *CNNs* typically are deterministic systems, Gal and Ghahramani [11] proposed a method to model the *epistemic uncertainty* in *CNNs*, exploiting a commonly applied reg-

ularization method in current research. Kendal and Gal [21] built on the work of Gal and Ghahramani [12] and modified the output layer of a *CNN* in order to model the *aleatoric uncertainty* additionally to to the *epistemic uncertainty*. Those methods typically evaluate the *epistemic uncertainty* by sampling during inference time. Instead, Lakshminarayanan et al. [28] investigated the effects of Ensembles on the *epistemic uncertainty* and additionally modeling the *aleatoric uncertainty*.

1.1.3 Estimation of the Uncertainty with BCNNs

The described methods try to estimate the same underlying statistical process as in their statistical equivalent, the Bayesian Convolutional Neural Networks (BCNNs). The introduction of the Bayesian probability framework to the concept of machine learning adds a way to reason about the quality and the uncertainty in the provided predictions using mathematical tools [30]. While Bayesian Neural Networks (BNNs) are studied for many years now, only recently, they had been successfully applied to *CNNs*, since they often come with a computationally prohibitive cost [11, 34, 40].

1.2 Contributions

In this work, we evaluated multiple *DL*-methods regarding their capability to fit the data but mainly investigate their ability to represent the label uncertainty of the predictions in the multifactorial age estimation task. The compared methods encompass the Monte Carlo-Dropout (MC-Dropout), Ensembles and *BCNN* to model the *epistemic uncertainty*. Additionally, we applied the Gaussian output layer to each model, including conventional *CNNs*, to extract the *aleatoric uncertainty* from the data.

We used the late fusion method by Štern et al. [52], combining the age information of the different bone sites in our network architecture. To evaluate the contribution of age information from different sites of the human body, we evaluated our methods on hand bones only and combined hand bones with clavicles and teeth. All methods were conducted for the 3D imaging volumes and just the 2D middle slices of the images. In order to investigate the influence of the information content with the theorized *epistemic uncertainty* and *aleatoric uncertainty* we applied those methods to two baseline experiments on generated functions.

We evaluated the ability of the individual methods to capture the label uncertainty present in the data, which is the measure of uncertainty in this age estimation task. Furthermore, we investigated the ability of each method to separate the *aleatoric uncertainty* from the *epistemic uncertainty*, using the Gaussian output layer.

In this thesis, we found that the compared methods are capable of fitting the data with an accuracy, which is in line with the results archived by established automated Machine Learning (ML)-based age estimation methods. The individual methods are successful in capturing the label uncertainty, although it can be seen the best in Ensembles with 2D

data. Overall, the results showed a strong dependence on the dimensionality of the training data. Applying the Gaussian output layer, to split the uncertainty was only successful for the Ensemble method, while *MC-Dropout* and *CNN* only manages to do so with 2D data. The *BCNN* captures the label uncertainty but failed to perform this separation of the uncertainties, where the label uncertainty is included in the *epistemic uncertainty*. Finally, we showed that the Ensemble method showed the most substantial agreement to the baseline results, in terms of the expressed uncertainties, for the individual cases.

1.3 Outline

Chapter 2 provides the introduction into the topic of conventional- and Bayesian deep learning and the theoretical background as a basis for this thesis. Chapter 3 presents and explains the applied methods. Additionally, in this section discusses, the experimental setup, with the used network architecture, data augmentation, and additional used configurations in detail. An explanation and comparison of the archived results are shown in Chapter 5. In Chapter 5 our results are presented. The first part states on the quantitative results of the accuracy of the fit for the individual methods. In the second part, plots of the extracted uncertainty are depicted. In Chapter 6, a thorough discussion, comparing the results for the different evaluated methods is provided, followed by concluding this thesis in Chapter 7.

2 Machine Learning

Contents

The aim of Machine Learning (ML) is the automated detection of patterns in data without human intervention. By presenting data to such a *ML*-system it automatically extracts features from the data and builds a mathematical model, which can describe the data. This learned mathematical model can then be applied to yet unseen data in order to predict corresponding annotations for the data.

2.1 Deep Learning

2.1.1 Supervised learning

Supervised learning is a type of *ML* in which we train a model on a combination of input data x and corresponding labels y to learn a functional representation of information present in the data. This functional representation allows the mapping from the input to some target output. By learning a good representation of this information in the processed samples, the model is able to automatically predict labels for unseen samples.

2.2 Neural Network

The idea of Neural Networks (NNs) originate from the human biological nervous system, which consists of interconnected biological neurons. In *NNs*, artificial neurons mimic bio-

logical neurons. An artificial neuron represents the very basic idea of information processing via neurons in the human brain in a highly abstracted and simplified mathematical formalism [4]. This formulation can be derived from the simplest model of regression, namely linear regression.

Each of the $i \in N$ data samples x_i consist of a d-dimensional feature vector. The model of each artificial neuron consists of a weight vector $\mathbf{w} \in \mathbb{R}^D; \mathbf{w} = (w_1, ..., w_N)^T$. The single elements of the input vector x_i is weighted by the entries w_i of the weight vector $w \in \mathbb{R}$ and a bias w_0 is added. The sum notation can be rewritten in matrix form [4], as shown bellow

$$a(\mathbf{x}, \mathbf{w}) = w_0 + \sum_{i=1}^{N} w_i x_i = \mathbf{w}^T \mathbf{x} + \mathbf{w}_0 \tag{2.1}$$

The output $a(\mathbf{x}, \mathbf{w})$ of this function is then mapped with an activation function $h(\cdot)$ to the output that becomes the input to the next neuron, also called unit, as shown in the following equation

$$y(\mathbf{x}) = h(a(\mathbf{x}, \mathbf{w})). \tag{2.2}$$

2.3 Feedforward Neural Networks

Feedforward *NNs* are extremely important parametric Deep Learning (DL) models, which layed the foundation for many of the nowadays most successfully applied methods in *DL* [13]. The aim of feedforward *NNs* is to approximate a function f^*. This function maps the input $\mathbf{x}$ to some output value y, using this learned function $y = f^*(\mathbf{x}; \theta)$. By processing data $\mathbf{x}$, the network learns the parameters θ, that define the function f in order to minimize a given optimization target.

The output of such a feedforward *NNs* is produced, by passing the input through a series of various functions $f^{(l)}$, which define f. In *NNs*, those functions are also called layers, where each layer is represented by numerous artificial neurons $f^{(l)}(\mathbf{x}_{l-1}; \boldsymbol{\theta}_l)$. Layers, which do create the desired output, are called hidden layers [13]. While the arrangement of those layers can basically be arbitrary, in feedforward *NNs* they are connected in a chain to form an acyclic, directed graph

$$f(\mathbf{x}) = f^{(3)}(f^{(2)}(f^{(1)}(\mathbf{x}))). \tag{2.3}$$

2.3.1 Activation Function

Linear transformations lack the complexity needed to learn functional representations of high dimensional features. To enable *NNs* to even learn complex functions, the linear transformation of the activation function $h(\cdot)$ is replaced by a differentiable, non-linear function.

One of the most successful choices as an activation function, in current research, is the Rectified Linear Unit (ReLU) function. The *ReLU* activation function represents

a piecewise linear function, defined by Eq. (2.4) with the visualization in Fig. 2.1. Although this function is not differentiable in every point, in the floating point accuracy caused the evaluation typically to not be exactly at $a = 0$. Therefore this function is commonly used in gradient-based optimization methods [13] Compared to other, more complex functions, such as the Selective Linear Unit (SeLU) [25] or a variation of *ReLU* named leaky-*ReLU* [33], the *ReLU* provides faster convergence speed and is also more stable in terms of convergence behaviour.

$$h(a) = \max(0, a) = \begin{cases} a \leq 0 & 0 \\ \text{else} & a(\mathbf{x}, \mathbf{w}) \end{cases} \tag{2.4}$$

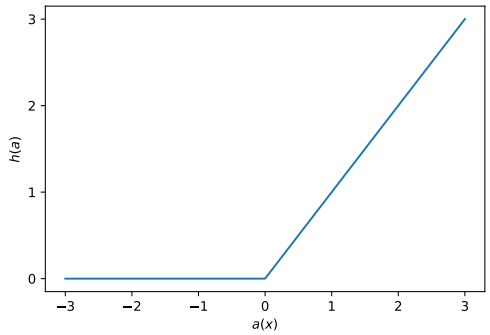


Figure 2.1: Visualization of the Rectified Linear Unit activation function.

Typically, the *ReLU* is only applied to the hidden units of a *NN*. For the units of the output layer, the activation function has to be modified, according to the task to solve. The most commonly encountered tasks are classification and regression. In case of regression, $g(\cdot)$ performs an identity transformation of the activation of the output units [4].

2.3.2 Optimization

In 1986 Rumelhart [44] proposed a computationally efficient algorithm to iteratively adjust the model weights, depending on their individual contributions to the error. *Error backpropagation*, or also called *backprop* in short, refers to the method for computing the gradients of an abitrary function. In *NNs*, those calculated gradients are then used with another algorithm, with one of the simplest being *stochastic gradient descent*, to perform learning of the correct model weights [4, 13].

The process of optimizing a *NN* can be split into a three step process, which described bellow.

In the forward-propagation step, the input is passed through the network starting from the input layer. Single neurons are activated based on their weights and their used activation function. An objective function $\mathcal{F}(\cdot, \cdot)$ is used to calculate the deviation of the

output $\hat{y}$ of the network from their corresponding groundtruth labels y. In the second step, the gradients of the objective function $\mathcal{F}$ with respect to the single model weights $\mathbf{w}$ are calculated at the current iteration τ, shown in the next equation

$$\nabla_{\mathbf{w}^{(\tau)}} \mathcal{F}(\hat{y}, y, \mathbf{w}^{(\tau)}) = \frac{\delta \mathcal{F}(\hat{y}, y, \mathbf{w}^{(\tau)})}{\delta w_{i,j}}.$$

The last step, the gradient descent step, updates the corresponding model weights based on the computed gradients

$$\mathbf{w}^{(\tau+1)} = \mathbf{w}^{(\tau)} - \eta \nabla_{\mathbf{w}^{(\tau)}} \mathcal{F}(\hat{y}, y, \mathbf{w}^{(\tau)}),$$

where $\eta > 0$ is the learning rate and $\tau + 1$ indicates the next iteration.

In this formulation, *gradient descent* is computationally very inefficient, since the weights are updated after all samples N of the training set were processed. By performing the weight update after mini-batches with batch size $K < N$, or even after single samples, the performance of this method can be increased significantly. This method is called Stochastic Gradient Descent and has shown an improvement in performance and convergence behaviour [4, 31]. A further improvement over Stochastic Gradient Descent can be gained by introducing a momentum in the process of the weight update [42, 44], shown in the following equation

$$\mathbf{w}^{(\tau+1)} = \beta \mathbf{w}^{(\tau-1)} - \eta \nabla_{\mathbf{w}^{(\tau)}} E(\hat{y}, y, \mathbf{w}^{(\tau)}). \tag{2.5}$$

The momentum remembers the weight update of the previous iteration and scales it by the momentum paramter β. With that the susceptibility to noise of the gradient, introduced in the current batch can be reduced. This enforces consistent gradient directions and often faster convergence speeds [13]. Kingma and Ba [22] proposed a stochastic optimization algorithm called *Adam* that is derived from adaptive moment estimation. *Adam* has shown to improve the convergence speed and stability of the computed gradients over standard Stochastic Gradient Descent.

2.4 Convolutional Neural Network

In fully-connected feedforward *NN* all neurons of the previous layer are connected to all neurons of the next layer. Therefore, especially deep architectures are made up of millions of neurons and require a significant amount of memory. The backpropagation of the error, in such architectures, can become complicated, if no impossible. In 1989 LeCun et al. [32] introduced Convolutional Neural Networks (CNNs), which leverage the fact that neighboring pixels share more information with each other, than pixels which are further apart. In the discrete domain in the 2D case, a weight kernel $K \in \mathcal{R}^{m \times n}$ is convolved with the input features $I \in \mathcal{R}^{i \times j}$ of the layer, where the kernel is comprised of the weights

of the layers according to the bellow equation

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{2.6}$$

The convolutional operation can be extended for the 3D case, by using the kernel $K \in \mathcal{R}^{m \times n \times o}$, as given in following equation

$$S(i,j,k) = (I * K)(i,j,k) = \sum_m \sum_n \sum_o I(i-m, j-n, k-o)K(m,n,o). \tag{2.7}$$

CNNs effectively perform weight sharing over multiple different locations of the input image, which significantly reduces the number of model parameters, compared to a feedforward *NN*. Using a kernel smaller in size than the input creates sparse interactions, which in combination with parameter sharing, enable a certain translation equivariance. This translation equivariance has the effect that the same feature produces the same output, independent of its location in the image.

CNNs outperform conventional *NNs* in terms of performance and additionally have less required memory needed to archive an equivalent performance. Usually, *CNNs* consist of a cascade of convolutional and pooling layers, whereby each convolutional layer extracts local features at different levels of image resolution. The combination of the extracted features allow the detection of higher order features [13].

2.4.1 Pooling Layer

Pooling layers are sub-sampling layers with the intention to reduce the size of the input tensor by retaining certain local pixel statistics of the input. The most commonly used pooling operation is Max-pooling [60], which only retains the pixel with the highest value within the sliding window. Additionally to the properties of convolutions, the pooling operation adds a certain translation , and up to a degree, also rotational invariance [13].

2.4.2 Fully Connected Layer

In *CNNs* fully connected layers are commonly used in the last few layers before the output layer of the network architecture. Fully connected layers are used for the final decision making by summarizing the activations from all previous layers. The input tensor into the fully-connected layer is flattened, thus reshaped to a one-dimensional vector. Consequently an input tensor with batch size B, number of channels C, width W and height N and of size $B \times C \times M \times N$ is reshaped into size $B \times C * M * N$. In fully connected layers all neurons of a layer are connected all the neurons in the next layer. Commonly the fully connected layer successively reduces the number of features until the output layer.

2.4.3 Dropout Regularization

Due to the introduced non-linearities, deeper *CNNs* are prone to memorize the training data and are not able to generalize to the whole domain, explained by the training data. This effect is called overfitting and becomes critical when dealing with a small amount of data samples. *Dropout*, proposed by Srivastava et al. [51] is a regularization technique designed to reduce the effect of overfitting. During training, a *dropout* layer randomly deactivates neurons with a dropout probability p. Therefore, the model can not simply relie on strong activations from single neurons. With *dropout*, a regularization effect is obtained, which can also be interpreted as adding noise to the hidden units [50]. In *CNNs*, *dropout* is commonly only applied to the intermediate outputs of fully-connected layers in the form of additional *dropout* layers, since *dropout* directly after the convolutional layers has shown to decrease the evaluation performance drastically [11].

2.4.4 Batch Normalization

The normalization of the input features has shown to improve the overall performance of the network and can increase its convergence speed. This data normalization is usually only possible before passing the input into the network [13].

Ioffe and Szegedy [19] extended on this idea and proposed a method called *Batch normalization* which can be applied to the outputs of the intermediate layers of the model. *Batch normalization* can be seen as a layer wise linear transformation of the output of the previous layer based on statistics of the current batch, to ensure a consistent distribution of the activations for all mini-batches. The batch normalization layers retain a moving average of the μ and σ of the activation of each unit.

During training, each mini-batch is normalized with the μ and σ of the current mini-batch according to the following equation

$$H^* = \frac{H - \mu}{\sigma}, \tag{2.8}$$

and during evaluation, the calculated moving average over the seen samples is used to normalize the data with Eq. (2.8) [13, 19].

2.4.5 Pobabilistic Layers

Bayesian Convolutional Neural Networks (BCNNs) require a modification of the conventional convolutional and dense layers in order to incorporate the bayesian framework. Based on the work of Bundell et al. [5], for both the kernel and bias, a posterior and a prior distribution is defined. The prior distributions $p_{(kernel,bias)}(\omega)$ remains constant and describes the assumption about the data distribution. The posterior distributions $p_{(kernel,bias;\theta)}(\omega|x)$ is parameterized by a Gaussian distribution, thus $\theta = \{\mu, \sigma\}$. During optimization, the parameters θ of the approximate posterior distribution is learned in

order to approximate the true posterior distribution of the presented data.

An example for such statistical layers are called Reparameterization layers [23] and are provided in the *TensorFlow Probability* library [8]. The Reparameterization layers, by definition, implement the idea of reparameterization in order to allow backpropagation through the network (Section 3.1.3). The implementation of Reparameterization layers averages the activations over multiple sets of weights, sampled from their trained approximate posterior distributions $q(\omega|\theta)$. In practice the Kullback Leibler (KL) loss is calculated layer wise. The *KL* loss of each layer contributes in sum to the Evidence Lower Bound (ELBO) loss in Eq. (3.7). By sampling model weights $w \sim p(w|\theta)$ and biases $b \sim p(b|\theta)$, probabilistic convolutional and dense layers can be applied as described in their deterministic counterparts in Section 2.4.

3 Methods

Contents

In the following sections, the different evaluation methods of the uncertainties are discussed in further detail, and the experimental setup is described.

3.1 Bayesian Deep Learning

Bayesian Neural Networks (BNNs) are the counterpart to their deterministic version, the Neural Networks (NNs). The idea behind Bayesian Deep Learning (DL) is based on the well known statistical model, the Gaussian Process. A Gaussian Process is a non-parametric approach, that tries to find a distribution of possible functions $f(x)$. A Gaussian Process consists of a set of random variables, which formulate a multivariate Gaussian distribution. An assumed prior distribution, which restricts the set of possible functions, is updated by observing data. For sufficiently enough data samples, a model posterior over possible functions is generated, which describes functions, that could have generated the seen data [43].

With this intention in mind, we introduce Bayesian statistics into the concept of *BNNs*, by putting a distribution over function f in the form of a prior distribution $p(f)$. Given the corresponding data and label pairs $\{x_i, y_i\}_{i=1}^{N}$ we can define a function f, parameterized by the model weights w, which maps the input to an output $y = f(x)$. With this alternation

the model is enabled to capture the *epistemic uncertainty* information directly in it's individual distributions.

Using the Bayesian Theorem, we can calculate the posterior distribution $p(f|x, y)$ as given in the equation bellow

$$p(f|x, y) = \frac{p(y|x, f)p(f|x)}{p(y|x)}, \tag{3.1}$$

where the likelihood is given by $p(y|x, f)$.

The predictive probability the combination of unseen data x^* and labels y^* is given in the following equation

$$p(y^*|x^*, x, y) = \int p(y^*|f^*)p(f^*|x^*, x, y)df^*. \tag{3.2}$$

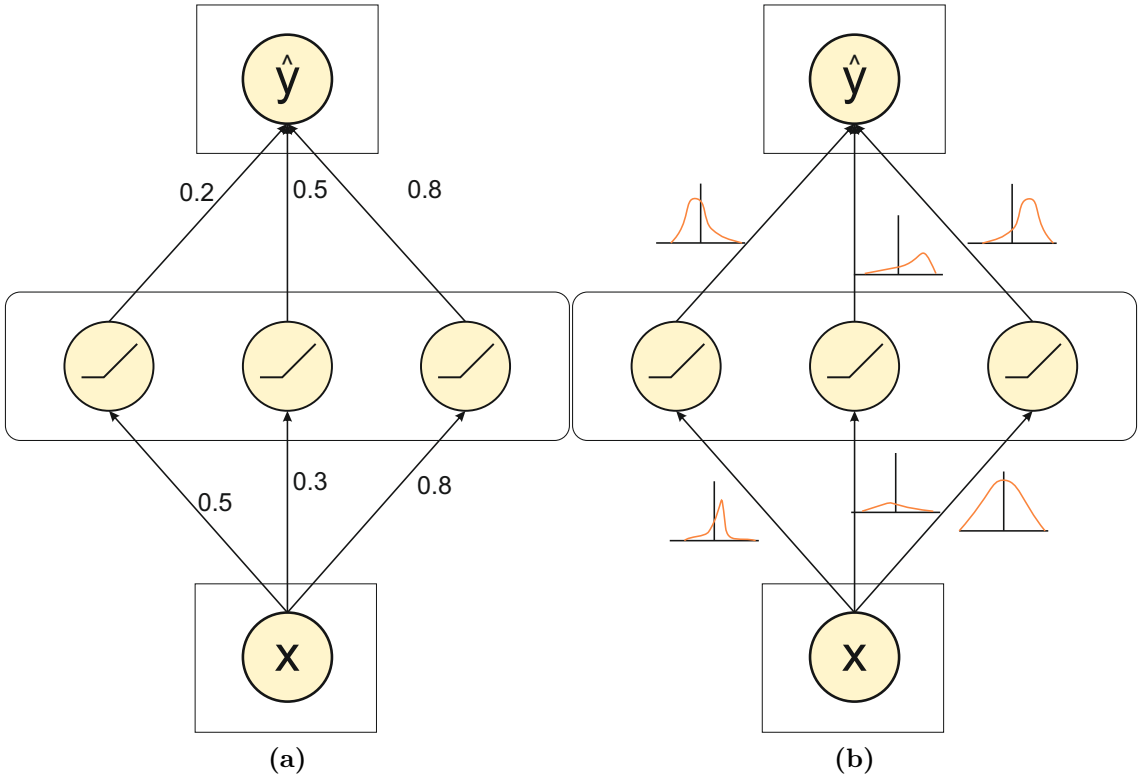


Figure 3.1: An exemplary visualization of the difference between a conventional CNN and a BCNN. (a) The weights of a CNN are represented by single values. (b) The weights are replaced by distributions.

3.1.1 Variational Inference

Equation (3.2) can possess an analytical solution in only very rare cases, but is typically intractable in praxis, since it requires the integration over all possible functions $f \in \Omega$ in the parameter space. This inhibits the calculation of an exact solution of the posterior distribution $p(y^*|x^*, x, y)$ [48].

Laplacian methods from MacKay [35], and Hamiltonian dynamics from Neal [39] have been proposed to approximate the true posterior distribution. Both methods either provide a poor approximation or are computational prohibitive [27] and can therefore not be used efficiently in Bayesian Convolutional Neural Networks (BCNNs).

A nowadays commonly used method to overcome intractable integrals in Bayesian *DL* is Variational Inference. Variational Inference aims to replace those intractable integrals with computable approximations. The predictive probability calculated in Eq. (3.2), depends on the data x and the corresponding labels y from the training set, which is not available during evaluation. By conditioning the model on a finite set of random variables ω, the predictive likelihood is decoupled from those data points. With that, it can be assumed that the random variables can represent the population, which turns them into sufficient statistics for the approximation model [48], as shown in the following equation

$$p(y^*|x^*, x, y) = \int p(y^*|f^*)p(f^*|x^*, \omega) \underbrace{p(\omega|x, y)}_{\text{Intractable}} df^* d\omega. \tag{3.3}$$

The Eq. (3.3) still is intractable. Variational Inference now replaces the intractable posterior distribution with a simplified, computable variational distribution $q(w)$, which should approximate the posterior as close as possible. With this replacement the approximate predictive distribution can be rewritten as shown in the equation bellow

$$p(y^*|x^*, x, y) \approx \int p(y^*|f^*)p(f^*|x^*, \omega)q(\omega)df^* d\omega. \tag{3.4}$$

To approximate the true posterior with the variational posterior, we use the Kullback Leibler (KL) divergence as a measure of the accordance between both distributions $q(\omega)$ and $p(\omega|x, y)$. The definition of the *KL* divergence in the continuous domain is given in the equation bellow

$$KL[q(\omega)||p(\omega|x, y)] = \int_{-\infty}^{\infty} q(\omega) \log \frac{q(\omega)}{p(\omega|x, y)} d\omega. \tag{3.5}$$

3.1.2 Bayes by Backprop

To archive a good representation of the true posterior distribution by the variational distribution, we aim to maximize the accordance between them. Blundell et al. [5] proposed a Variational Inference method named *Bayes by Backprop* in which the given problem in Eq. (3.4) can be converted in an optimization problem.

The variational posterior is paramterized by the paramters θ, which can be updated to increase the similarity in Eq. (3.5). The optimal paramters $\hat{\theta}$ are given by the following optimization problem

$$\begin{aligned}
\theta^* &= \arg\min_{\theta} KL[q(\omega|\theta)||p(\omega|x,y)] \\
&= \arg\min_{\theta} \int q(\omega|\theta) \log \frac{q(\omega|\theta)}{p(\omega|x,y)} \\
&= \arg\min_{\theta} \int q(\omega|\theta) \log \frac{q(\omega|\theta)}{\frac{p(\omega)p(x,y|\omega)}{p(x,y)}} \\
&= \arg\min_{\theta} \int q(\omega|\theta) \log \frac{q(\omega|\theta)}{p(\omega)} d\omega - \int q(\omega|\theta) \log p(x,y|\omega) d\omega + \int q(\omega|\theta) \log p(x,y) d\omega \\
&= \arg\min_{\theta} KL\left[q(\omega|\theta)||p(\omega)\right] - \mathbb{E}_{q(\omega|\theta)}\left[\log p(x,y|\omega)\right] + \int q(\omega|\theta) \log p(x,y) d\omega.
\end{aligned} \tag{3.6}$$

The integral in the last term in Eq. (3.6) reduces to $p(x,y)$ which is constant, and therefore, can be neglected in the optimization process. This gives the objective function

$$\mathcal{F}(x,y,\theta) = KL\left[q(\omega|\theta)||p(\omega)\right] - \mathbb{E}_{q(\omega|\theta)}\left[\log p(x,y|\omega)\right], \tag{3.7}$$

which is widely known as the *variational free energy* or Evidence Lower Bound (ELBO) loss [20, 40, 45, 59]. The optimization of this cost function provides an inherent regularization effect, which is able to reduce the likelihood of overfitting to the data [5].

3.1.3 The Reparameterization Trick

Bayes by backprop aims to use the *backpropagation* algorithm to update the distribution parameters $\boldsymbol{\theta}$ in *BCNNs* and requires the calculation of the partial derivatives the established loss function (Eq. (3.7)), as shown bellow

$$\nabla_\theta \mathcal{F}(x,y,\theta) = \nabla_\theta KL\left[q(\omega|\theta)||p(\omega)\right] - \nabla_\theta \mathbb{E}_{q(\omega|\theta)}\left[\log p(x,y|\omega)\right]. \tag{3.8}$$

Due to the usage of parameterized distributions for the weights, the nodes are now stochastic. For simplicity, we define the output of a stochastic variable as

$$z \sim q(w|\theta) = \mathcal{N}(\mu, \sigma), \tag{3.9}$$

and

$$f_\theta(z) = \log p(x,y|z). \tag{3.10}$$

Although, the usage of *backpropagation* with this cost function is theoretically possible, its use for backpropagation would be not feasible. The main reason for infeasibility arises

from the gradient of the second term of Eq. (3.8), which is shown in the following equation

$$\begin{aligned}\nabla_\theta \mathbb{E}_{q(\omega|\theta)}\left[f_\theta(z)\right] &= \nabla_\theta \left[\int_\Omega q(z|\theta) f_\theta(z) dz\right] \\ &= \int_\Omega \nabla_\theta \left[q(z|\theta) f_\theta(z)\right] dz \\ &= \int_\Omega f_\theta(z) \nabla_\theta q(z|\theta) dz + \int_\Omega p(z|\theta) \nabla_\theta f_\theta(z) dz \\ &= \int_\Omega f_\theta(z) \nabla_\theta q(z|\theta) dz + \mathbb{E}_{q(z|\theta)}\left[\nabla_\theta f_\theta(z)\right].\end{aligned} \tag{3.11}$$

The calculated gradients in Eq. (3.11) are calculated based on samples from $q(z|\theta)$. Therefore the calculated gradients can have discontinuities, which do not represent the true underlying gradients. This causes the optimization procedure to be rendered impossible, since the contribution of individual parameters cannot be distinguished (see Fig. 3.2a).

The Reparameterization trick was introduced by Kingma et al. [23] in order to overcome the problem of this high variance, and enable the efficient application of *backpropagation* with *BCNNs*. Instead of calculating the partial derivative of $\mathbb{E}_{q(\omega|\theta)}\left[f_\theta(z)\right]$, using the stochastic node, first we reformulate the sampling procedure, by introducing a new random variable $\epsilon \sim \mathcal{N}(0,1)$. With that, we can define a deterministic function

$$g_\theta(x, \epsilon) = \mu + \sigma \cdot \epsilon = \mathcal{N}(\mu, \sigma), \tag{3.12}$$

from which we generate the samples

$$z = g_\theta(\epsilon, x). \tag{3.13}$$

Instead of sampling from $q(w|\theta)$ directly, we sample from the, now decoupled stochastic variable ϵ and apply it to the now deterministic variables, μ and σ. This way, we can substitute

$$\mathbb{E}_{q(\mathbf{z}|\theta)}\left[f(\mathbf{z}^{(i)})|\theta)\right] = \mathbb{E}_{q(\epsilon)}\left[f(g_\theta(\epsilon, \mathbf{x}^{(i)}))\right], \tag{3.14}$$

through which the gradient of the expectation in Eq. (3.8) can be expressed as the expectation of the gradient, as shown in the equation bellow

$$\begin{aligned}\nabla_\theta \mathbb{E}_{q(\mathbf{z}|\theta)}\left[f(\mathbf{z}^{(i)})|\theta)\right] &= \nabla_\theta \mathbb{E}_{q(\epsilon)}\left[f(g_\theta(\epsilon, \mathbf{x}^{(i)}))\right] \\ &= \mathbb{E}_{q(\epsilon)}\left[f(\nabla_\theta g_\theta(\epsilon, \mathbf{x}^{(i)}))\right],\end{aligned} \tag{3.15}$$

in order to guarantee, the gradient to be a valid solution (see Fig. 3.2b). Furthermore, by assuming that $g_\theta(\epsilon, x)$ is differentiable, Monte Carlo (MC) integration can be used as an

approximation of the resulting expectation [23], given in the following equation

$$\mathbb{E}_{q(\epsilon)}\left[f(\nabla_\theta g_\theta(\epsilon, \mathbf{x}^{(i)}))\right] \approx \frac{1}{T}\sum_{t=1}^{T} f(\nabla_\theta g_\theta(\epsilon^{(t)}, \mathbf{x}^{(i)})) \tag{3.16}$$

Averaging over $l \in L$ samples of $\epsilon^{(l)}$, the random noise variable ϵ can be considered to be constant, since the error introduced by it will be integrated out, if a large dataset is used [23, 24].

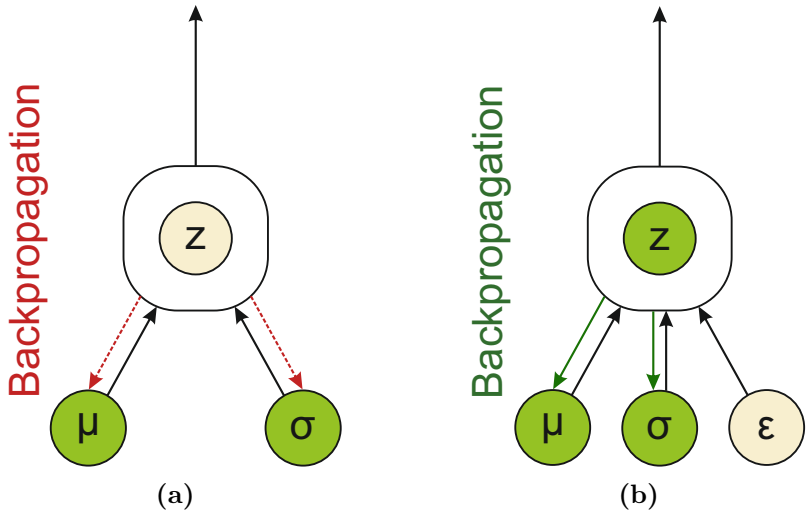


Figure 3.2: Schematic representation of the effect of reparamterization on the output of a stochastic node z. (a) The calculated gradients with respect to the variables μ and σ indicates a high variance and therefore are inaccurate. (b) Applying reparameterization to the stochastic node, the noise ϵ can be separated from the node, which now becomes deterministic. The calculated gradients are accurate, which allows the measurement of the individual contribution. This allows backpropagation of the error through the node to both μ and σ.

3.2 Monte Carlo-Dropout

As already described in Section 3.1, exact Bayesian modeling is not possible due to computational restraints. Therefore, alternative ways to estimate the uncertainty are required instead. While sampling-based approaches, to approximate the exact posterior distribution, are theoretically plausible, in praxis, they are computationally very demanding for current hardware. In 2015 Gal and Ghahramani [12] proposed a method termed Monte Carlo-Dropout (MC-Dropout), which can be used with conventional Convolutional Neural Networks (CNNs). *MC-Dropout* couples the widely used regularization method of *dropout* [51] with a sampling-based approach at inference time to approximate the intractable variational posterior distribution in Eq. (3.2). *Dropout* has proven to be an

effective way to prevent networks from overfitting to the training data, and when used in the context of Bayesian reasoning, it can offer additional insight into the uncertainty that comes with the prediction provided from a model. *MC-Dropout* not only offers additional insight into the quality of the prediction, but it can also increase the performance over the standard Frequentist approach [11, 12].

The application of *dropout* effectively minimizes the *KL* divergence between an approximate distribution and the posterior of an in-depth Gaussian process [12]. Therefore, the authors state, *MC-Dropout* can be used as an approximation of the Gaussian Process, thus approximating *BCNNs*. *MC-Dropout* avoids the intractable integral in Eq. (3.3), by averaging the predictions over T stochastic forward passes, as shown in Eq. (3.17) [11]. With that, the predictive performance of the model can be evaluated.

$$p(y^*|x^*, x, y) \approx \int p(y^*|f^*)p(f^*|x^*, \omega)p(\omega)df^*d\omega \approx \frac{1}{T}\sum_{t=1}^{T} p(y^*|x^*, \hat{\omega}_t) \tag{3.17}$$

Dropout causes for each of the $t \in T$ stochastic forward passes a different set of network weights to be sampled from the approximative variational posterior distribution $w_t \sim q(w)$. Concerning the regression task, the *epistemic uncertainty* of sample x_i is expressed in the variance over T stochastic forward passes in following equation

$$\sigma_i^2 = Var(\{f(x_i; w_t)\}_{t=1}^{T}). \tag{3.18}$$

According to Gal and Ghahramani [11], the implementation of a *CNN*, which applies *MC-Dropout*, is equivalent to performing dropout after every convolutional layer and dense layer. Since the pooling operation is comparable to a non-linear operation, dropout has to be performed before a pooling layer [11].

3.3 Uncertainty predictions

To extract the the *epistemic uncertainty* of the data sample x_n in Ensembles, the *MC-Dropout* method and *BCNNs* we calculate the variance of the predictions $\hat{y_n}$ of T forward passes of the input sample x_n as shown in the equation bellow

$$\sigma_n^2 = \text{Var}\left(\{\hat{y}_n\}_{t=1}^{T}\right) \tag{3.19}$$

3.3.1 Gaussian Output Layer

Neither conventional *CNNs*, nor Bayesian *CNNs* capture the *aleatoric uncertainty* by default and require some modifications to the output of the model.

To overcome this problem, we extend the output of a network by a second prediction [21]. This second prediction expresses the variability of information, thus the

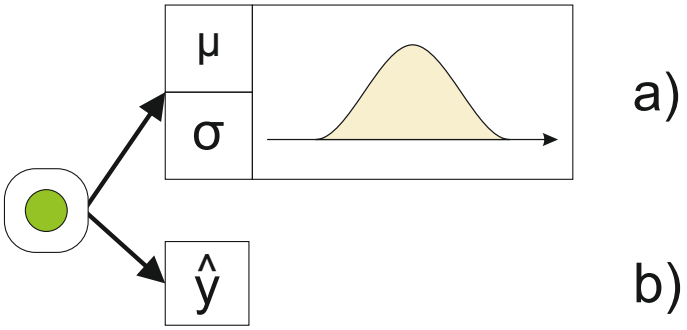


Figure 3.3: A visualization of the possible formulations of the output. *a)* The model predicts both μ and σ at the output which are used to instantiate a normal distribution. This approach is refered to as *Gaussian output layer* in this thesis. *b)* The predicts only a single value $\hat{y}$.

aleatoric uncertainty. in the data sample x, directly at the output. We formulate the cost function, such that both predictions of the model parameterize a Normal distribution $\mathcal{N}(y; \mu(x,\theta), \sigma(x,\theta))$, of which we aim to maximize the probability for a given the groundtruth label y of the corresponding data sample $\boldsymbol{x}$. The optimal paramters of the model are given by

$$\hat{\theta} = \arg\max_{\theta} \mathcal{N}(y; \mu(x,\theta), \sigma(x,\theta)). \tag{3.20}$$

Moreover, we can increase the numerical stability by minimizing the negative log-probability, namely the Negative Log Likelihood of Eq. (3.20) instead

$$\hat{\theta} = \arg\min_{\theta} -\log \mathcal{N}(y; \mu(x,\theta), \sigma(x,\theta)). \tag{3.21}$$

The objective function for optimizing a network with a Gaussian output layer, therefore, is equivalent to

$$\begin{aligned} \mathcal{F}(x,\theta) &= -\log \mathcal{N}(y; \mu(x,\theta), \sigma(x,\theta)) = \\ &= -\log\left(\frac{1}{\sqrt{2\pi}\sigma(x,\theta)} \exp -\frac{(\mu(x,\theta)-y)^2}{2\sigma(x,\theta)^2}\right) \\ &= -\frac{\log \sigma(x)}{\sqrt{2\pi}} + \frac{(y-\mu(x))^2}{2\sigma(x)^2} \end{aligned} \tag{3.22}$$

Expressing the variability at the output of the network enables the use of standard supervised learning, in order learn the variability σ present in the data. With this method, no additional labels of the uncertainty are required, which improves the applicability for many different tasks. A Gaussian output layer can be used for *CNNs* and *BCNNs* to extract the sample-wise *aleatoric uncertainty*.

3.4 Ensemble Model Combination

In the work of Srivastava et al. [51], the authors noted that the regularization method dropout could be interpreted as an Ensemble model combination, which often can improve the performance by averaging the predictions. The large amount of possible non-linear parameter configurations that contribute to the representation of the training data gives single networks, which are trained on the same tasks, a high variety around a true optimum. A reduction of the variability of the single voters can be archived by using an Ensemble of models. This way, predictions of individual models are combined to get a single estimate.

Multiple different approaches using the concept of Ensemble model combination exist. Randomization-based approaches, like random forests, rely on independent branches, which are all trained in parallel. The outcomes of the single branches are then combined and used as the Ensemble's prediction. Another approach, named boosting in the literature, trains graphs sequentially on grouped sets of images [28]. The simplest approach involves training several identical models, which deviate from each other due to the initialization scheme of the weights. Each model extracts different features and aspects of the input data. Further diversification of the single models can be obtained by additionally applying dropout during training. Ensemble model combination provides a measure against overfitting due to the combination of multiple votes. Averaging of the single predictions of the same sample has shown an improvement of performance and reduced the variance between the predictions themself [26].

While conventional Ensembles are applied to various *CNN* architectures with success and have shown to be able to increase the overall predictive performance [28], those models do not consider the uncertainty of the predictions in any way. Lakshminarayanan et al. [28] proposed an approach that requires minor changes to standard *CNNs* and is argued to produce a high-quality predictive estimate of the *epistemic uncertainty* as well as the *aleatoric uncertainty*. Similarly to Kendall and Gal [21] the output of the model is extended, as a Gaussian output layer, described in Section 3.3.1. The predictions of the individual models $m \in M$ are averaged to compute $\overline{\mu}(x) = M^{-1} \sum_m^M \mu_m(x)$ and $\overline{\sigma}(x) = M^{-1} \sum_m^M \sigma_m(x)$ which are then used in the loss function, described in Eq. (3.22).

3.5 Loss Functions

In the case of a conventional *CNN*, we used the ℓ_2 norm of the difference of the prediction $\hat{y}$ and the groundtruth y plus a standard weight decay as a regularization term. Our objective function, in this case, is given by

$$\mathcal{F}(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{B} ||\hat{\boldsymbol{y}}(\boldsymbol{x}) - \boldsymbol{y}||^2 \tag{3.23}$$

where B is the size of the current mini-batch. Using the Gaussian output layer, we utilize the Negative Log Likelihood in Eq. (3.22) as a loss function.

To optimize the *BCNN*, we minimize the *ELBO* loss in Eq. (3.7). We used a layer-wise *KL*-loss Eq. (3.5), which contributes in sum to the *ELBO*-loss and calculate the data term using the Negative Log Likelihood from Eq. (3.22). In experiments with *BCNNs*, which do not express the variability σ at the output, we assume the variance of each sample to be a constant of $\sigma = 1$. This assumption simplifies the Negative Log Likelihood in Eq. (3.22) according to Eq. (3.24) to effectively a scaled ℓ_2 norm up to a constant of $\log 2\pi$, that can be neglected in optimization.

$$\begin{aligned} -\log p(y|x) &= -\log \mathcal{N}(y(x); \mu, \sigma^2)|_{\sigma^2=1} \\ &= -\log \left(\frac{1}{\sqrt{2\pi}} \exp -\frac{(\hat{\mu} - y)^2}{2} \right) \\ &= -\frac{1}{2} \left(-\log 2\pi - (x - \mu)^2 \right) \end{aligned} \tag{3.24}$$

4 Experimental setup

Contents

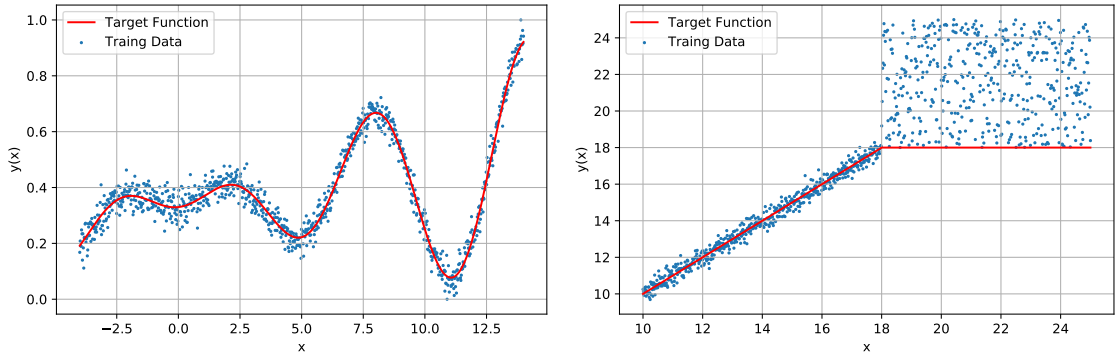
4.1 Baseline Experiments

To visualize the impact of the data on the *aleatoric uncertainty* and the *epistemic uncertainty*, we conducted baseline experiments on a one dimensional regression task. We sample 1000 uniformly distributed samples from two different functions (see Fig. 4.1).

The first function is shown in Fig. 4.1a. In this function, we added noise by sampling values from a standard normal distribution. The second function is given in Fig. 4.1b which should approximate a function, which describes the age information content, in our hand bones. We added noise sampled from a standard normal distribution for samples $x < 18$ and sampled a uniform distributed noise between 18 and 25 for samples $x > 18$.

For testing, we sample the model output at 1000 uniformly distributed values of x in the range between -10 and 20 for the sinusoidal baseline and $10 < x < 25$ for the approximative function. We use $T = 20$ Monte Carlo (MC)-samples to calculate the *epistemic uncertainty* according, as described in Section 3.3. The *aleatoric uncertainty* is expressed, by the extension of the Monte Carlo-Dropout (MC-Dropout) with the Gaussian output layer.

For the baseline, we applied our explained methods to a fully-connected Neural Network (NN), which consists of four layers, each layer generating 128 outputs. We train the



(a) **(b)**

Figure 4.1: The generated functions are shown in red and the noisy samples are depicted in blue.

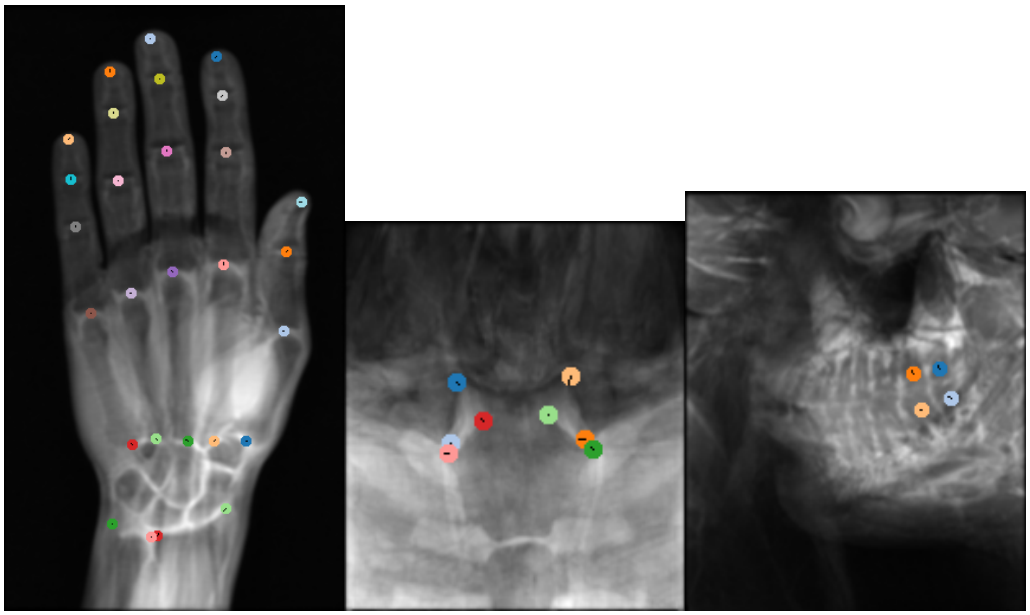
NN, the *MC-Dropout* method and the Ensemble for 30000 iterations and the Bayesian Convolutional Neural Network (BCNN) for 100000 iterations and feed data in batches of size 360 to the model. We chose an Ensemble consisting of $T = 5$ individual models. For *MC-Dropout* we apply dropout with the dropout probability of $p = 0.3$ after each dense layer. We use a scaling factor for the Kullback Leibler (KL)-loss in the *BCNN* of $\lambda = 0.00001$. For the optimization of both methods, we used a learning rate of $\eta = 0.0001$.

4.2 Age Estimation Experiments

4.2.1 Material

The used Magnetic Resonance Imaging (MRI) dataset was acquired at the Ludwig Boltzmann Institute for Clinical Forensic Imaging in Graz as part of a study investigating the role of *MRI* in forensic age estimation. All applied methods in this thesis were evaluated on a dataset of T1-weighted 3D MR images from a dataset, consisting of $N = 328$ subjects with a known Chronological Age (CA), approximately uniformly distributed between 13 and 25 years. The dataset is comprised of *MRI* scan volumes from the left hand, the upper thorax, and the jaw, all acquired in a single imaging session. Hand and clavicle images had been acquired using a T1-weighted gradient-echo sequence with fat saturation. The volumes of the jaw had been imaged using a proton density-weighted turbo spin echo. The size of the acquired images is $288 \times 512 \times 72mm$ with a voxel spacing of $0.45 \times 0.45 \times 0.9mm^3$ for the hand, $168 \times 192 \times 44$ $(0.9 \times 0.9 \times 0.9mm^2)$ for the upper thorax and $208 \times 256 \times 56$ $(0.59 \times 0.59 \times 1.0mm^3)$ for the jaw. An example for the used images can be seen in Fig. 4.2.

We used a 4-fold cross-validation for the evaluation of the model performance. We aimed to evaluate the contribution of the different sites to the performance of the models. Therefore, all methods were carried out for training on hand bones alone and hand bones



(a) (b) (c)

Figure 4.2: Example projection images of the recorded MR images with their according landmarks showing the (a) hand bones, (b) clavicle bones and (c) teeth.

combined with clavicle bones and teeth. Additionally to using the volumetric data, we repeated all experiments, using just the 2D middle slices of the cropped volumes. The middle slices are extracted, such that the bone of interest lies perpendicular to the 2D plane.

4.3 Network architecture

Our network architecture was chosen to be the *late fusion* architecture of Štern et al. [52]. The *late fusion* architecture is oriented on how forensic experts currently perform combination of the age information of hand bones, clavicle bones and teeth. This architecture combines the age information for each site individually and concatenates the information between all sites, bevor the final prediction. The used *late fusion* architecture for Convolutional Neural Networks (CNNs), Ensembles and *BCNNs* can be seen in Fig. 4.3 with its corresponding Deep Convolutional Neural Network (DCNN) block in Fig. 4.5. This particular architecture mimics the stating of different anatomical sites, where each *DCNN* block acts as a feature extractor per cropped input volume. In Fig. 4.4 the the architecture in Fig. 4.3 is modified in order to implement *MC-Dropout*. The corresponding *DCNN* block can be seen in Fig. 4.6.

Each *DCNN* block consists of three sequential levels of two convolution layers and a pooling layer that halves the input tensor to each level. The convolutions do not use padding. Thus each convolutional layer reduces the input tensor in each dimension by

2. At the end of a *DCNN* block, the tensor was flattened and passed through a dropout layer to the fully connected layer, which produced 96 outputs. This dropout layer drops neurons with the probability of p_d. The extracted features of each cropped volume were combined per site using a fully connected layer with 32 units. A dense layer acts as the output layer, which produces task-dependent predictions. The output layer was modified, depending on the experiment (see Section 3.3) and is visualized in Fig. 3.3.

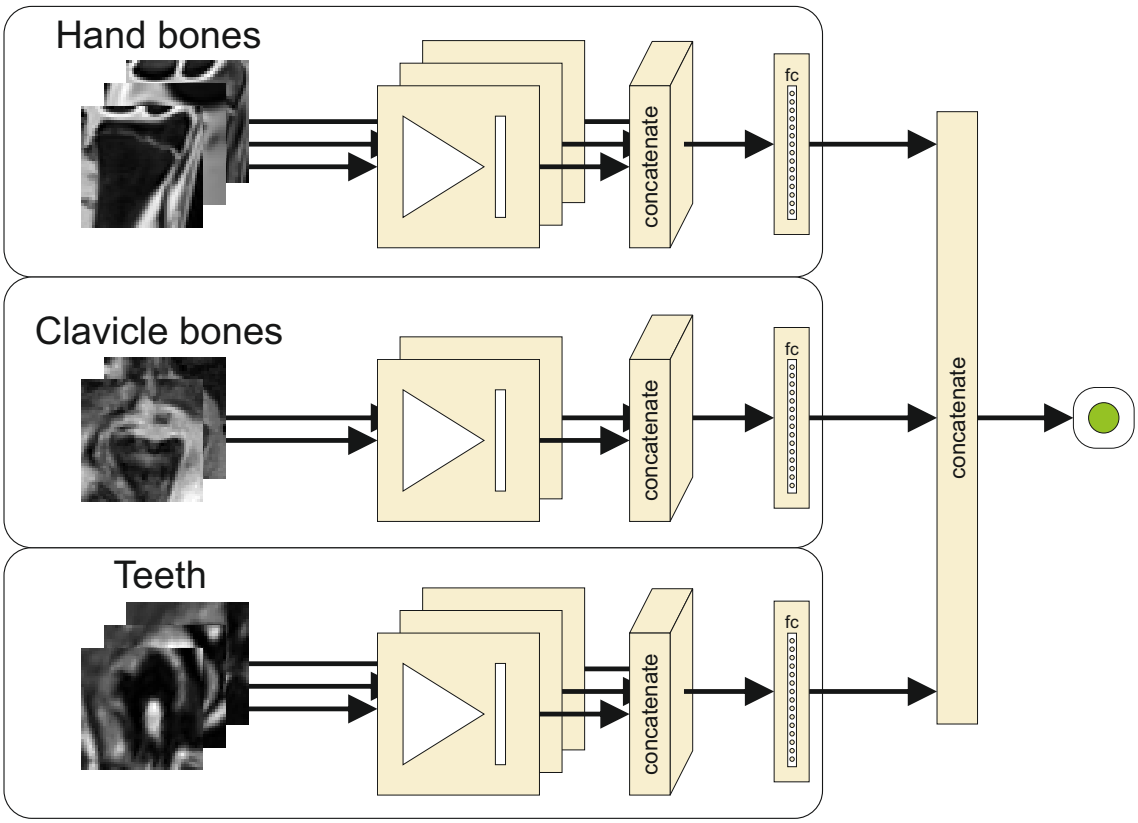


Figure 4.3: Schematic representation of the used network architecture. The images of each site are connected via a separate DCNN block to a fully connected layer f_c. The features of the final fully connected layer of each site are merged into a final prediction.

4.4 Data augmentation

We used an on-the-fly data augmentation of the training images by sampling values of a uniform distribution in defined ranges. According to these sampled values, the images were scaled, rotated, and translated. The landmarks of the cropped volumes of each site were aligned. After the augmentation of the whole volume, the images were cropped around the landmarks with a bounding box of the size $[44mm, 44mm, 44mm]$. We interpolated to a uniform voxel spacing of $[1mm, 1mm, 1mm]$ using linear interpolation. We applied a random rotation between $[-5.73°, 5.73°]$ for the x and z-axis and between $[-11.4°, 11.4°]$

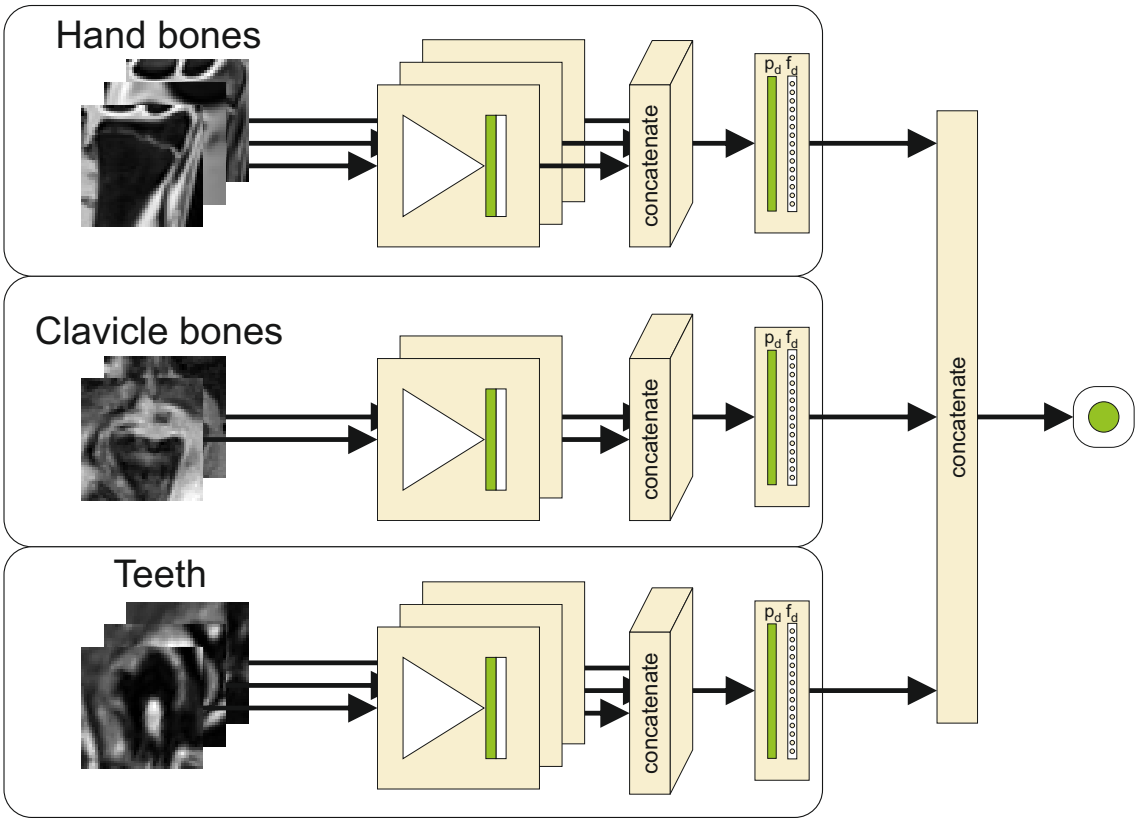


Figure 4.4: Schematic representation of the used late fusion network architectures. The images of each site are connected via a separate DCNN block to a fully connected layer f_c. The final fully connected layer f_c and the DCNN block implement*MC-Dropout*. The features of the final fully connected layer of each site are merged into a final prediction.

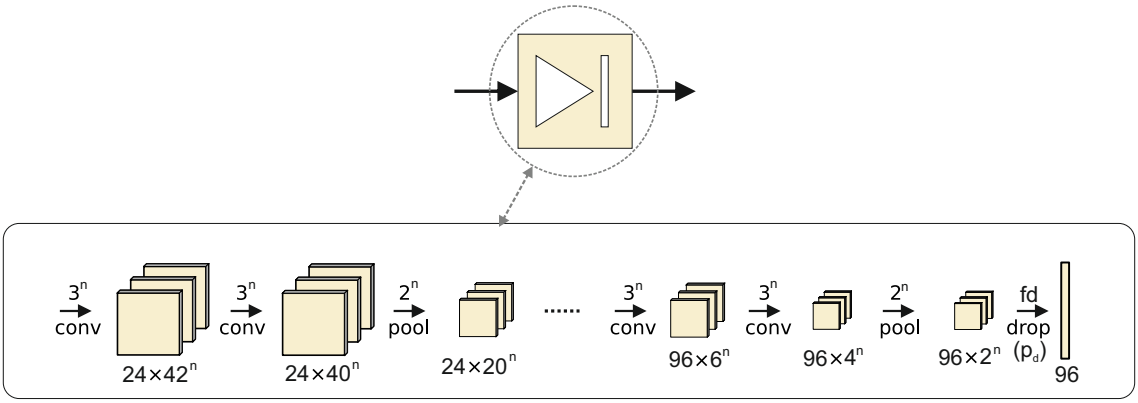


Figure 4.5: Schematic representation of a DCNN block. It consists of multiple levels of two consequential convolutional operations without padding, followed by a pooling operation. The features are then passed via a dropout layer with dropout probability p_d to a fully connected layer (f_d) The n indicates the dimensionality of the kernels, which can be 2 and 3 in this work.

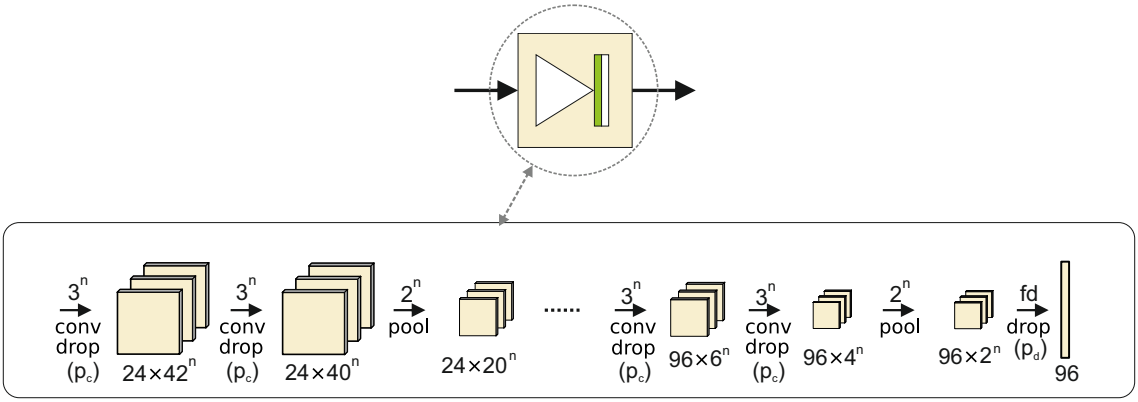


Figure 4.6: Schematic representation of a DCNN block using *MC-Dropout*. This block extends Fig. 4.5 with additional dropout layers with dropout probability p_c, after each convolutional layer. It consists of multiple levels of two consequential convolutional operations without padding, followed by a pooling operation.

for the y-axis. The volumes of the finger joints and wrist bones were scaled in the range of $[0.85, 1.15]$ for all dimensions and translated by $[-2mm, 2mm]$ for all dimensions. The volumes of the clavicles were rotated about the voxel center in the range of $[-5.73°, 5.73°]$, translated by $[-2mm, 2mm]$ and scaled by $[0.85, 1.15]$ for all dimensions. This random augmentation was applied during training, each image in each iteration individually.

The intensity values of the training images were randomly shifted in the range of $[0.75, 1.25]$ and scaled within $[0.6, 1.4]$. The intensities were additionally clamped in the range of $[-2, 2]$.

4.5 Training setup

We used a mini-batch size of $B = 8$ for all experiments. We applied *Tensorflow* as Machine Learning (ML) framework [1] and the *Adam* optimizer [22]. For the initialization of the model weights, we used the *He Normal* initialization proposed by He et al. [17]. We utilized a maximal number of iterations of 20000 and a learning rate of $\eta = 0.0001$ for all methods except *BCNN*, where we used a learning rate of $\eta = 0.001$ up till 10000 iterations and $\eta = 0.0001$ from 10000 iterations onwards.

To ensure convergence for experiments, which involved the optimization of the Negative Log Likelihood from the Gaussian output layer, as well as for *BCNNs*, we performed normalization of the age labels for experiments. We calculated the mean and standard deviation of the age labels of all samples occurring in the current training set beforehand and used those statistics to normalize the labels during training and evaluation. For all experiments conducted in this work, we used the Rectified Linear Unit (ReLU)[38] as a non-linearity in the network.

4.5.1 Convolutional Neural Networks

In the case of a conventional *CNN*, we considered only one output of the network, and used the ℓ_2 loss in Eq. (3.23) as our loss function.

During training, we used a dropout rate of $p_d = 0.5$ for the fully connected layers. This approach was extended by the Gaussian output layer in Section 3.3.1 to additionally express the *aleatoric uncertainty*. In this case, we optimized the Negative Log Likelihood from Eq. (3.22).

4.5.2 Monte Carlo-Dropout

For this method, we applied *dropout* after every dense layer with a rate of $p_d = 0.5$ and after every convolutional layer with a *dropout* rate of $p_c = 0.1$ during training and evaluation time. To overcome the potential decrease of evaluation performance due to *dropout* after convolutions, we averaged the results over $T = 20$ forward passes.

The architecture is reformulated in order to extract both *epistemic uncertainty* and *aleatoric uncertainty* as described in Section 3.3. We evaluated the *model uncertainty* using Eq. (3.19).

4.5.3 Ensemble

We trained Ensemble consisting of $T = 5$ separately trained models $\{f_t\}_{t=1}^{T}$. The *DCNN* paths of each model consists of $(24, 48, 96), (24, 32, 48), (32, 48, 96), (24, 24, 96), (24, 96, 96)$ filters per level of convolution and the same number of features for the fully connected layers as described in Section 4.3. The experiments were carried out for both types of output, as described in Section 3.3.

4.5.4 Bayesian Convolutional Neural Networks

The network architecture in our implementation of the *BCNN* is equivalent to the one of *CNNs* and *MC-Dropout*. To capture the *epistemic uncertainty* inherently in the model weights, we convert the default *CNN* to a *BCNN*, by replacing the convolutional and dense layers with Reparameterization layers, as described in Section 2.4.5. Due to the statistical behavior of those layers, *MC-Dropout* for the purpose of *model uncertainty* extraction is not needed. We evaluated the *epistemic uncertainty* over $T = 20$ forward passes per sample.

The *aleatoric uncertainty* is given per sample at the model output, according to Section 3.3. A scaling factor of $\lambda = 10^{-6}$ of the *KL*-divergence was selected. To ensure the positivity of the variance, we transformed the output using the *Softplus* function on the prediction σ of the network, as given in the equation bellow

$$\sigma(x) = 10^{-3} + \ln\left(1 + \exp(\hat{\sigma})\right) \tag{4.1}$$

and added a constant of 10^{-3} to ensure numerical stability.

In practice, model the weights are sampled each iteration from their individual distribution over the model weights. While it is possible to average the activations over multiple sampled weights for each layer, in this thesis, we used one set of sampled model weights per forward pass. We conducted the experiments with both formulations of the output layer as described in Section 3.3 and used the corresponding loss functions described in Section 3.5.

5 Results

Contents

In this thesis we evaluated different Deep Learning (DL) methods, regarding their capability of extracting the uncertainty, introduced by to the difference in the biological variation of the Biologial Age (BA) and the Chronological Age (CA). The applied methods encompass Convolutional Neural Networks (CNNs), Ensemble model combination, *CNNs* implementing Monte Carlo-Dropout (MC-Dropout) and Bayesian Convolutional Neural Networks (BCNNs). All of those methods were extended with a Gaussian output layer in order to extract the *aleatoric uncertainty* directly from the model predictions. We performed a 4-fold cross-validation using the cropped 3D imaging volumes of bones and using just the 2D middle slices of those cropped volumes. To evaluate the influence of the age information included in different sites of the human body, we repeated the experiments for training on just hand and wrist images (H) and a combination of hand and wrist images, including clavicles and wisdom teeth (HCT). To evaluate the overall performance of the different methods, all results needed to be evaluated quantitatively and visually. In order to theoretically evaluate the influence of the training data onto the *epistemic uncertainty* and the *aleatoric uncertainty*, we additionally conducted some experiments on two one dimensional functions.

5.1 Baseline Experiments

The followings results are gained from the experiments on both the sinusoidal (Fig. 4.1a) and the linear function (Fig. 4.1b). Fig. 5.4 and Fig. 5.1 shows the extracted *aleatoric*

uncertainty from the Neural Network (NN), by using the Gaussian output layer. Figure 5.2 and Fig. 5.5 visualize the expressed *epistemic uncertainty* for the *MC-Dropout* method and the Bayesian Neural Network (BNN), using the Gaussian output layer. Figure 5.2 and Fig. 5.5 depict the *epistemic uncertainty* for the Ensemble, the *MC-Dropout* method and the *BNN* without the usage of the Gaussian output layer. Fig. 5.3 and Fig. 5.6 depict the extracted *aleatoric uncertainty* and *epistemic uncertainty* by the Ensemble, the *MC-Dropout* method and a *BNN*, by applying the Gaussian output layer.

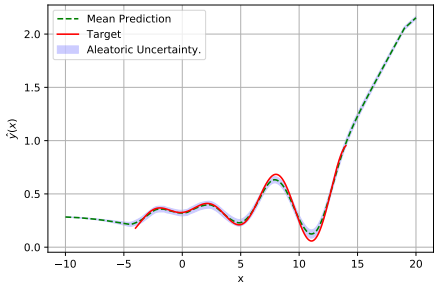
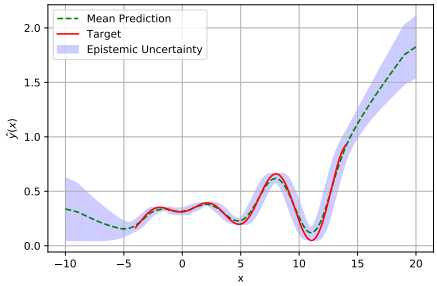
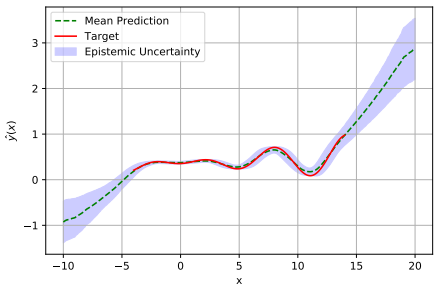


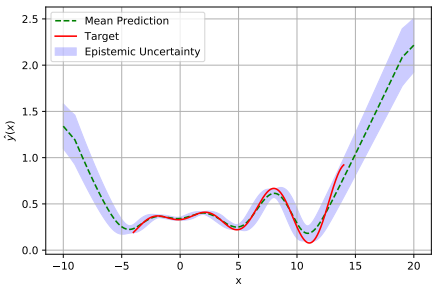
Figure 5.1: Visualization of the extracted *aleatoric uncertainty* from the NN.



(a) Ensemble

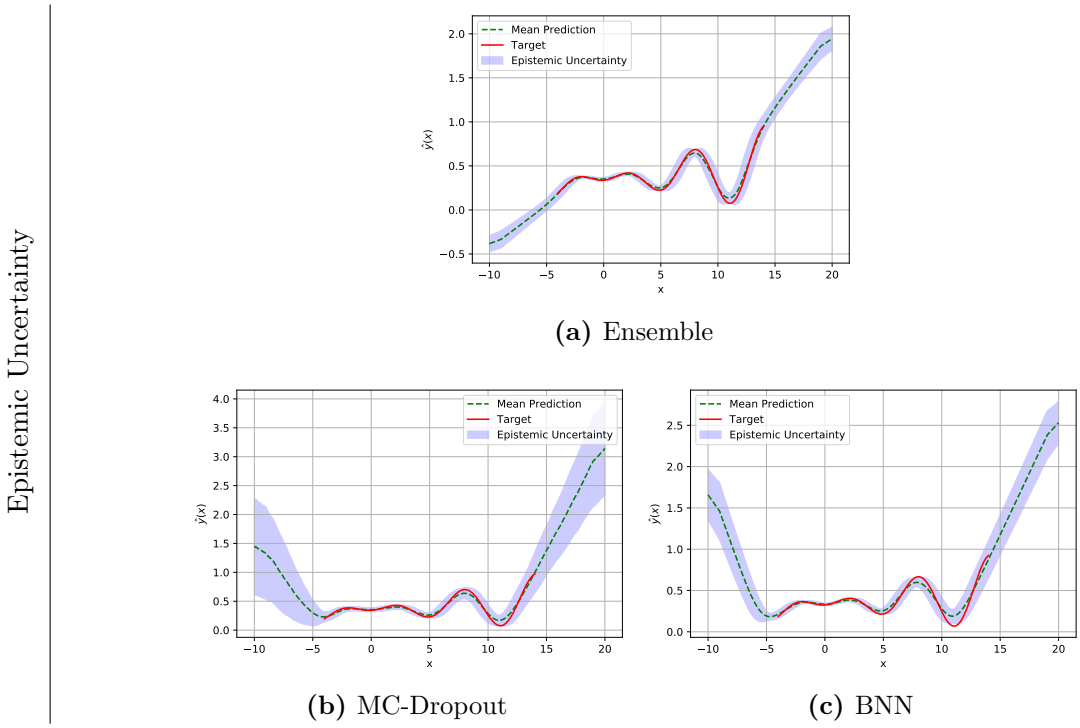


(b) MCD



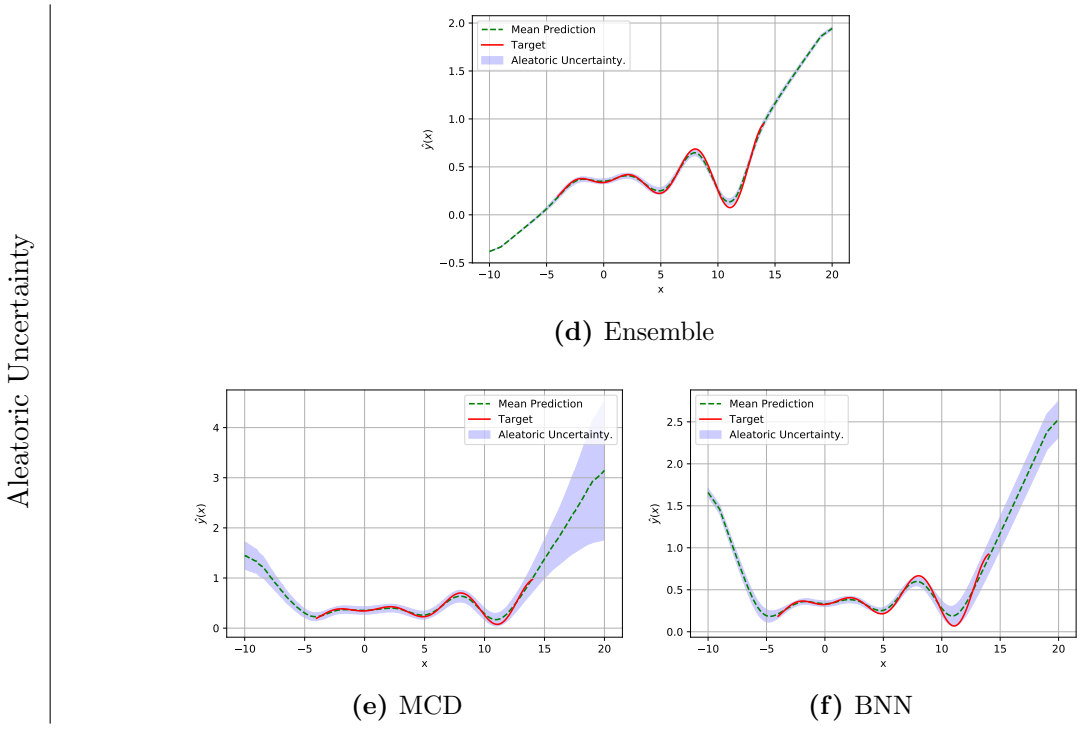
(c) BNN

Figure 5.2: Visualization of the *epistemic uncertainty* for the MC-Dropout method and a Bayesian NN for the sinusoidal function.



(a) Ensemble

(b) MC-Dropout **(c)** BNN



(d) Ensemble

(e) MCD **(f)** BNN

Figure 5.3: Visualization of the *epistemic uncertainty* and the *aleatoric uncertainty* generated for the stated methods, implementing the Gaussian output layer.

In Table 5.1 the averaged *aleatoric uncertainty* and *epistemic uncertainty* for the compared methods is listed. The average uncertainty was calculated for samples $x \geq 18$ and $x < 18$. For $x < 18$, we sampled our training data from a uniform distribution with standard deviation of 0.3. For $x \geq 18$ based on the samples uniform distribution, the expected standard deviation is 2.02. Additionally to the averaged uncertainty the standard deviation of the predictions is stated.

Experiment type	Epistemic		Aleatoric	
	$x < 18$	$x \geq 18$	$x < 18$	$x \geq 18$
NN plus σ	-	-	0.30 ± 0.06	1.82 ± 0.25
Ensemble	0.71 ± 0.42	0.60 ± 0.51	-	-
Ensemble plus σ	0.76 ± 0.53	0.65 ± 0.43	0.28 ± 0.03	1.83 ± 0.20
MCD	0.81 ± 0.36	0.94 ± 0.39	-	-
MCD plus σ	0.97 ± 0.28	1.12 ± 0.45	1.22 ± 0.49	2.44 ± 0.55
BNN	0.75 ± 0.5	0.58 ± 0.45	-	-
BNN plus σ	0.73 ± 0.34	0.60 ± 0.42	0.39 ± 0.98	2.00 ± 0.19

Table 5.1: Listing of the *epistemic uncertainty* and *aleatoric uncertainty* extracted for the given methods, from the baseline experiments on the linear function.

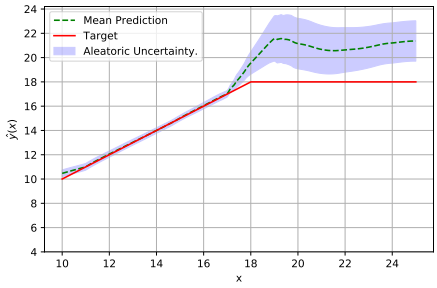
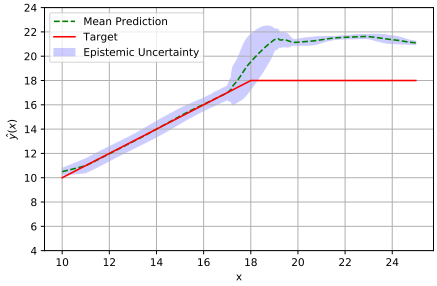
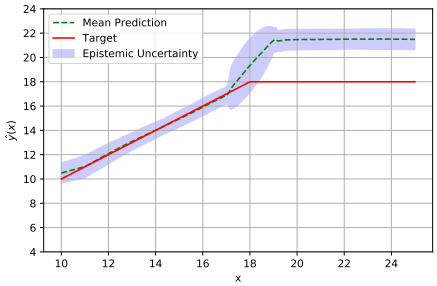


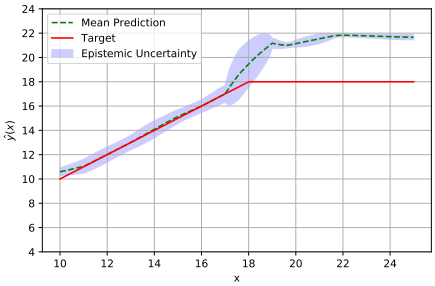
Figure 5.4: Visualization of the extracted *aleatoric uncertainty* from the NN.



(a) Ensemble



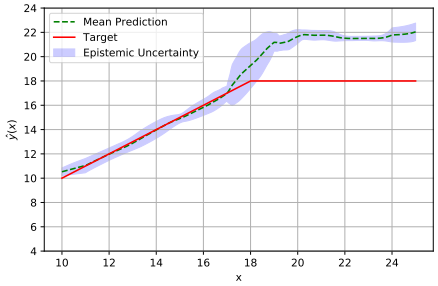
(b) MC-Dropout



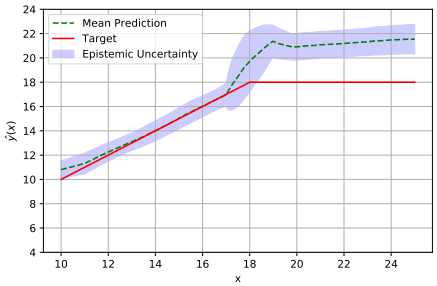
(c) BNN

Figure 5.5: Visualization of the *epistemic uncertainty* for the MC-Dropout method and a Bayesian NN for the linear function

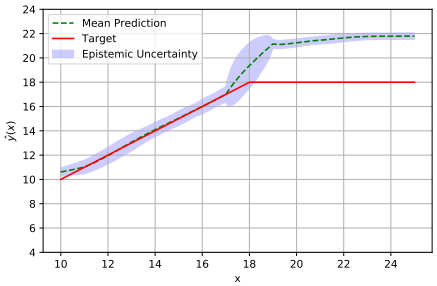
Epistemic Uncertainty



(a) Ensemble

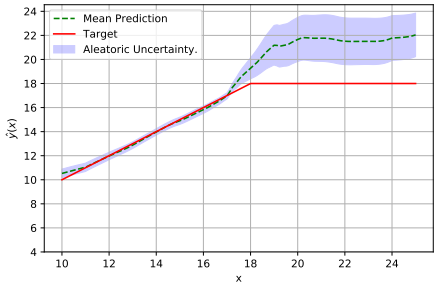


(b) MC-Dropout

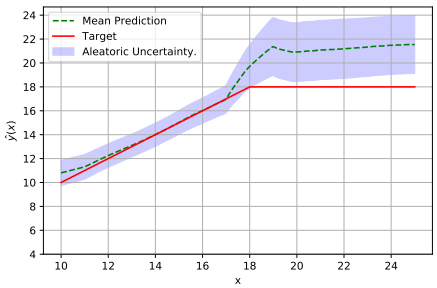


(c) BNN

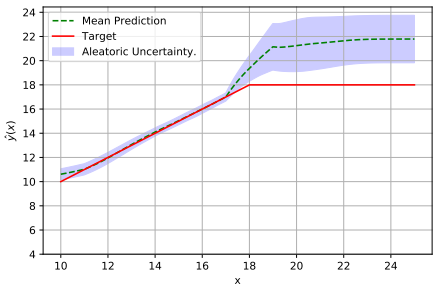
Aleatoric Uncertainty



(d) Ensemble



(e) MC-Dropout



(f) BNN

Figure 5.6: Visualization of the *epistemic uncertainty* and the *aleatoric uncertainty* generated for the stated methods, implementing the Gaussian output layer.

5.2 Quantitative Evaluation of the Results

Table 5.3 and Table 5.2 list the results for the evaluated methods for 2D and 3D experiments, trained on the two different combinations of the sites of the hand bones (H) and clavicle bones plus teeth (CT). All of the stated results are extracted from the regression plots for the single methods in Figs. 5.7 to 5.10. The Mean Absolute Deviation (MAD) describes the average distance between each prediction to the mean of the whole population of predictions. For methods, which are trained on hand bones , the *MAD* was calculated only for subjects $\leq$ 18 years in age. Caused by the method, the *epsitemic uncertainty* is not available for the standard *CNN*. The *aleatoric uncertainty* is only available for methods which are extended with the Gaussian output layer.

Investigating the results for methods trained on hand bones in Table 5.2, the best performance regarding the *MAD* was archived in 3D with the *BCNN* with the Gaussian output layer (0.94 ± 0.80). Both *BCNN* and Ensemble methods follow the best result in 3D very closely, with a slightly increased standard deviation of the predictions (0.98 ± 0.85 and 0.98 ± 0.87). Compared results, using 2D data, show an overall slightly increased *MAD*, compared to the results in 3D. In 2D, the Ensemble performs the best (0.99 ± 0.90), closely followed by *MC-Dropout* (1.04 ± 0.97) and the conventional *CNN* (1.05 ± 0.97).

The results of methods, trained on the combination of sites HCT show an overall increase in *MAD*, compared to training on hand bones only. Despite the increase in *MAD*, the overall standard deviation for the single experiments is decreased when trained on the combination of sites HCT, over training on hand bones alone. With 3D data, the best fit was gained by the Ensemble method (0.98 ± 0.77), which is followed by *MC-Dropout* (1.02 ± 0.78) and the conventional *CNN* (1.06 ± 0.77). Using the 2D middle slices, the best *MAD* of 1.02 ± 0.81 was archived by the Ensemble method. As opposed to the models trained on H, the other results in 2D deviate more from the best result, namely by the *MC-Dropout* (1.01 ± 0.83) and the conventional *CNN* (1.16 ± 0.87).

Experiment type	Sites	MAD	Epistemic	Aleatoric
CNN	H	1.04 ± 0.88	-	-
	HCT	1.06 ± 0.77	-	-
CNN plus σ	H	1.17 ± 0.96	-	1.61 ± 0.42
	HCT	1.23 ± 0.97	-	1.41 ± 0.41
Ensemble	H	0.98 ± 0.87	0.30 ± 0.22	-
	HCT	0.98 ± 0.77	0.38 ± 0.23	-
Ensemble plus σ	H	1.14 ± 1.03	0.65 ± 0.41	1.74 ± 0.39
	HCT	1.21 ± 0.93	0.99 ± 0.55	1.82 ± 0.35
MCD	H	1.03 ± 0.89	0.11 ± 0.05	-
	HCT	1.02 ± 0.78	$\mathbf{0.09 \pm 0.04}$	-
MCD plus σ	H	1.06 ± 0.98	0.31 ± 0.14	1.59 ± 0.36
	HCT	1.17 ± 0.94	0.29 ± 0.20	1.39 ± 0.36
BCNN	H	0.98 ± 0.85	0.52 ± 0.20	-
	HCT	1.13 ± 0.87	0.54 ± 0.24	-
BCNN plus σ	H	$\mathbf{0.94 \pm 0.80}$	0.11 ± 0.06	$\mathbf{0.52 \pm 0.17}$
	HCT	1.19 ± 0.94	0.24 ± 0.14	$\mathbf{0.74 \pm 0.28}$

Table 5.2: Quantitative results evaluating the performance of the different methods using the 3D volumes of each site. *H* in indicates the usage of just hand bones, whereas *HCT* indicates the use of hand bones, clavicle bones and teeth as training data. Methods, which implement the Gaussian output layer are denoted by the addition of σ to the method name. The MAD and the standard deviation of the predictions for methods, trained on only hand bones, were calculated in the age range between 13 and 19 years. For hand bones, clavicle bones and teeth, the MAD and the predictions standard deviation was calculated over the whole age range between 13 and 25 years.

Experiment type	Sites	MAD	Epistemic	Aleatoric
CNN	H	1.05 ± 0.97	-	-
	HCT	1.16 ± 0.87	-	-
CNN plus σ	H	1.27 ± 1.21	-	1.38 ± 0.52
	HCT	1.33 ± 1.11	-	1.15 ± 0.42
Ensemble	H	0.99 ± 0.90	$0.43 \pm 0.20/$	-
	HCT	$\mathbf{1.02 \pm 0.81}$	0.33 ± 0.18	-
Ensemble plus σ	H	1.32 ± 1.20	0.79 ± 0.44	1.97 ± 0.49
	HCT	1.24 ± 0.99	0.78 ± 0.41	1.55 ± 0.41
MCD	H	1.04 ± 0.97	$\mathbf{0.15 \pm 0.08}$	-
	HCT	1.10 ± 0.83	$\mathbf{0.15 \pm 0.08}$	-
MCD plus σ	H	1.16 ± 1.19	0.29 ± 0.13	1.48 ± 0.61
	HCT	1.28 ± 1.04	0.29 ± 0.20	1.15 ± 0.42
BCNN	H	1.10 ± 0.93	0.47 ± 0.25	-
	HCT	1.18 ± 0.88	0.55 ± 0.27	-
BCNN plus σ	H	1.07 ± 0.94	0.26 ± 0.12	$\mathbf{0.97 \pm 0.37}$
	HCT	1.24 ± 0.98	0.42 ± 0.21	1.15 ± 0.34

Table 5.3: Quantitative results evaluating the performance of the different methods using the 2D middle slices of the cropped volumes of each site. *H* in indicates the usage of just hand bones, whereas *HCT* indicates the use of hand bones, clavicle bones and teeth as training data. Methods, which implement the Gaussian output layer are denoted by the addition of σ to the method name. The MAD and the standard deviation of the predictions for methods, trained on only hand bones, were calculated in the age range between 13 and 19 years. For hand bones, clavicle bones and teeth, the MAD and the predictions standard deviation was calculated over the whole age range between 13 and 25 years.

5.2.1 Visualization of the Regression Plots

In Figs. 5.7 to 5.10 the regression plots for *CNNs*, the *MC-Dropout* method, Ensembles and *BCNNs* are depicted. The Subfigures (a) and (b) show the regression plot for the corresponding method for 3D data, trained on hand bones, clavicles and teeth. In the Subfigs. (e) and (f) the regression plot for 3D data, trained on hand bones, is depicted. Subfigs. (c, d) visualize the regression plots for the stated methods, trained on 2D data from hand bones, clavicles and teeth, while (g) and (h) show the results when training on hand bones. In those plots, the predicted sample age is plotted versus the groundtruth age.

Convolutional Neural Networks

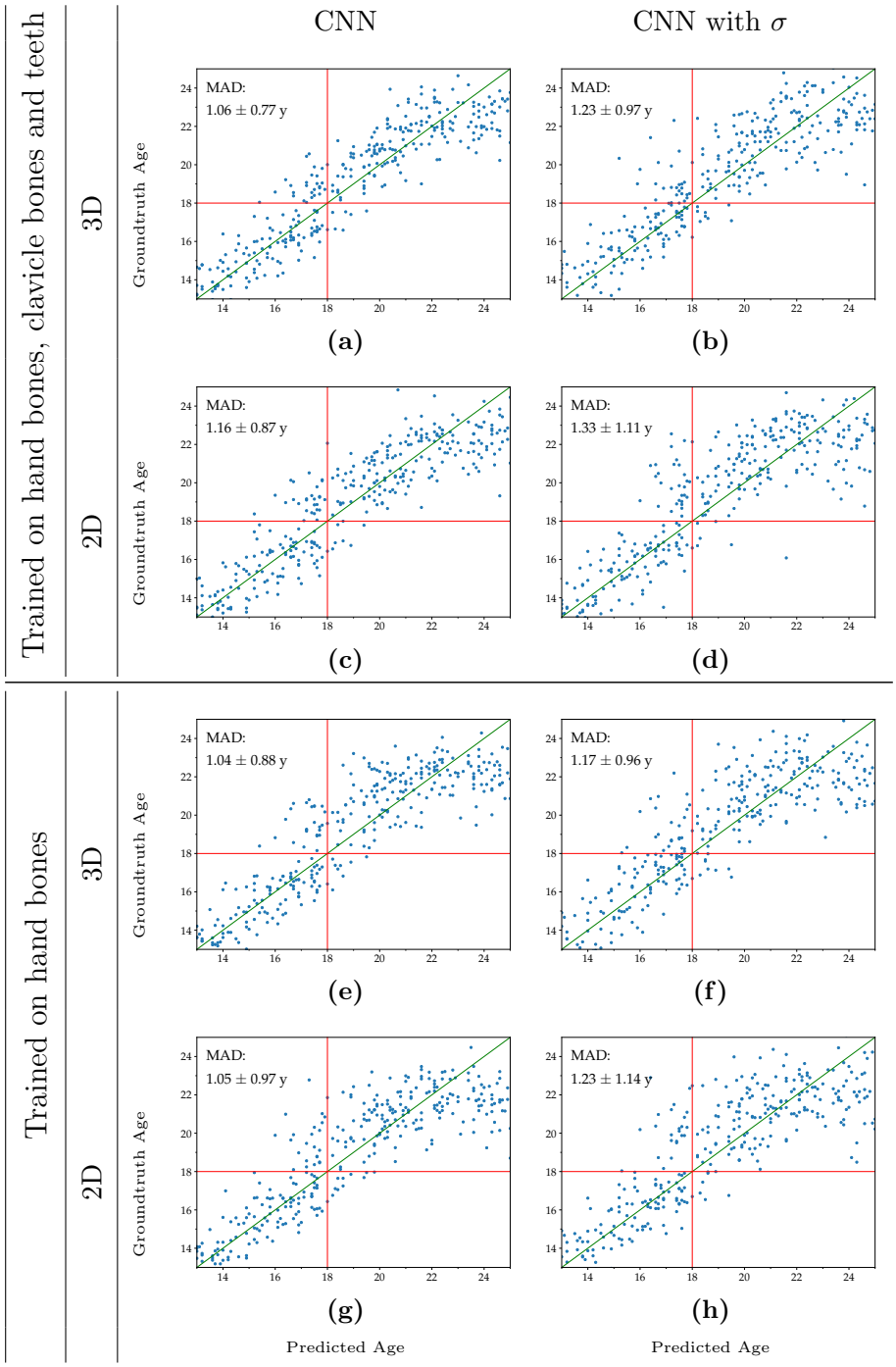


Figure 5.7: Visualization of the regression plots for CNNs, with both 2D and 3D data.

Ensemble

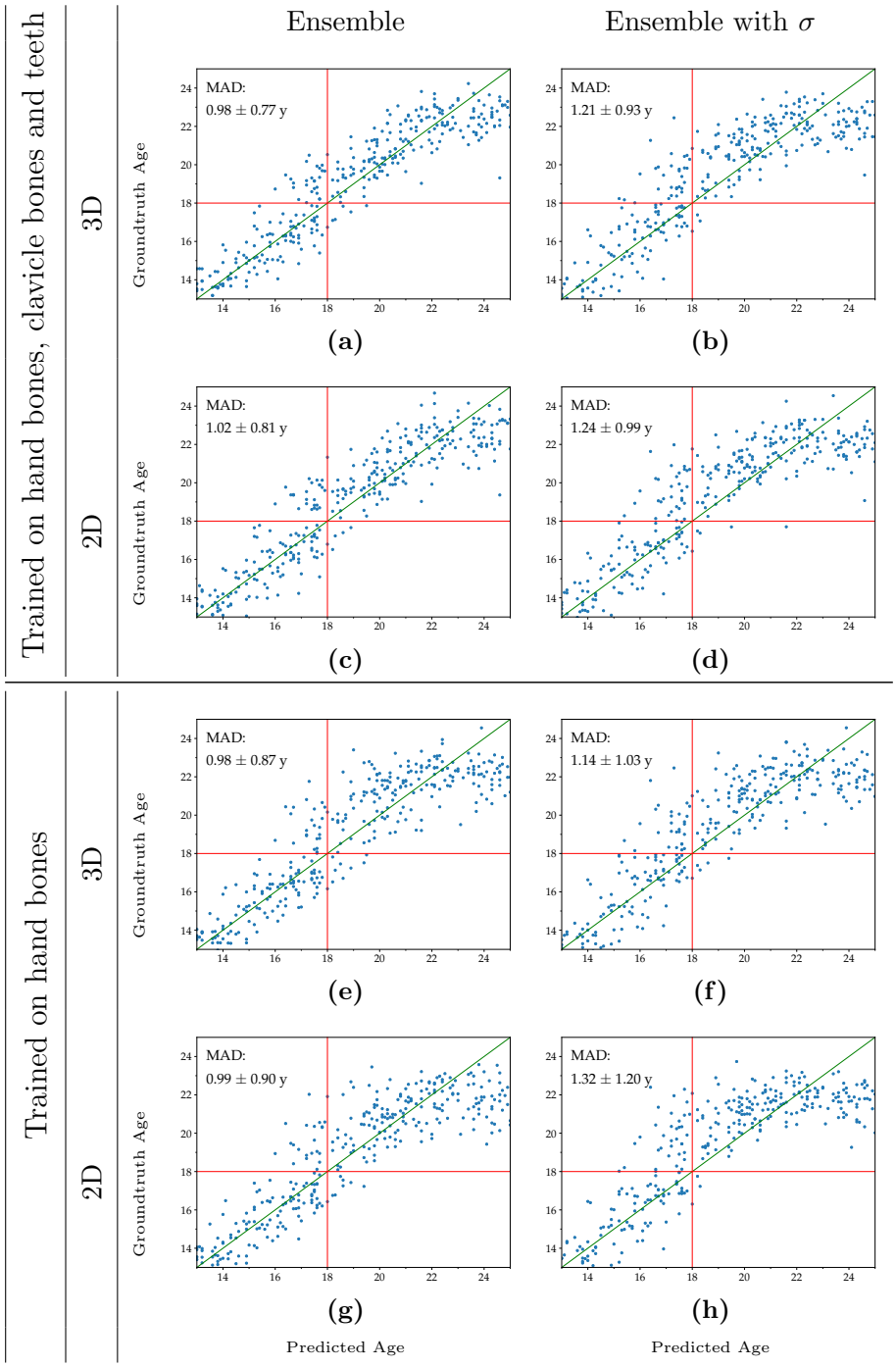


Figure 5.8: Visualization of the regression plots for Ensembles, with both 2D and 3D data.

Monte Carlo-Dropout

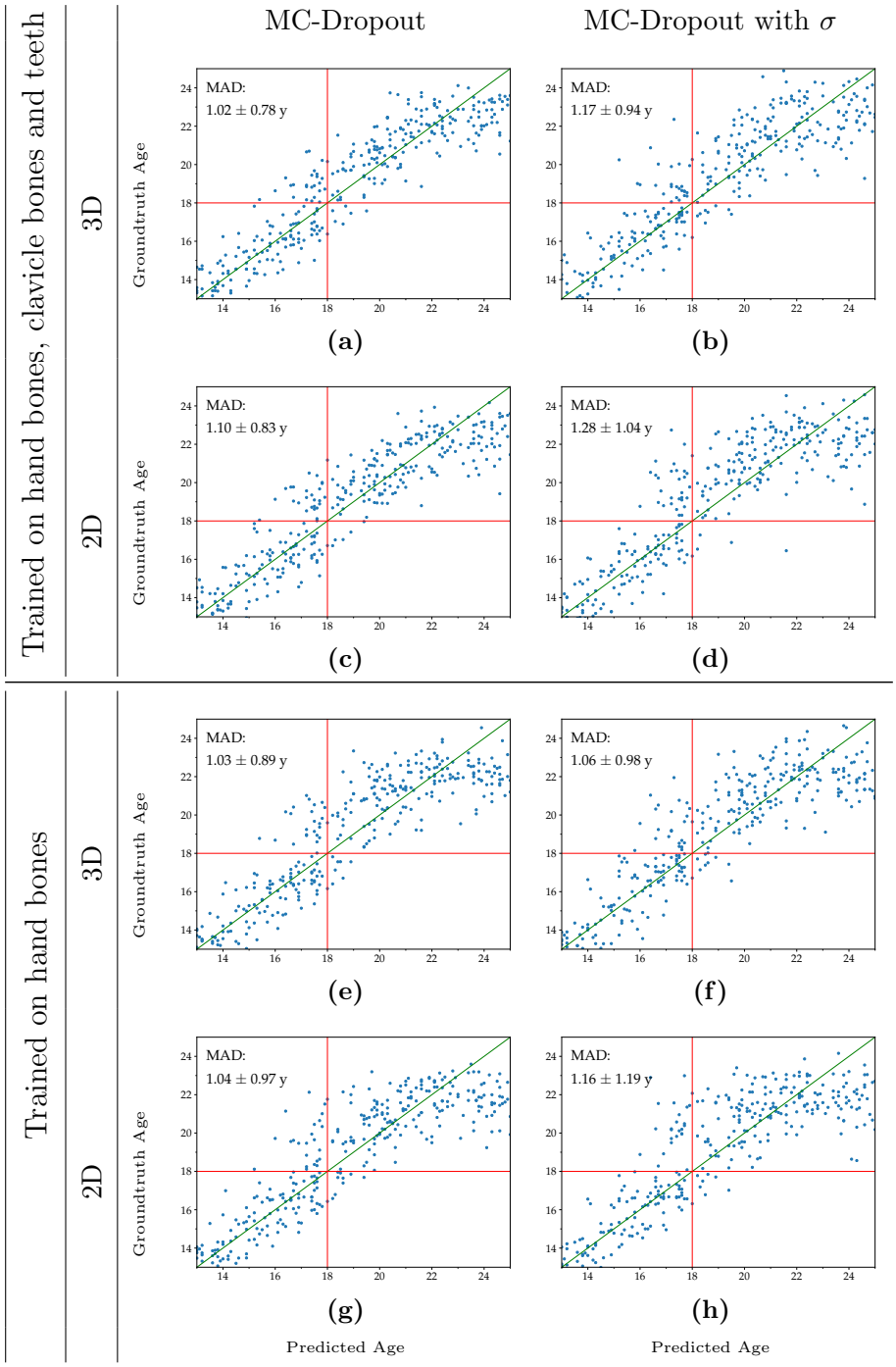


Figure 5.9: Visualization of the regression plots for MC-Dropout, with both 2D and 3D data.

Bayesian Convolutional Neural Networks

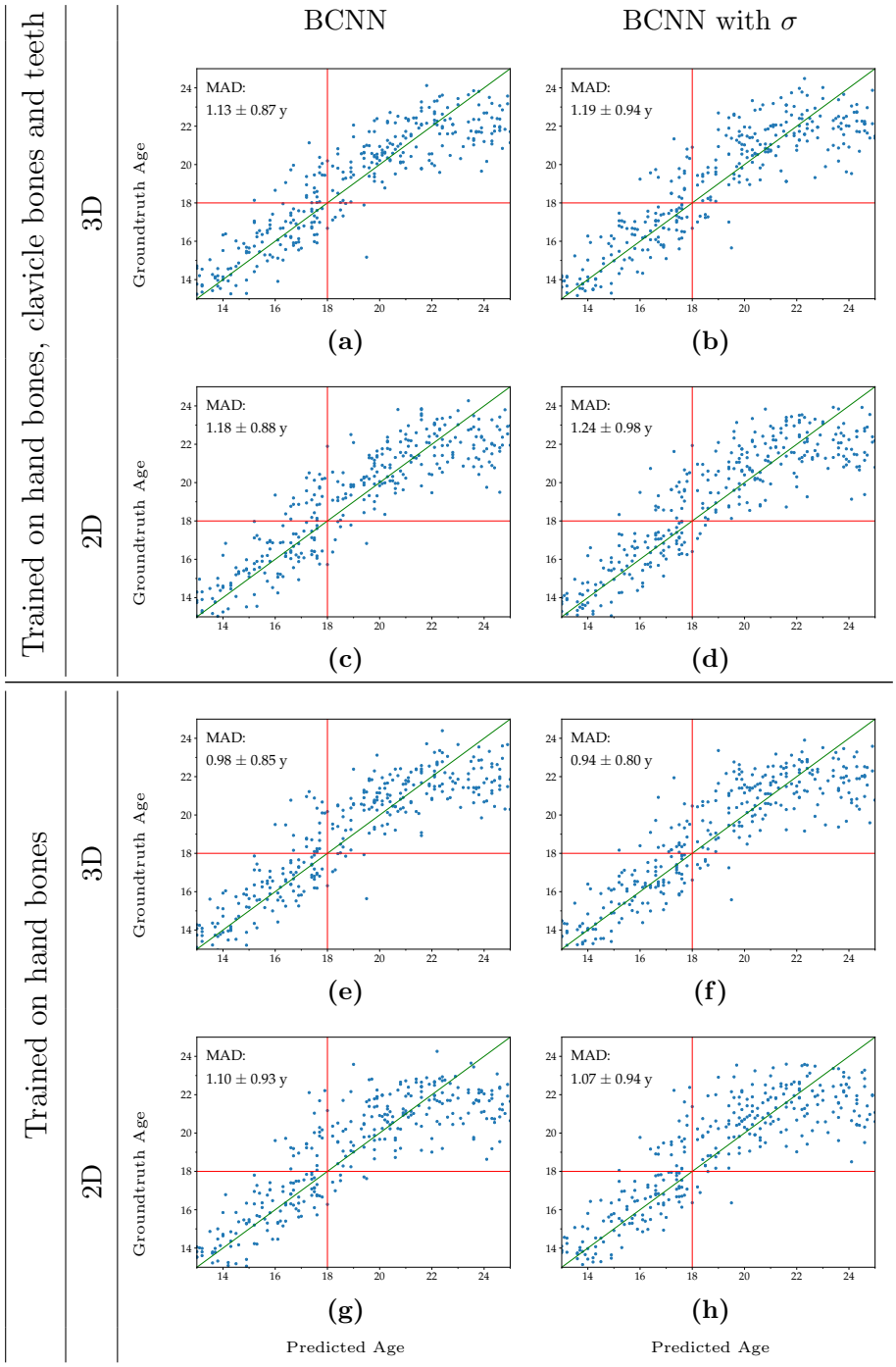


Figure 5.10: Visualization of the regression plots for BCNNs, with both 2D and 3D data.

5.3 Visual Evaluation of the Uncertainty Plots

In the Subfigs. (a) and (b) in Fig. 5.11 and Fig. 5.12 the *aleatoric uncertainty* of *CNNs*, extended by the Gaussian output layer are visualized. The Subfigs. (a, b) in Figs. 5.13 to 5.18 represent the sample-wise *epistemic uncertainty* of the corresponding methods. The Subfigs. (b, e) visualize the extracted *epistemic uncertainty* and the Subfigs. (c, f) visualize the *aleatoric uncertainty* for the stated methods, which are extended by the Gaussian output layer. All plots depict the corresponding uncertainty plotted versus the predicted age of the model.

5.3.1 Convolutional Neural Networks

3D Experiments

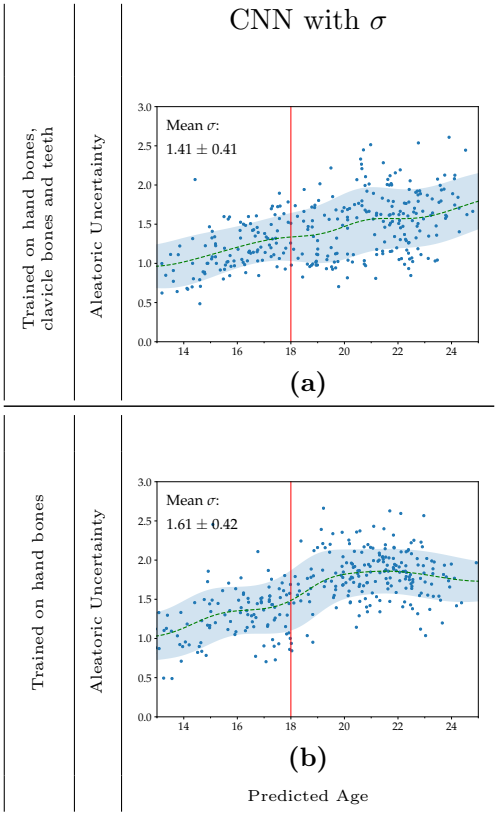


Figure 5.11: Visualization of the *aleatoric uncertainty* for the CNN, using the Gaussian output layer and 3D data.

2D Results

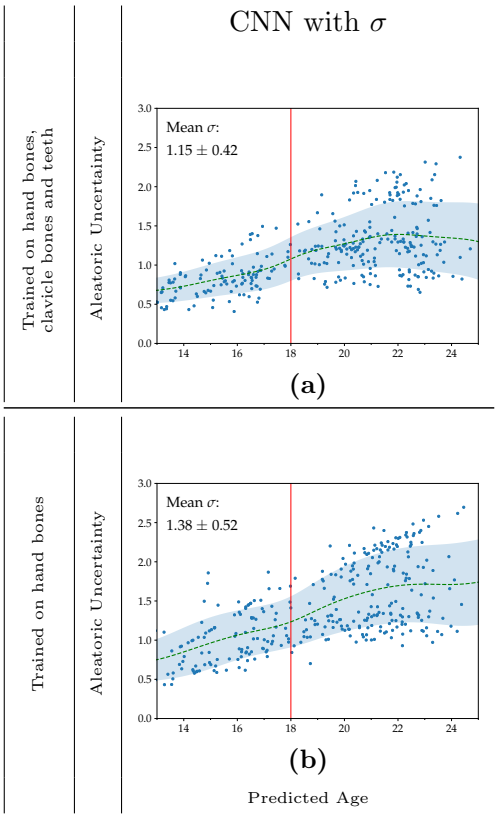


Figure 5.12: Visualization of the *aleatoric uncertainty* for the CNN, using the Gaussian output layer and 2D data.

5.3.2 Ensembles

3D Results

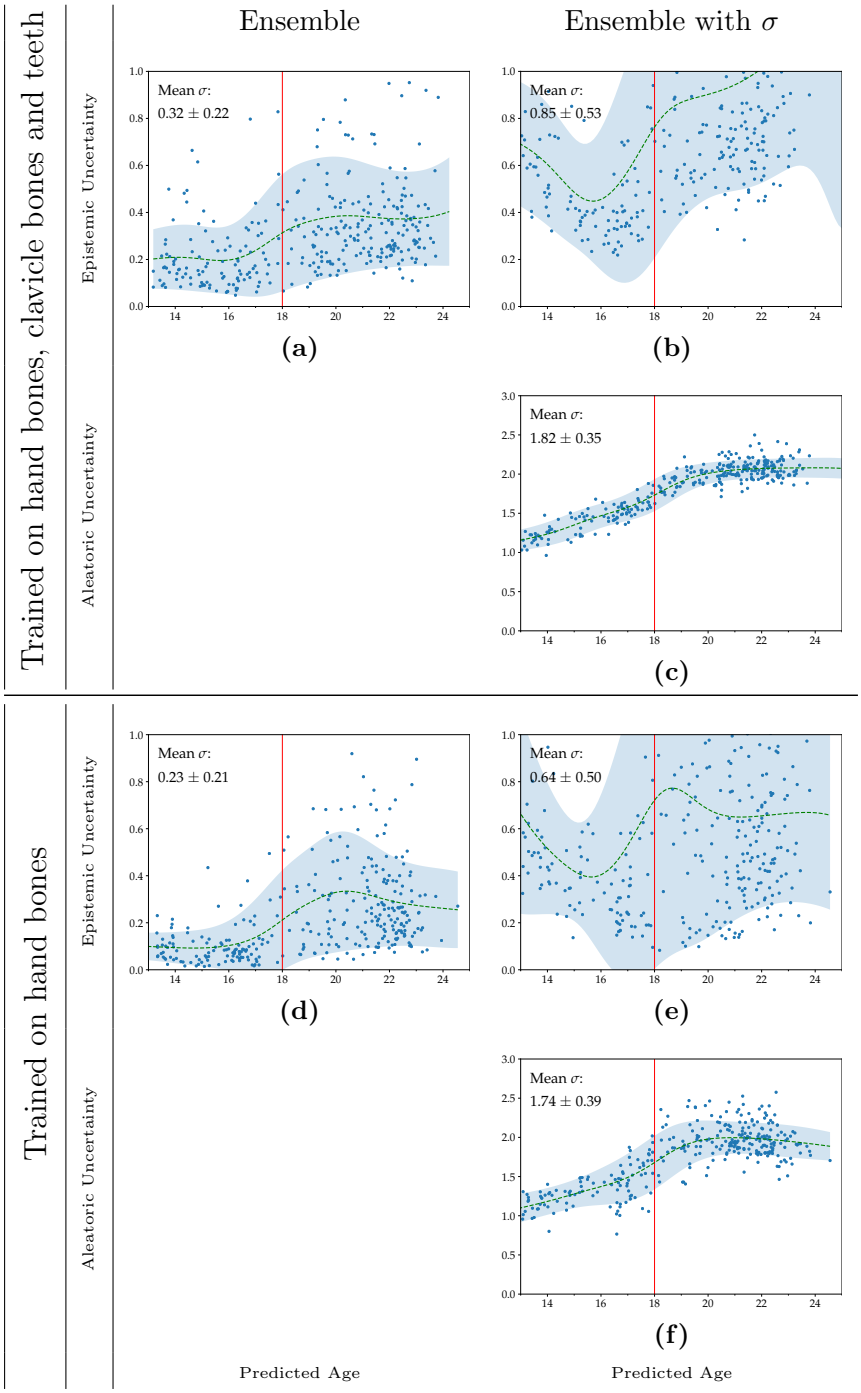


Figure 5.13: Visualization of the extracted uncertainties with the Ensemble, using 3D data.

2D Results

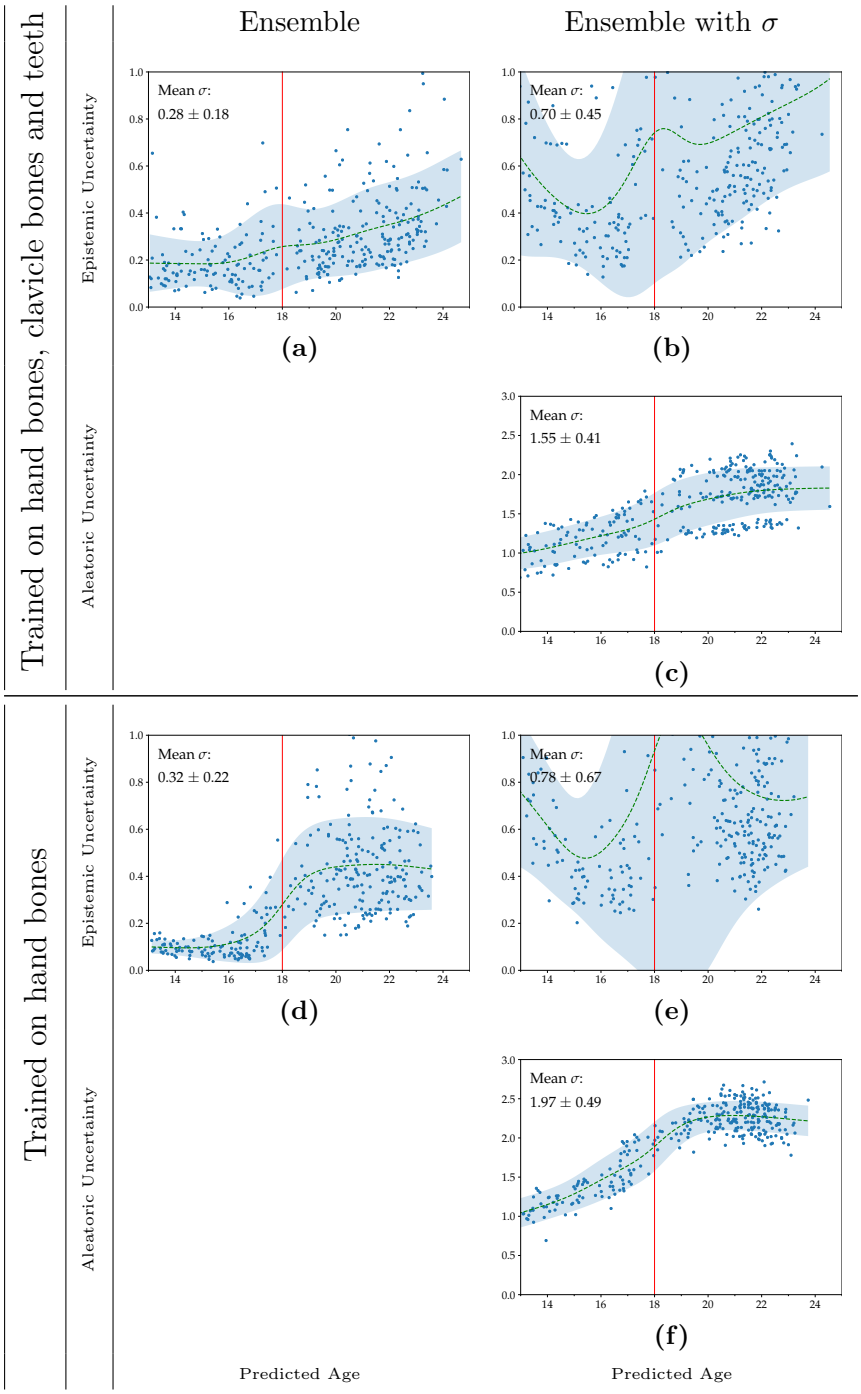


Figure 5.14: Visualization of the extracted uncertainties with the Ensemble, using 2D data.

5.3.3 Monte Carlo-Dropout

3D Results

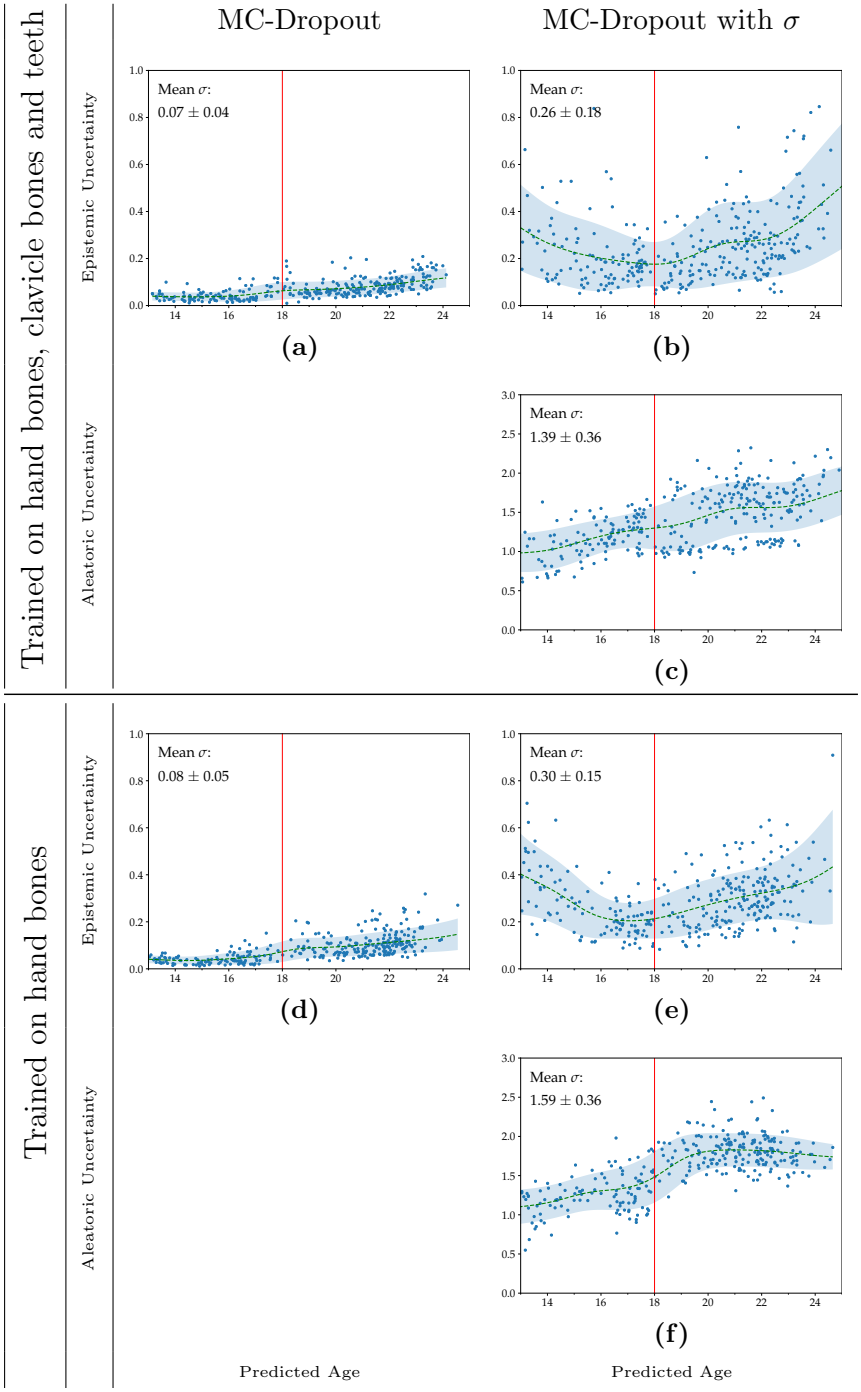


Figure 5.15: Visualization of the extracted uncertainties with the MC-Dropout method, using 3D data.

2D Results

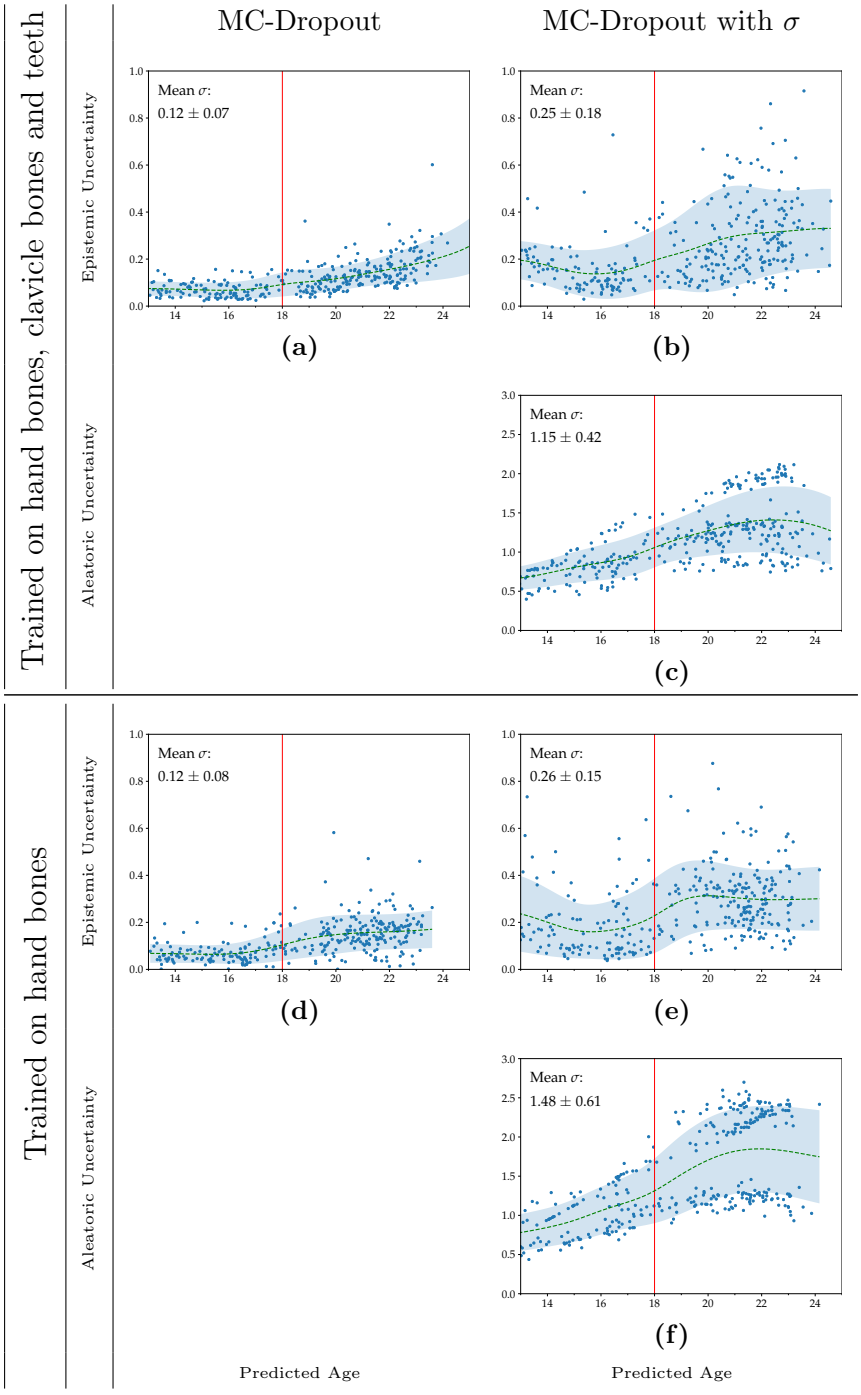


Figure 5.16: Visualization of the extracted uncertainties with the MC-Dropout method, using 2D data.

5.3.4 Bayesian Convolutional Neural Networks

3D Results

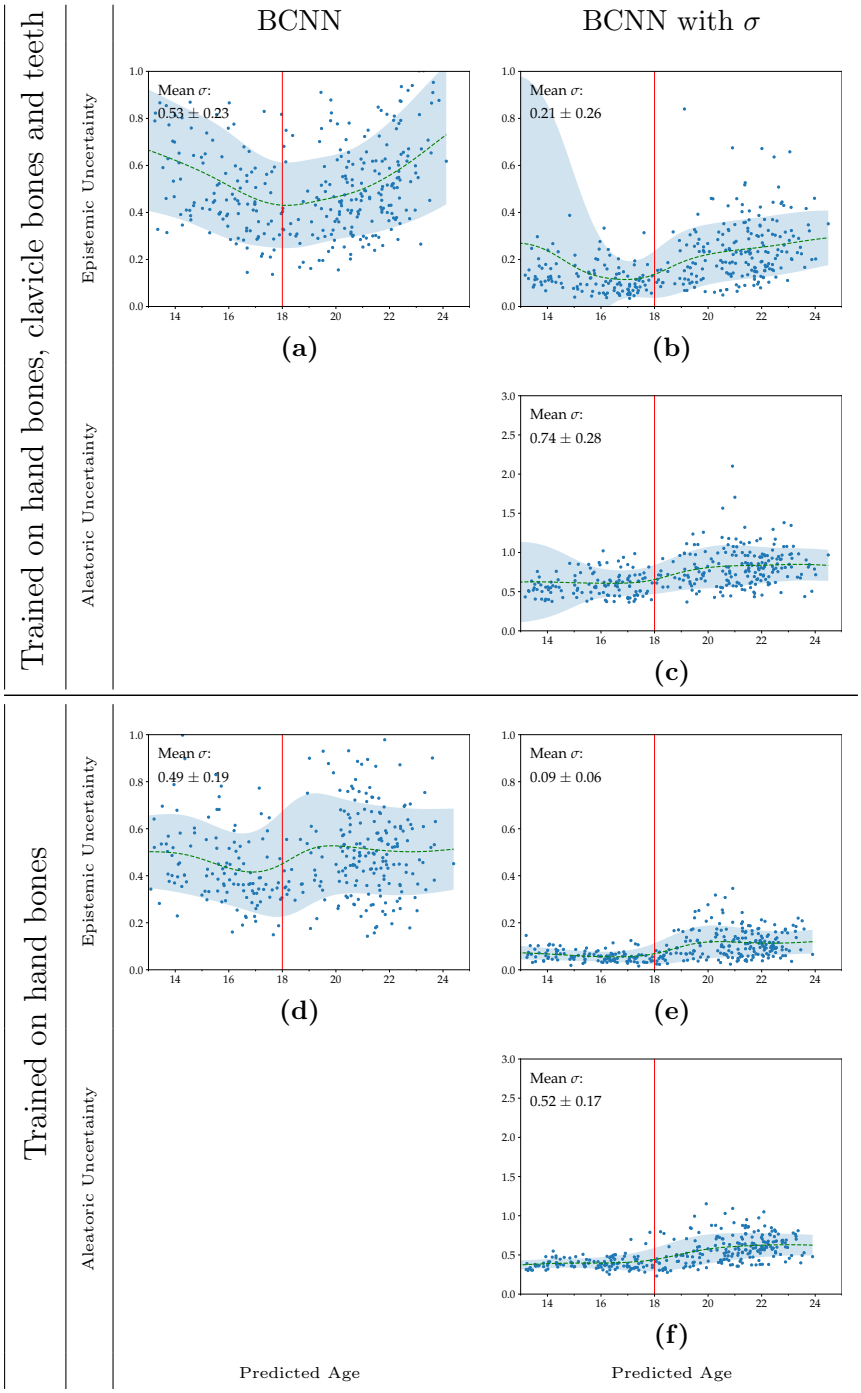


Figure 5.17: Visualization of the extracted uncertainties with the BCNN, using 3D data.

2D Results

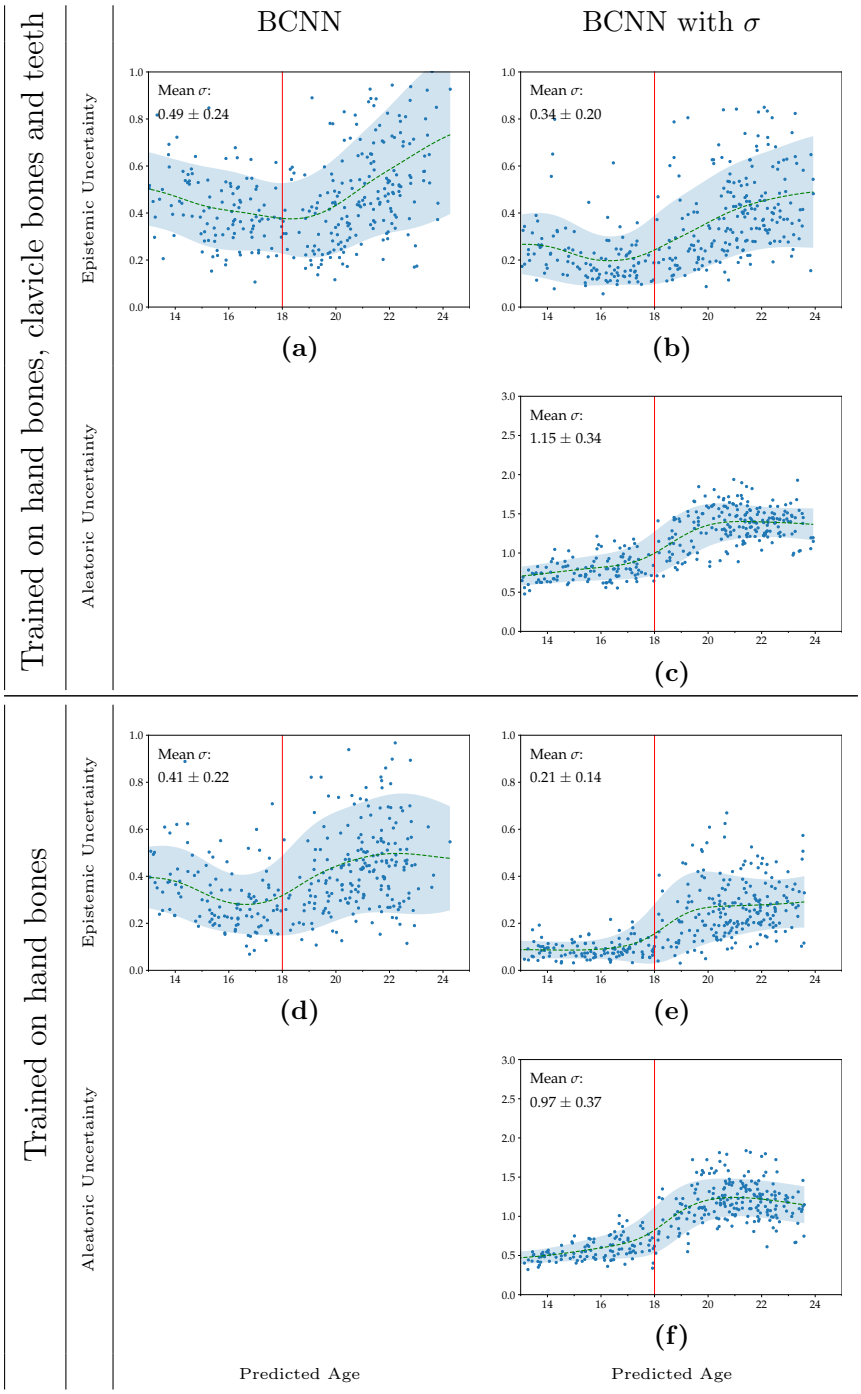


Figure 5.18: Visualization of the extracted uncertainties with the BCNN, using 2D data.

6 Discussion

6.1 Baseline Experiments

In order to visualize the impact of the quality of the data on the *epistemic uncertainty* and *aleatoric uncertainty*, we evaluated Ensembles, the Monte Carlo-Dropout (MC-Dropout) method and the Bayesian Convolutional Neural Network (BCNN) on a simple regression task. This section aims to evaluate and discuss results from those baseline experiments. To start with, we investigate the *aleatoric uncertainty* extracted by the Neural Network (NN), using the Gaussian output layer in Fig. 5.1. In this plot, it becomes visible that the *aleatoric uncertainty* for the trained range shows a small increase in uncertainty over the uncertainty outside this range. This result is in line with the theory that the *aleatoric uncertainty* only captures the noise in the data. The slight increase in uncertainty can be explained since we added noise to the training samples. Samples, which lie outside of the training range, were not captured during training, and the *aleatoric uncertainty* is lowered.

For methods, which model only the *epistemic uncertainty* the uncertainty is constant for samples, which lie inside of the training data and show an increased uncertainty for samples outside of this range. This is the case for the Ensemble (Fig. 5.2a), the *MC-Dropout* method (Fig. 5.2b) and the Bayesian Neural Network (BNN) in Fig. 5.2c. All methods can fit the function accurately, and even show highly similar results for the *epistemic uncertainty* inside the trained range. On the tested samples, which lie outside of this range, we can also see similar results regarding the *epistemic uncertainty*, although the *BNN* predicts different values for samples $x < -5$, compared to the Ensemble and the *MC-Dropout* method. The observed behavior of the *epistemic uncertainty* agrees with the theory that the *epistemic uncertainty* arises from the lack of knowledge about a problem. In with this experiment, the model captured information about the training data, and therefore does not have any information about samples outside of the trained range.

In Fig. 5.3 we can see, that the application of the Gaussian output layer slightly alters the predictive behavior of the function for the Ensemble (Fig. 5.3a) and the *MC-Dropout*

method (Fig. 5.3b). However, all methods still represent the *epistemic uncertainty*, as described before, which is increased for samples, which lie outside of the training range. The *epistemic uncertainty* is even increased for the *MC-Dropout* method, over the results in Fig. 5.2b

Investigating the expressed *aleatoric uncertainty* with the Gaussian output layer, we can see that the Ensemble method (Fig. 5.3d) shows the highest accordance to the result gained from the *NN* in Fig. 5.1. In this example, the *MC-Dropout* method (Fig. 5.3e) failed to capture the *aleatoric uncertainty* since it shows a strong increase of uncertainty towards both borders. While the *BNN* in Fig. 5.3f shows a comparably low *aleatoric uncertainty* for samples before the training range, and a mostly constant *aleatoric uncertainty* for the training range itself, the uncertainty is increased with $x = 10$ to a constant value, even outside of the range. Since this increase occurs close to the border of the trained function, and additionally the function shows increased fluctuations in this range, the *NN* captured this change the overall behavior of the function, by an increased *aleatoric uncertainty*. Since those amplitudes of the function lie close to the border, the model transfers the captured statistics to samples outside of this range, thereby increasing the *aleatoric uncertainty*.

Investigating the results from the uncertainties in Table 5.1 shows that the *epistemic uncertainty* extracted by the Ensemble and the *BNN* are in line with each other and approximate an uncertainty of 0.74 for the range $x < 18$. In the range of $x \geq 18$, this uncertainty is lowered to approximately 0.61. In contrast, the *MC-Dropout* method indicates a strong increase of the *epistemic uncertainty* for $x < 18$ and $x \geq 18$. This suggests that the *MC-Dropout* method overestimates the uncertainty for the baseline example.

In our function, we sampled from a uniform distribution in the range between 18 and 25. The expected standard deviation of this uniform distribution is calculated as $\sigma = \frac{25-18}{\sqrt{12}} = 2.02$. Therefore, the expressed *aleatoric uncertainty* of the methods should predict a standard deviation of 2.02 for $18 < x < 25$. This standard deviation way estimated the closest with the *BNN* with an *aleatoric uncertainty* of 2.00, followed by the Ensemble with 1.83. Again, the *MC-Dropout* method seems to overestimate the *aleatoric uncertainty* with an estimated *aleatoric uncertainty* of 2.44.

The experiments performed on the approximative function in Fig. 4.1b reveal that the extracted *aleatoric uncertainty* indicates a strong dependence on the noise content of the training data. We sampled data for $x > 18$ intentionally from a uniform distribution, which does not hold any information pattern, but the average of those uniform samples. Therefore the *aleatoric uncertainty* predicted by the *NN* Fig. 5.4 shows a low uncertainty for samples bellow $x = 18$ and a strong increase after this point. The trained model seems to predict the average value of the sampled uniform distribution, which is 21.5. This is expected, since no clear pattern is present for samples $x > 18$, and the model predicts the mean value of those samples Due to the noise of the samples $x > 18$, the model seemingly does not capture the statistics of the uniform distribution of the noise, which causes a

substantial increase in *aleatoric uncertainty* after $x = 18$. The Ensemble in Fig. 5.5a only extracts the *epistemic uncertainty*. It can fit the training data in the linearly increasing section and predicts the average value of the uniform distribution for $x > 18$. Between $x = 17$ and 21.5, the model shows an increased *epistemic uncertainty* compared to samples $x < 17$. After $x = 22$, the *epistemic uncertainty* is significantly decreased, compared to the rest of the function.

The *MC-Dropout* method (Fig. 5.5b) shows an overall constant *epistemic uncertainty* for the whole tested range, with the exception of the transition at $x = 18$, where the model shows the same behavior as the Ensemble. The *BNN* in Fig. 5.5c shares overall the same behavior with the Ensemble. Between $18 < x < 21$ the *BNN* shows an increased *epistemic uncertainty* over the rest of the function, and a lower *epistemic uncertainty* for $x > 19$. Moreover, the models are able to predict the average value of the uniform distribution, for $x > 19$. The observations in this paragraph show that the compared methods can represent the *epistemic uncertainty* as expected. In contrast to the Ensemble and the *BNN*, the *MC-Dropout* method indicates an increased *epistemic uncertainty* over the whole range. Furthermore, the *epistemic uncertainty* is increased for $x > 18$, than compared to $x < 18$. For $x < 18$, we generated samples from a Normal distribution with a standard deviation of $\sigma = 0.3$. In contrast to the Ensemble Fig. 5.5a and the *BNN* Fig. 5.5b, the *MC-Dropout* method increases the *epistemic uncertainty* for $x > 18$. Similarly, to our baseline result on the sinusoidal function, we would expect the *epistemic uncertainty* to be constant for the whole range for which training data is available. This increase in *epistemic uncertainty* from the *MC-Dropout* method could indicate that the method does not capture the *epistemic uncertainty* alone, but additionally expresses the *aleatoric uncertainty* which should be increased in this range.

Splitting the uncertainties with the Gaussian output layer in Fig. 5.6 shows the same increase of the *epistemic uncertainty* for $x > 17$, as explained in the paragraph above, expressed by the Ensemble (Fig. 5.6a). Same, as without the usage of the Gaussian output layer the *MC-Dropout* method (Fig. 5.6b) is able to fit the target function. However, the claim that the *MC-Dropout* method fails to clearly separate between both uncertainty is reinforced, since the *epistemic uncertainty* shows the same behavior as in Fig. 5.5b and the *aleatoric uncertainty* in Fig. 5.6e shows an overall increase, compared to the other methods. The *BNN* in Fig. 5.6c shares the same behavior with the Ensemble, but indicates a more stable uncertainty, compared to Fig. 5.5c additionally for samples $x < 10$.

The *aleatoric uncertainty* extracted from the Ensemble (Fig. 5.6d) and the *BNN* (Fig. 5.6f) are in line with the *aleatoric uncertainty* from Fig. 5.4. In both results, the models indicate a comparably low *aleatoric uncertainty* before $x = 17$ with a sudden increase after this point. The uncertainty is constant for the rest of the tested range. Same as already mentioned, the *MC-Dropout* method cannot separate between both uncertainties, therefore it fails the split with the Gaussian output layer. Additionally, the estimated *epistemic uncertainty* and *aleatoric uncertainty* by the *MC-Dropout* method

indicates From all methods, the Ensemble and the *BNN* show the most accordance with the expected behavior of the *epistemic uncertainty* and *aleatoric uncertainty* and are able to capture the expected standard deviation of the uniform distribution the best in the *aleatoric uncertainty*.

To summarize the results of the baseline experiments, we can state that the *aleatoric uncertainty* shows a strong dependence on the noise of the data. The *aleatoric uncertainty* is mostly increased in regions, where the target function shows increased fluctuations or noisy training samples. Moreover, the *epistemic uncertainty* in our experiments is also in line with the hypothesized behavior, namely a constant uncertainty for the range, in which training data is available, and an increased uncertainty for samples outside this range. In our baseline for the approximative function, we found that the *epistemic uncertainty* is mostly constant, but shows an increase in uncertainty, between $17 < x < 21$, where the transition between the linear function and the uniform distribution occurs. The increased *epistemic uncertainty* in this range is caused due to the sudden transition from an explainable function, to the average of the uniform distribution. In this range, the model lacks the complexity to fit this steep gradient and misses out on the necessary feature. Here, the model indicates the lack of knowledge by an increased *epistemic uncertainty*.

The examined approximative function Fig. 4.1b is engineered such that it should represent the contained age information in our hand bones. When training on only images of the hand, the ossification of hand bones causes the saturation of the available information at around 18 years. Therefore, data above this age does not contain any valuable age information. Based on this ossification, we, therefore, expect a constant increase in uncertainty for subjects older than 18 years.

To describe this behavior in our approximation function, we draw samples from a uniform distribution for $x > 18$. Based on this hypothesis, we expect a similar behavior of the uncertainties in our age estimation experiments, as to the results found in this section. By additionally introducing images of clavicle bones and teeth to the training data, the whole age range is covered, and the saturation should vanish.

6.2 Age Estimation

This section is focussed on the discussion and evaluation of the results archived for our age estimation task. To begin with, we compare our results of the 3D experiments using Convolutional Neural Network (CNN), trained on hand bones, clavicles and teeth (1.06 ± 0.77 years), we archive a similar performance in terms of the Mean Absolute Deviation (MAD) to the results of Štern et al. [52] (1.01 ± 0.74 years). With Ensembles trained on the combination of sites, we are able to surpass this performance (0.98 ± 0.77 years). In general, the results indicate that the introduction of imaging volumes instead of only the 2D middle slices increases the ability to fit the data, which can be seen by

the overall reduced *MAD*. Furthermore, the introduction of additional sites does not necessarily reduce the *MAD*, but rather the standard deviation of the predictions, which can be seen upon the comparison of the results in Table 5.2 and Table 5.3.

6.2.1 Convolutional Neural Network

Evaluating the extracted *aleatoric uncertainty* from conventional *CNNs*, applying the Gaussian output layer (Fig. 5.11b), we can see that the method trained on hand bones shows an increased uncertainty for subjects older than 18. The model estimates this *aleatoric uncertainty* with a standard deviation of 1.8 years. Compared to younger subjects, we can see an increasing uncertainty, which is on average 1.4 years. Introducing additional sites to the training provides age information for the range of subjects older than 18 years. This can be seen in Fig. 5.11a, where the *aleatoric uncertainty* steadily increases with the subjects age. The results from our *CNN*, using the output layer, approximate the assumed error due to the biological variation of one year.

With 3D data, the *aleatoric uncertainty* was captured successfully from the *CNN*. Using 2D data instead shows that the same effect can be found in the results, although it is less pronounced, which can be seen in Fig. 5.12b. Additionally, the model tends to cluster the predicted *aleatoric uncertainty* of the samples. The same effect is present when training on the combination of sites in Fig. 5.12a. The application of the Gaussian output layer shows that the *CNN* is able to represent the expected behavior of the *aleatoric uncertainty* although this effect is more pronounced with 3D data.

6.2.2 Ensemble

The *epistemic uncertainty* for the Ensemble method, trained on 3D data is visualized in Fig. 5.13d. When using only hand bones for training, the age information saturates due to the ossification of hand bones around the age of 18 years. Since the model does not separate the uncertainties, both are included in the expressed *epistemic uncertainty*. Therefore we can see the sudden increase in *epistemic uncertainty* around 18 years. Moreover, the *epistemic uncertainty* shows an increased standard deviation of the uncertainty predictions at 20 years. Compared to our baseline results for the Ensemble in Fig. 5.5a, we see the same expression of an increased uncertainty, which is shifted to around 20 years. Interestingly, the same saturation effect can also be found, when training the Ensemble on the combination of the hand bones, clavicles, and teeth, as shown in Fig. 5.13a. This could indicate, that the Ensemble method does not successfully combine the age information from all the different sites in 3D, but instead focuses on the age information present in hand bones only, therefore neglecting available information in the clavicle bones and teeth.

The application of the Gaussian output layer causes the *epistemic uncertainty* of the model in Fig. 5.13e to not express the label uncertainty, but instead generates noisy predictions, with an increased standard deviation. A similar effect can be seen in the

baseline (Fig. 5.6a), where the *epistemic uncertainty* in the saturated age range shows an increased uncertainty. This is the case for the Ensemble, with the Gaussian output layer for training on hand bones (Fig. 5.13e) and the combination of hand bones, clavicles and teeth (Fig. 5.13b).

The extracted *aleatoric uncertainty* (Fig. 5.13f) successfully visualizes the label uncertainty, which was present in the *epistemic uncertainty* before the split in Fig. 5.13d. In subjects older than 18 years, the *aleatoric uncertainty* is estimated with a standard deviation of 1.97 years, which is in line with the baseline experiments. Below this range, the *aleatoric uncertainty* is in averaged to 1.14 years and therefore agrees with the assumed error underlying the age estimation task of 1 year.

Training the Ensemble on hand bones, clavicles and teeth instead causes the same saturation behavior for the *aleatoric uncertainty* (Fig. 5.13c) and a noisy *epistemic uncertainty* (Fig. 5.13c), as described before. This distinct behavior between the extracted *epistemic uncertainty* and *aleatoric uncertainty*, therefore, suggests that both uncertainties are presumably decoupled. However, since the introduction of clavicles and teeth does not cause the saturation behavior of the uncertainty to vanish in Fig. 5.13a and Fig. 5.13c, we suggest, that the Ensemble does not use the available age information to its full extend. It rather neglects the essential age information available in the data.

Investigating the *epistemic uncertainty* for the Ensemble trained on 2D data, we found that the Ensemble can combine the age information of the different sites into one final age estimate, as opposed to the 3D experiments. Therefore, the saturation behavior, when trained on hand bones, can be seen in Fig. 5.14d, which is not present anymore, when training on the combination of the sites, as depicted in Fig. 5.14a. In the latter case, we can see that the *epistemic uncertainty* increases continuously with the subject's age, instead of the saturation. This behavior is expected since the additional sites extend the covered age range, and therefore cause the saturation to vanish. Since the Ensemble on 3D data is not able to use the age information of all trained sites to its full extension, while with 2D data, it can distinguish the sites, we suggest that the Ensemble method is highly dependent on the dimensionality of the data.

Using the Gaussian output layer on the Ensemble with 2D data shows the same behavior for the *epistemic uncertainty* for both sets of training data (Fig. 5.14d and Fig. 5.14a), as described for 3D data above. In contrast to the 3D experiments, here, the Ensemble method can fully utilize the training data of hand bones, clavicles and teeth, which causes the saturation effect in Fig. 5.14c to vanish.

The same observation of the *aleatoric uncertainty* and the *epistemic uncertainty* can also be found in the 2D experiments for Ensembles, for hand bones (Fig. 5.14f and Fig. 5.14c) and hand bones, clavicles and teeth (Fig. 5.14e and Fig. 5.14b). However, with 2D data, the extracted *aleatoric uncertainty* for younger subjects is approximated with 1.55 years, which indicates a slight increase over the 3D results. These results show a different expressive behavior and a consistent representation of the *aleatoric uncertainty*

of the Ensemble using the Gaussian output layer. This indicates that the Ensemble is successfully able to utilize the Gaussian output layer with 2D data in order to split the *epistemic uncertainty* from the label uncertainty in a meaningful way.

6.2.3 Monte Carlo-Dropout

While the Ensemble fails to merge the age information with 3D data, when using all bone sites (Fig. 5.13a), the *MC-Dropout* method captures the label uncertainty, when trained on 3D hand bone data and indicates a slight saturation behavior in Fig. 5.15d. Similarly, we can see the sudden increase in the *epistemic uncertainty* of the baseline Fig. 5.5b at $x = 18$, which is immediately reduced again to the same constant uncertainty. Introducing additional sites into the training data, diminishes the saturation effect and instead causes a steady increase of the *epistemic uncertainty*, as seen in Fig. 5.15a.

Interestingly, using 2D data for the *MC-Dropout* method causes the *epistemic uncertainty* to be increased, but to keep the same expression of the label uncertainty, as in the 2D case. Here, the *epistemic uncertainty* visualizes the expression of the label uncertainty better for training on hand bones (Fig. 5.16d) and the combination of sites (Fig. 5.16a).

Upon splitting the uncertainties with the Gaussian output layer we can see, that the label uncertainty is now only contained in the *aleatoric uncertainty* for both combinations of training data (Fig. 5.15f and Fig. 5.15c). The estimated *aleatoric uncertainty* for training on hand bones estimates a label uncertainty of 1.31 years for subjects younger than 18 years. In the saturated region, this uncertainty is increased to 1.97 years. The *epistemic uncertainty* in those cases (Fig. 5.15e and Fig. 5.15b) provide a similar behavior, since they indicate an increased *epistemic uncertainty* towards the borders. Those results agree with our baseline result of the *epistemic uncertainty* in Fig. 5.6b, where the uncertainty is also increased towards the borders. The estimated *aleatoric uncertainty* for the baseline is approximately 1 and increased to approximately 3. However, training on all bone sites causes the overall *epistemic uncertainty* to be increased over training on hand bones only.

For the *MC-Dropout* method, implementing the Gaussian output layer in the 2D case shows on first impression, that the *aleatoric uncertainty* for our hand bones data in Fig. 5.16f contains the label uncertainty. However, the *aleatoric uncertainty* for the combined training data in Fig. 5.16c indicates the same expression. Therefore, we suggest that both models do not contain the expected label uncertainty. Instead, this method shows a tendency to cluster predictions. This result does not agree with the baseline experiments since the baseline is able to extract the *aleatoric uncertainty* with the Gaussian output layer, which can be seen in Fig. 5.6e.

The *epistemic uncertainty* when trained on hand bones (Fig. 5.16e), depicts the saturation of subjects older than 18 years, with an increased uncertainty towards the lower border. In contrast, the *epistemic uncertainty* indicates a steady increase with the subject age, when trained on the combination of sites (Fig. 5.16b) Similarly to our baseline experiment in Fig. 5.5b, here the *epistemic uncertainty* is mostly constant, but only shows

an increased uncertainty for subjects younger than 18 years. The Gaussian output layer causes the uncertainties to be increased in both our baseline experiment, as well as in the age estimation results.

To summarize, the *MC-Dropout* method can capture the label uncertainty for 2D and 3D data. Using the Gaussian output layer visualizes, that this method is able to perform this split between both uncertainties only with 3D data and fails to do so with 2D data. Our age estimation results are mostly in line with those of the baseline experiments.

6.2.4 Bayesian Convolutional Neural Network

In contrast to the described behaviour of the *epistemic uncertainty* in Ensembles and with the *MC-Dropout* method, the results of the *BCNNs* indicate for both 2D and 3D experiments an increased *epistemic uncertainty* in the age below 18 years, for hand bones and the combination of sites (see Fig. 5.18a and Fig. 5.17a).

When training the *BCNN* on 3D hand bone data, we found that the *epistemic uncertainty* is increased for subjects younger than 18 years (Fig. 5.17d). A similar effect can be seen in the 2D Ensemble experiment, trained on hand bones, in Fig. 5.18d. For subjects older than 18 years, we can see the saturation behavior in both cases, which is expected, when training on hand bones. This increased uncertainty stands in contrast to our baseline experiment in Fig. 5.5c, where we see an overall constant behavior for the trained samples within the target function. However, we can see in our age estimation result, that the increase and immediate reduction of the *epistemic uncertainty* agree with the baseline results.

The introduction of clavicles and teeth to the training data causes the *epistemic uncertainty* to adopt the expected behavior for subject older than 18 years. This can be seen in both our 3D and 2D experiments (Fig. 5.17d and Fig. 5.17a). Again, the *epistemic uncertainty* indicates an increased uncertainty towards the border for subjects younger than 18 years. This increased uncertainty could be explained, since the *BCNN* is restricted in its ability to represent the uncertainty by choice of the prior distribution. Therefore, the model captures the statistics around 18 years better, but neglect samples in the range bellow, suggesting an increased uncertainty in this range.

The application of the Gaussian output layer to *BCNNs* with 3D data, causes the *aleatoric uncertainty* to overall being significantly smaller, than with the other methods. Training on hand bones leads to an estimated label uncertainty of 0.41 years, which is increased to 0.59 years for mature subjects. Training the *BCNN* on 3D hand bone data, causes the *aleatoric uncertainty* to start increasing at around 17 years and to saturate at around 22 years, which can be seen in Fig. 5.17f. However, the same effect can be found in the *epistemic uncertainty* in Fig. 5.17e, where the saturation is much more dominant. In comparison, when training on the combination of sites, the *aleatoric uncertainty* (Fig. 5.17c) indicates the same behavior as for the *aleatoric uncertainty*, when training on only hand bones (Fig. 5.17e). The introduction of additional sites seems to cause an over-

all increased *epistemic uncertainty*, which can be found in Fig. 5.17b. Here, the *epistemic uncertainty* indicates a significant increase, especially for subjects younger than 16 years. Apart from the *aleatoric uncertainty*, the found results do not agree with the baseline experiments, since the model in the baseline was able to clearly separate the *aleatoric uncertainty* from the *epistemic uncertainty* (Fig. 5.6f and Fig. 5.6c). Instead, we found that the *epistemic uncertainty* captures the label uncertainty and is highly similar to the *aleatoric uncertainty*, suggesting that the uncertainty is not split successfully.

Using 2D data instead, we found a very similar behavior for the *epistemic uncertainty* (Fig. 5.18b and Fig. 5.18b) and *aleatoric uncertainty* (Fig. 5.18c and Fig. 5.18c), while the uncertainties show an overall increase over the uncertainties in the 3D experiments. In the 2D case, the estimated label uncertainty for minors is 0.66 years in average, where this uncertainty is increased to 1.19 years for adults. The same behavior suggests that the increased information content in 3D data can be utilized to lower the overall uncertainty. However, similarly to the 3D experiments, the 2D results depict a saturation of *epistemic uncertainty* when trained on hand bones. This behavior recommends that the *epistemic uncertainty* contains the label uncertainty. This statement is reinforced since the introduction of additional sites in Fig. 5.18b causes the *epistemic uncertainty* to show a behavior that would be expected for this training data. Here the *epistemic uncertainty* increased for subjects older than 18 years, while training on only hand bones, clearly shows the saturation of age information in Fig. 5.18e. Although the saturation behavior can also be found in the *aleatoric uncertainty* for both training on hand bones and the combination of sites, it seems to be less dominant and independent on the trained site. In the described 2D and 3D experiments, the split *aleatoric uncertainty* and *epistemic uncertainty* are correlated. This suggests that the application of the Gaussian output layer to *BCNNs* is not able to separate both uncertainties successfully.

While in the baseline experiments, the *epistemic uncertainty* is constant towards the borders (Fig. 5.5c), the increase is much more dominant in the age estimation task (Fig. 5.17d). Here, the extracted *aleatoric uncertainty* is approximated with a standard deviation of 0.5 for $x < 18$ and approximately 2 afterward. Our age estimation results suggest that the methods are not able to estimate the included label uncertainty successfully, which is in line with the results from our baseline.

Same, as for the *MC-Dropout* method, the results obtained from the experiments performed with the *BCNNs* applying the Gaussian output layer, therefore suggest, that this method is not able to split both uncertainties. Additionally, when comparing the results between the *MC-Dropout* method and the *BCNN*, we found that the *aleatoric uncertainty* for both methods does not explicitly contain the label uncertainty. Furthermore, both methods show an increased uncertainty towards the borders, which can for example be found in Fig. 5.18b and Fig. 5.15e.

By further investigating the uncertainties in Table 5.2, we found that the *MC-Dropout* method reaches the lowest *epistemic uncertainty* when trained on 3D hand images (0.09).

However, in our baseline experiments, the *MC-Dropout* method indicated the overall highest *epistemic uncertainty* (0.95), which is increased by the application of the Gaussian output layer (1.12). The same case occurs for the age estimation results on 2D data, given in Table 5.3.

We hypothesized that Fig. 4.1b approximates the function, which describes the age information content in our hand images. In our baseline, we saw that the Ensemble and the *BNN* showed the highest agreement of the standard deviation of 0.3 for $x < 18$. The same methods also captured the expected standard deviation of 2.02 for $x > 18$ in the *aleatoric uncertainty*.

Moreover, the estimated *epistemic uncertainty* in our age estimation task agrees the most to the baseline for the Ensemble method and the *BNN*. The expected standard deviation for subjects older than 18 years was estimated the best by the Ensemble (1.97) and the *MC-Dropout* method (1.79). Furthermore, all methods, except for the *BCNN* approximated the underlying error of the label uncertainty of 1 year. In contrast to our baseline experiments, we mostly found that the *aleatoric uncertainty* for subjects younger than 18 years is not constant, but instead increases with the subject age until 18 years. However, since the *MC-Dropout* method failed to perform a successful split of both uncertainties, we suggest, that the Ensemble method showed the most success in estimating the hypothesized uncertainties in our age estimation experiments.

6.3 Final Remarks

This section focuses on the discussion of problems of the individual methods themselves and to propose potential improvements over problems that have been encountered in this thesis.

A significant challenge in this work involves the evaluation of the quality of the predictive uncertainty. In this thesis, we are able to consider the deviation of the Biologial Age (BA) from the Chronological Age (CA) as the label uncertainty. An alternative way of evaluating the quality of the predicted uncertainty, to consider the generalization to a specific domain. Thus shifting the domain of input images, the predicted uncertainty should be increased since the data of the out-of-domain data is not included in the training data distribution [28]. Our baseline experiments investigated this proposal.

In this work only, the *aleatoric uncertainty* and *epistemic uncertainty* were considered, based on literature in current research [21, 27, 48]. Osband [41] further makes a distinction between risk and uncertainty, saying that risk is inherent to the stochastic in a model while the uncertainty provides the confusion about which model parameters apply.

For our experiments, we used Ensembles, consisting of $T = 5$ models in total, in order to get a measure of the *epistemic uncertainty*, instead of $T = 20$ forward passes, as done in the other methods. While it would be possible to extend the number of trained models, this would have increased the time needed for training enormously, since we were not able to train the models at once, due to the increased memory consumption for 3D data.

An important point to note regarding the *MC-Dropout* method is that a dropout probability for the convolutional layers, which, if chosen too high, can cause a severe performance decrease. Even dropping just 10% of the neurons could remove activations of significant features with an image volume of size 44^3.

In the optimization process of the *BCNN*, similarly to Kingma et al. [24], we found it to be necessary that both parts of the Evidence Lower Bound (ELBO) loss are in the same vale range, to ensure convergence. Therefore, we scaled the contribution of the Kullback Leibler (KL) term in Eq. (3.7) by the factor $\lambda = 10^{-6}$. While this scaling enabled us to train the *BCNN*, it could also introduce an error to the model, by assigning the *KL* loss less importance.

While not only the range of both parts is significant to the learning performance, the learning rate additionally affects the optimization of the *KL* loss. An increased learning rate causes the *KL* loss to converge faster than the data term and vice versa. Minimizing the *ELBO* loss increases the fit to the data while keeping the approximated posterior as close as possible to the assumed prior distribution. In the case of the faster convergence of the *KL* loss, the model mainly increases the weight distributions similarity to the prior distribution and neglects the fit of the data for the most part. Therefore the model can not learn to represent the input data since its range of possible descriptive functions is decreased.

Apart from that, a faster than the *KL* loss converging data term indicates that the model learns an accurate representation of the data, but loses its ability to generalize to a large part of the domain. In this case, satisfying the assumed prior distribution is neglected in the calculation of the posterior distribution. This neglection defies Bayesian theory and, in theory, leads to the same deterministic behavior as with conventional *CNNs*. As a possible approach to overcome this potential problem, Blundell [5] proposed to weight the *KL* loss stronger at the beginning of the training and reduce its contribution with ongoing epochs.

In the Reparameterization layers, the prior distribution over the model weights is selected to be a Gaussian distribution. This simplicity of the Gaussian distribution could reduce the ability to generalize to the data since it constricts the model complexity. Interchanging the Gaussian distribution with a more complex distribution could increase the performance and the ability to represent the uncertainty of the data. This will increase the difficulty of the optimization problem further and increase the difficulty for the model to converge towards a minimum.

In this thesis, we only used one set of sampled model weights per iteration to approximate the posterior distribution. Graves [14] has shown that it is sufficient to use only one set of weights as an approximation. However, especially after initialization, those samples underlie a significant variation since the approximate posterior distributions are initialized with a mean of 0 and a standard deviation of 1. Therefore, it can be assumed that averaging the activations over multiple sets of weights can improve the convergence speed and predictive performance, especially in the early stages of training. Increasing the

amount of sampled sets of weights causes a tremendous increase in memory consumption, due to the static formulation of the computational graph in *TensorFlow*. Due to hardware restrictions, we were not able to follow this approach.

As an alternative to the utilized Reparameterization layers, another implementation of the probabilistic convolutional and dense layers exist. Wen et al. [58] proposed their method name *Flipout approximation*, which is implemented in the Flipout convolutional and dense layers in the Tensorflow probability library [8]. Those Flipout layers have shown to reduce the variance of the calculated gradients and lead to a potential increase in convergence speed when compared to the used Reparameterization layers [58].

7 Conclusion

In this thesis, we evaluated four different methods on a multi-factorial age estimation task regarding their ability to fit the data and the quality of the extracted uncertainty. In order to investigate the impact of the information content of the data on the *epistemic uncertainty* and *aleatoric uncertainty*, we performed baseline experiments on two different functions.

The investigated methods encompass Ensembles, Monte Carlo-Dropout (MC-Dropout) and Bayesian Convolutional Neural Network (BCNN) for the extraction of the *epistemic uncertainty* and the Gaussian output layer, for the split into *epistemic uncertainty* and *aleatoric uncertainty*. Our baseline experiments provided insight into the influence of the information content of the data on both uncertainties. We expected an estimated standard deviation of the *aleatoric uncertainty* for $x < 18$ of 0.3. The Neural Network (NN) predicted an *aleatoric uncertainty* of 0.30, the Ensemble 0.28 and the Bayesian Neural Network (BNN) predicted 0.39. Those methods were able to predict the expected uncertainty. The *MC-Dropout* method failed to do so and overestimates the *aleatoric uncertainty* with 1.22.

Furthermore, the *NN*, the Ensemble and the *BNN* agreed with their predicted *epistemic uncertainty* for the individual ranges with an uncertainty of around 0.73. Similarly, as with the *aleatoric uncertainty*, the *MC-Dropout* overestimates the *epistemic uncertainty* and fails to represent the same accuracy. Based on our baseline results, we can state that the *epistemic uncertainty* represents the ability of the model to capture data. We suggest, based on the estimated uncertainty and the behavior of the *epistemic uncertainty* and the *aleatoric uncertainty* that the Ensemble performed the best on this baseline experiment. It provided a small *epistemic uncertainty* over the whole range and additionally was able to capture the expected uncertainties the best.

Our experiments showed the expected results regarding the behavior of the expressed uncertainties. The *epistemic uncertainty* should remain constant for the range in which training data is available, while the *aleatoric uncertainty* should increase in the case where data is present, but the labels are noisy. We were able to verify that our engineered

approximative function can describe the age information present in our hand bone data, although the chosen standard deviation was chosen too small with 0.3. We hypothesized that our approximative function could represent the true age information content present in our hand bones. We were able to verify that our engineered approximative function can describe the age information present in our hand bone data, although the chosen standard deviation was chosen too small with 0.3.

Our age estimation results showed that the Convolutional Neural Network (CNN), applying the Gaussian output layer can represent the expected saturation when trained on hand bones in the predicted *aleatoric uncertainty*. While this effect can be seen with 3D data, it vanishes with 2D data and introduces grouping of the predictions The *NN* estimated an *aleatoric uncertainty* of 1 year for minors, which increases with the age of the subject. After 18 years, the estimated *aleatoric uncertainty* approximates the expected standard deviation of 2.02 of the uniform distribution in the saturated range.

The result of the Ensemble method, trained on hand bones, visualizes the saturation effect and estimate the expected standard deviation correctly with the *aleatoric uncertainty*. The introduction of hand clavicle and teeth causes the label uncertainty to be increased with the age of the subject. Bellow 18 years, the *aleatoric uncertainty* is estimated with 1 year, which is in line with the expected underlying label uncertainty due to the biological variation. The Ensemble can perform a successful split of the uncertainty, and also shows the label uncertainty in the *epistemic uncertainty* without the Gaussian output layer.

The *MC-Dropout* method indicates that the extracted *aleatoric uncertainty* agrees with the theoretical development of the label uncertainty and shows the saturation for training on hand bones with 3D data. However, using 2D data causes this expressed behavior to be less pronounced. The *MC-Dropout* method fails to estimate the expected uncertainties, as compared to the *NN* and the Ensemble. Moreover, the *MC-Dropout* method is not able to perform a successful split between both uncertainties, using the Gaussian output layer.

In our *BCNN* results, we found the expected increase of uncertainty, with increased subject age. However, the method seems to underestimate the label uncertainty in the experiments with an *aleatoric uncertainty* of approximately 0.6. Moreover, this method seems to fail the split of both uncertainties, using the Gaussian output layer, since both uncertainties are correlated. While the saturation effect is present in the experiments with 3D data, it is more pronounced with 2D data. Furthermore, the *BCNN* fails to perform the split of the uncertainties, with 2D data. However, the Ensemble indicates a successful split. Therefore, each method shows different results regarding the agreement of the expressed behavior.

We were able to confirm the expected behavior of the *epistemic uncertainty* and the *aleatoric uncertainty* in our baseline experiments. The results from the baseline indicate that all methods, except for the *MC-Dropout* can capture the statistics of the underlying noise with success, while the *MC-Dropout* method overestimates this uncertainty. We ver-

ified that our assumed baseline function approximates the function, which describes the age information content in our age estimation data. Similar to the baseline, the age estimation results demonstrate that not all compared methods capture the label uncertainty. The gained results indicate a strong dependence of the expressed behavior and value of the uncertainty on the dimensionality of the used data. Overall, the expected saturation is pronounced the most when using 2D data. Based on the agreement of the estimated *aleatoric uncertainty* and the expressed behavior of the uncertainty, we conclude this thesis by stating that Ensembles in combination with the Gaussian output layer provided the best approximation of the underlying label uncertainty in this age estimation task.

A

List of Acronyms

AGFAD	Study Group on Forensic Age Diagnostics
BA	Biologial Age
BCNN	Bayesian Convolutional Neural Network
BNN	Bayesian Neural Network
CA	Chronological Age
CNN	Convolutional Neural Network
CT	Computed Tomography
DCNN	Deep Convolutional Neural Network
DL	Deep Learning
ELBO	Evidence Lower Bound
GP	Greulich-Pyle Atlas method
KL	Kullback Leibler
MAD	Mean Absolute Deviation
MC	Monte Carlo
MC-Dropout	Monte Carlo-Dropout
ML	Machine Learning
MR	Magnetic Resonance
MRI	Magnetic Resonance Imaging
NN	Neural Network
ReLU	Rectified Linear Unit
SeLU	Selective Linear Unit
TW2	Tanner-Whitehouse

B

List of Publications

My work at the Institute of Computer Graphics and Vision led to the following publication, which was accepted at the *Medical Imaging meets NeurIPS* workshop in 2019:

S. Eggenreich, C. Payer, M. Urschler, and D. Štern. Variational Inference and Bayesian CNNs for Uncertainty Estimation in Multi-Factorial Bone Age Prediction. *arXiv*, abs/2002.10819, 2020

Bibliography

[1] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, et al. Tensorflow: A System for Large-scale Machine Learning. In *12th USENIX Symposium on Operating Systems Design and Implementation*, pages 265–283, 2016. (page 30)

[2] P. Baldi, P. A. Sadowski, and D. O. Whiteson. Searching for Exotic Particles in High-energy Physics with Deep Learning. *Nature Communications*, 5:4308, 2014. (page 3)

[3] U. Baumann, R. Schulz, W. Reisinger, A. Heinecke, A. Schmeling, and S. Schmidt. Reference study on the time frame for ossification of the distal radius and ulnar epiphyses on the hand radiograph. *Forensic Science International*, 191(1-3):15–18, 2009. (page 1)

[4] C. M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006. (page 8, 9, 10)

[5] C. Blundell, J. Cornebise, K. Kavukcuoglu, and D. Wierstra. Weight Uncertainty in Neural Network. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 1613–1622. PMLR, 2015. (page 3, 12, 17, 18, 65)

[6] W. de Back, S. Seurig, S. Wagner, B. Marré, I. Roeder, and N. Scherf. Forensic Age Estimation with Bayesian Convolutional Neural Networks Based on Panoramic Dental X-ray Imaging. In *International Conference on Medical Imaging with Deep Learning*, 2019. (page 2)

[7] A. Der Kiureghian and O. Ditlevsen. Aleatory or Epistemic? Does It Matter? *Structural safety*, 31(2):105–112, 2009. (page 3)

[8] J. V. Dillon, I. Langmore, D. Tran, E. Brevdo, S. Vasudevan, D. Moore, B. Patton, A. Alemi, M. D. Hoffman, and R. A. Saurous. TensorFlow Distributions. *arXiv*, abs/1711.10604, 2017. (page 13, 66)

[9] J. Dvorak, J. George, A. Junge, and J. Hodler. Application of MRI of the Wrist for Age Determination in International -17 Soccer Competitions. *British Journal of Sports Medicine*, 41(8):497–500, 2007. (page 2)

[10] S. Eggenreich, C. Payer, M. Urschler, and D. Štern. Variational Inference and Bayesian CNNs for Uncertainty Estimation in Multi-Factorial Bone Age Prediction. *arXiv*, abs/2002.10819, 2020. (page)

[11] Y. Gal and Z. Ghahramani. Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. *arXiv*, abs/1506.02158, 2015. (page 3, 4, 12, 21)

[12] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1050–1059, New York, New York, USA, 20–22 Jun 2016. PMLR. (page 3, 4, 20, 21)

[13] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. (page 8, 9, 10, 11, 12)

[14] A. Graves. Practical Variational Inference for Neural Networks. In *Advances in Neural Information Processing Systems*, volume 24. 2011. (page 65)

[15] W. W. Greulich and S. I. Pyle. *Radiographic Atlas of Skeletal Development of the Hand and Wrist*. Stanford University Press, 1959. (page 1, 2)

[16] A. Gupta, H. Wang, and M. Ganapathiraju. Learning Structure in Gene Expression Data Using Deep Architectures, with an Application to Gene Clustering. In *Proceedings of the IEEE International Conference on Bioinformatics and Biomedicine*, pages 1328–1335, 2015. (page 3)

[17] K. He, X. Zhang, S. Ren, and J. Sun. Delving Deep into Rectifiers: Surpassing Human-level Performance on ImageNet Classification. In *IEEE International Conference on Computer Vision*, pages 1026–1034, 2015. (page 30)

[18] E. Hillewig, J. de Tobel, O. Cuche, P. Vandemaele, M. Piette, and K. Verstraete. Magnetic Resonance Imaging of the Medial Extremity of the Clavicle in Forensic Bone Age Determination: A New Four-minute Approach. *European Radiology*, 21(4):757–767, 2011. (page 2)

[19] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. In *Proceedings of the International Conference on Machine Learning*, pages 448–456, 2015. (page 12)

[20] T. S. Jaakkola and M. I. Jordan. Bayesian Parameter Estimation Via Variational Methods. *Statistics and Computing*, 10(1):25–37, 2000. (page 18)

[21] A. Kendall and Y. Gal. What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in Neural Information Processing Systems 30*, pages 5574–5584. 2017. (page 3, 4, 21, 23, 64)

[22] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. (page 10, 30)

[23] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In *International Conference on Learning Representations*, 2014. (page 13, 19, 20)

[24] D. P. Kingma, T. Salimans, and M. Welling. Variational Dropout and the Local Reparameterization Trick. In *Advances in Neural Information Processing Systems*. Curran Associates, Inc., 2015. (page 20, 65)

[25] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter. Self-Normalizing Neural Networks. In *Advances in Neural Information Processing Systems*, pages 971–980, 2017. (page 9)

[26] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, pages 1097–1105. 2012. (page 23)

[27] Y. Kwon, J. Won, B. Kim, and M. C. Paik. Uncertainty Quantification Using Bayesian Neural Networks in Classification: Application to Biomedical Image Segmentation. *Computational Statistics and Data Analysis*, 142, 2020. (page 17, 64)

[28] B. Lakshminarayanan, A. Pritzel, and C. Blundell. Simple and Scalable Predictive Uncertainty Estimation Using Deep Ensembles. In *Advances in Neural Information Processing Systems 30*, 2017. (page 4, 23, 64)

[29] D. B. Larson, M. C. Chen, M. P. Lungren, S. S. Halabi, N. V. Stence, and C. P. Langlotz. Performance of a Deep-Learning Neural Network Model in Assessing Skeletal Maturity on Pediatric Hand Radiographs. *Radiology*, 287 1:313–322, 2017. (page 2)

[30] F. Laumann, K. Shridhar, and A. L. Maurin. Bayesian Convolutional Neural Networks. abs/1806.05978, 2018. (page 4)

[31] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. (page 10)

[32] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Backpropagation applied to handwritten zip code recognition. *Neural Computation*, 1(4):541–551, 1989. (page 10)

[33] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *Proceedings of the International Conference on Machine Learning*, volume 30, page 3, 2013. (page 9)

[34] D. J. C. MacKay. A Practical Bayesian Framework for Backpropagation Networks. *Neural Computation*, 1992. (page 4)

[35] D. J. MacKay. *Bayesian methods for adaptive models*. PhD thesis, California Institute of Technology, 1992. (page 17)

[36] D. D. Martin, J. M. Wit, Z. Hochberg, R. R. van Rijn, O. Fricke, G. Werther, N. Cameron, T. Hertel, S. A. Wudy, G. Butler, H. H. Thodberg, G. Binder, and

M. B. Ranke. The Use of Bone Age in Clinical Practice - Part 2. *Hormone Research in Paediatrics*, 76(1):10–16, 2011. (page 1)

[37] T. Nair, D. Precup, D. L. Arnold, and T. Arbel. Exploring uncertainty measures in deep networks for Multiple sclerosis lesion detection and segmentation. *Medical Image Analysis*, 2020. (page 3)

[38] V. Nair and G. E. Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the International Conference on Machine Learning*, pages 807–814, 2010. (page 30)

[39] R. M. Neal. Bayesian Learning via Stochastic Dynamics. In *Advances in Neural Information Processing Systems*, pages 475–482, 1993. (page 17)

[40] R. M. Neal and G. E. Hinton. A View of the EM Algorithm that Justifies Incremental, Sparse, and other Variants. In *Learning in Graphical Models*, pages 355–368. Springer, 1998. (page 4, 18)

[41] I. Osband. Risk versus Uncertainty in Deep Learning: Bayes, Bootstrap and the Dangers of Dropout. In *NIPS Workshop on Bayesian Deep Learning*, 2016. (page 64)

[42] B. T. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964. (page 10)

[43] C. Rasmussen. Gaussian Processes for Machine Learning. *Lecture Notes in Computer Science*, 2009. (page 15)

[44] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 1986. (page 9, 10)

[45] L. K. Saul, T. Jaakkola, and M. I. Jordan. Mean Field Theory for Sigmoid Belief Networks. *Journal of Artificial Intelligence Research*, 1996. (page 18)

[46] A. Schmeling, P. Manuel, J. Luis, and M. Irene. Forensic Age Estimation in Unaccompanied Minors and Young Living Adults. In *Forensic Medicine - From Old Problems to New Challenges*. InTech, 2011. (page 2)

[47] T. R. Shaham, T. Dekel, and T. Michaeli. Singan: Learning a generative model from a single natural image. In *The IEEE International Conference on Computer Vision (ICCV)*, pages 4569–4579, 2019. (page 2)

[48] K. Shridhar, F. Laumann, and M. Liwicki. A Comprehensive guide to Bayesian Convolutional Neural Network with Variational Inference. *Computing Research Repository*, 2019. (page 3, 17, 64)

[49] C. Spampinato, S. Palazzo, D. Giordano, M. Aldinucci, and R. Leonardi. Deep Learning for Automated Skeletal Bone Age Assessment in X-ray Images. *Medical Image Analysis*, 36:41–51, 2017. (page 2)

[50] N. Srivastava. Improving Neural Networks with Dropout. Master's thesis, 2013. (page 12)

[51] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15:1929–1958, 2014. (page 12, 20, 23)

[52] D. Štern, C. Payer, N. Giuliani, and M. Urschler. Automatic Age Estimation and Majority Age Classification from Multi-Factorial MRI Data. *IEEE Journal of Biomedical and Health Informatics*, 2018. (page 1, 3, 4, 27, 58)

[53] D. Štern, C. Payer, and M. Urschler. Automated age estimation from MRI volumes of the hand. *Medical Image Analysis*, 58, 2019. (page 2)

[54] D. Štern and M. Urschler. From Individual Hand Bone Age Estimates to Fully Automated Age Estimation Via Learning-based Information Fusion. In *IEEE International Symposium on Biomedical Imaging*, pages 150–154, 2016. (page 2)

[55] J. M. Tanner, R. H. Whitehouse, N. Cameron, W. A. Marshall, M. J. R. Healy, and H. Goldstein. *Assessment of Skeletal Maturity and Prediction of Adult Height (TW2 Method)*. Saunders London, 1983. (page 2)

[56] Y. Terada, S. Kono, D. Tamada, T. Uchiumi, K. Kose, R. Miyagi, E. Yamabe, and H. Yoshioka. Skeletal Age Assessment in Children Using an Open Compact MRI System. *Magnetic Resonance in Medicine*, 69(6):1697–1702, 2013. (page 2)

[57] H. H. Thodberg, S. Kreiborg, A. Juul, and K. D. Pedersen. The BoneXpert Method for Automated Determination of Skeletal Maturity. *IEEE Transactions on Medical Imaging*, 2009. (page 2)

[58] Y. Wen, P. Vicol, J. Ba, D. Tran, and R. Grosse. Flipout: Efficient Pseudo-Independent Weight Perturbations on Mini-Batches. In *International Conference on Learning Representations*, 2018. (page 66)

[59] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Generalized Belief Propagation. In *Neural Information Processing Systems*, 2000. (page 18)

[60] Y.-T. Zhou and R. Chellappa. Computation of Optical Flow Using a Neural Network. In *IEEE International Conference on Neural Networks*, volume 1998, pages 71–78, 1988. (page 11)