BRNO UNIVERSITY OF TECHNOLOGY
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

FACULTY OF INFORMATION TECHNOLOGY
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

DEPARTMENT OF COMPUTER GRAPHICS AND MULTIMEDIA
ÚSTAV POČÍTAČOVÉ GRAFIKY A MULTIMÉDIÍ

DEEP LEARNING MODEL UNCERTAINTY IN MEDICAL IMAGE ANALYSIS

NEJISTOTA MODELŮ HLUBOKÉHO UČENÍ PŘI ANALÝZE LÉKAŘSKÝCH OBRAZOVÝCH DAT

MASTER'S THESIS
DIPLOMOVÁ PRÁCE

AUTHOR AUTOR PRÁCE	Bc. DUŠAN DREVICKÝ
SUPERVISOR VEDOUCÍ PRÁCE	Ing. OLDŘICH KODYM

BRNO 2019

Brno University of Technology
Faculty of Information Technology

Department of Computer Graphics and Multimedia (DCGM) Academic year 2018/2019

Master's Thesis Specification

22094

Student: **Drevický Dušan, Bc.**

Programme: Information Technology Field of study: Computer Graphics and Multimedia

Title: **Deep Learning Model Uncertainty in Medical Image Analysis**

Category: Image Processing

Assignment:

1. Study the basics of convolutional neural networks and their applications to medical imaging.
2. Acquaint yourself with current methods of neural network uncertainty estimation.
3. Choose or design a method of anatomical landmark detection in medical data, including suitable uncertainty representation.
4. Identify a suitable dataset for the experiments.
5. Implement the proposed method and conduct experiments with the dataset.
6. Compare the achieved results and discuss possible future development.
7. Create a brief poster or video presenting the work, its goals and results.

Recommended literature:

- Olaf Ronneberger, Philipp Fischer, Thomas Brox: U-Net: Convolutional Networks for Biomedical Image Segmentation

Requirements for the semestral defence:

- Points 1 to 3.

Detailed formal requirements can be found at http://www.fit.vutbr.cz/info/szz/

Supervisor: **Kodym Oldřich, Ing.**
Head of Department: Černocký Jan, doc. Dr. Ing.
Beginning of work: November 1, 2018
Submission deadline: May 22, 2019
Approval date: May 9, 2019

Abstract

This thesis deals with quantifying uncertainty in the predictions of deep learning models. While they achieve state of the art results in many areas of computer vision, their outputs are usually deterministic and provide by themselves little information about how certain the model is about its prediction. This is important especially in the domain of medical image analysis where mistakes are costly and the ability to filter uncertain predictions would allow a supervising physician to review the relevant cases.

This work applies several different uncertainty measures developed in recent research to deep learning models trained on a cephalometric landmark localization task. They are then evaluated and compared in a set of experiments which aim to determine whether each of the uncertainty measures provides us with useful information about the model's confidence in its predictions.

Abstrakt

Táto práca sa zaoberá určením neistoty v predikciách modelov hlbokého učenia. Aj keď sa týmto modelom darí dosahovať vynikajúce výsledky v mnohých oblastiach počítačového videnia, ich výstupy sú väčšinou deterministické a neposkytujú mnoho informácií o tom, ako si je model istý svojou predpoveďou. To je obzvlášť dôležité pri analýze lekárskych obrazových dát, kde môžu mať omyly vysokú cenu a schopnosť detekovať neisté predikcie by umožnila dohliadajúcemu lekárovi spracovať relevantné prípady manuálne.

V tejto práci aplikujem niekoľko rôznych metrík vyvinutých v nedávnom výskume pre určenie neistoty na modely hlbokého učenia natrénované pre lokalizáciu cefalometrických landmarkov. Následne ich vyhodnotím a porovnávam v sade experimentov, ktorých úlohou je určiť, nakoľko jednotlivé metriky poskytujú užitočnú informáciu o tom, ako si je model istý svojou predpoveďou.

Keywords

deep learning, uncertainty, medical image analysis, landmark localization, cephalometry

Klíčová slova

hlboké učenie, neistota, analýza lekárskych obrazových dát, lokalizácia landmarkov, cefalometria

Reference

DREVICKÝ, Dušan. *Deep Learning Model Uncertainty in Medical Image Analysis*. Brno, 2019. Master's thesis. Brno University of Technology, Faculty of Information Technology. Supervisor Ing. Oldřich Kodym

Rozšířený abstrakt

Úvod

Táto práca sa zaoberá určením neistoty v predikciách modelov hlbokého učenia. Aj keď sa týmto modelom darí dosahovať vynikajúce výsledky v mnohých oblastiach počítačového videnia, ich výstupy sú väčšinou deterministické a neposkytujú mnoho informácií o tom, ako si je model istý svojou predpoveďou. To je obzvlášť dôležité pri analýze medicínskych obrazových dát, kde môžu mať omyly vysokú cenu a schopnosť detekovať neisté predikcie by umožnila dohliadajúcemu lekárovi spracovať relevantné prípady manuálne.

V tejto práci aplikujem niekoľko rôznych metrík vyvinutých v nedávnom výskume pre určenie neistoty, na modely hlbokého učenia natrénované pre lokalizáciu cefalometrických landmarkov. Následne ich vyhodnotím a porovnávam v sade experimentov, ktorých úlohou je určiť, nakoľko jednotlivé metriky poskytujú užitočnú informáciu o tom, ako si je model istý svojou predpoveďou.

Popis riešenia

Pre porovnanie analyzovaných metód som si vybral dataset cefalogramov [45], ktorý obsahuje naanotované pozície devätnástich anatomických landmarkov. Navrhnuté modely pre vstupný obrázok predikujú devätnásť heat máp, každá odpovedá jednému landmarku. Heat mapa je následne konvoluovaná s gaussovským filtrom a ako finálna pozícia landmarku je vybraná lokácia maxima vo výslednej aktivačnej mape.

V práci porovnávam tri metriky pre odhad neistoty modelov hlbokého učenia. Dve pochádzajú z nedávneho výskumu v tejto oblasti a sú aplikovateľné na rôzne typy úloh (klasifikácia, detekcia objektov a ďalšie). Tretiu metriku som navrhol sám, pričom je aplikovateľná primárne pre regresiu landmarkov.

Pre riešenie úlohy som navrhol architektúru konvolučnej neurónovej siete (CNN) vychádzajúcu zo siete U-Net [36], ktorá je populárna pri spracovaní medicínskych dát. Na jej základe som natrénoval tri modely zvané Baseline, Ensemble a MC-Dropout (každý využívajúci jednu z metrík), ktoré poskytujú predikciu pozície landmarkov spolu s odhadom jej neistoty.

Baseline je CNN model bez dropout vrstiev, ktorý využíva pre odhad neistoty maximum aktivácie heat mapy predikovanej pre každý z landmarkov. Pri návrhu tejto metriky som očakával, že pre landmarky, u ktorých si model predikciou nie je istý, bude táto hodnota relatívne nižšia v porovnaní so správne predikovanými landmarkami.

Ensemble je ensemble model zložený z 15 Baseline modelov a vychádza z myšlienok prezentovaných v práci Lakshminarayanan et al. [5]. Pre odhad neistoty využíva rozptyl predikcií jednotlivých členov ensemblu.

Model MC-Dropout aplikuje prístup, ktorého primárnym autorom je Gal [9], a ktorý reformuluje konvolučné siete využívajúce dropout vrstvy ako Bayesovské modely. Pre odhad neistoty využíva rovnako rozptyl predikcií, ktorý sa ale v tomto prípade počíta z 15 vzorkov vygenerovaných z modelu Monte Carlo vzorkovaním.

Výkon všetkých troch implementovnaých modelov na lokalizačnej úlohe je porovnateľný s najlepším riešením [45] v súťaží z ktorej pochádza použitý dataset. Natrénované modely zaostávajú v úspechu detekcie s chybovou toleranciou 2 a 2.5 mm, čo je spôsobené tým, že boli trénovací dataset obsahoval obrázky podvzorkované na veľkosť 128x128. To je z hľadiska cieľov práce akceptovateľné, keďže v nej ide primárne o porovnanie metrík neistoty a nie o dosiahnutie čo najlepšieho výsledku z hľadiska lokalizácie.

Efektivitu jednotlivých metrík neistoty som porovnal pomocou sady experimentov, ktorých cieľom bolo nájsť existenciu vzťahu medzi správnosťou predikcie a výškou danej metriky. Pre každý predikovaný landmark produkujú natrénované modely jednak jeho pozíciu a zároveň hodnotu neistoty. Pre testovacie obrázky je predikovaná pozícia landmarku porovnaná s jeho anotovanou pozíciou, pričom výsledkom je radiálna detekčná chyba.

Prvým experimentom bola korelačná analýza medzi veľkosťou radiálnej detekčnej chyby pre daný landmark a odpovedajúcou hodnotou metriky neistoty. Maximum aktivácie heat mapy zaznamenalo slabý výsledok s Pearsonovým korelačným koeficientom $\rho = -0.13$ (záporné ρ pre túto metriku indikuje užitočnosť, keďže predpokladáme, že neistota modelu klesá so stúpajúcou hodnotou aktivácie). Rozptyl predikcie modelu Ensemble dosiahol $\rho = 0.31$ a a rozptyl predikcie modelu MC-Dropout $\rho = 0.22$, čo naznačuje pre tento experiment vyššiu efektivitu týchto metrík neurčitosti.

Vizuálna analýza predpovedí modelu naznačila, že testovacie dáta sú príliš podobné trénovacím a modely na nich dosahujú tak vysokú úspešnosť, že metriky neistoty nemajú veľkú výpovednú hodnotu. V reálnom nasadení je však možné očakávať, že modely budú pracovať aj s dátami, ktoré sú značne odlišné od tých, ktoré videli počas tréningu. Model by mal byť schopný upozorniť na dáta, ktoré sú preň natoľko neznáme, že si svojou predpoveďou nie je istý. Vzhľadom na to, že som nemal k dispozícii naanotovaný dataset s odlišnou distribúciou, všetky tri modely som podtrénoval (tréning bol ukončený pred dosiahnutím konvergencie) a experiment zopakoval.

Rozptyl predikcie modelu MC-Dropout dosiahol v tomto prípade v korelačnej analýze s radiálnou detekčnou chybou $\rho = 0.86$ a rozptyl predikcie modelu Ensemble $\rho = 0.85$, čo je podstatné zlepšenie oproti plne natrénovaným modelom. Maximum aktivácie heat mapy sa podobne zlepšilo a dosiahlo $\rho = -0.35$. Je teda možné usúdiť, že užitočnosť všetkých troch metrík rastie spolu so vzdialenosťou evaluovaných dát od tréningovej distribúcie dát.

V ďalšom experimente som aplikoval na celý testovací dataset elastickú deformáciu. Vytvoril som tak jeho 40 kópií, pričom na každú z nich bola aplikovaná deformácia s rôznou magnitúdou. Cieľom bolo overiť, či existuje korelácia medzi hodnotou jednotlivých metrík neistoty pri predikcii a rastúcou mierou deformácie dát. Hypotézou bolo, že neistota modelov bude rásť spolu so silou deformácie aplikovanej na dataset, čo opäť overí ich schopnosť detekovať dáta vzdialené od tréningovej distribúcie. V experimente bola vykonaná korelačná analýza medzi silou deformácie aplikovanej na dataset a priemernou hodnotou metriky neistoty naprieč predikovanými landmarkami. Pre všetky metriky bola spomenutá schopnosť jednoznačne potvrdená. Najlepší výsledok dosiahlo maximum aktivácie heat mapy Baseline modelu s $\rho = -0.95$, ďalej MC-Dropout s $\rho = 0.85$ a Ensemble s $\rho = 0.81$.

Pri tvorbe cefalogramu je v ideálnom prípade pacientova hlava perfektne zarovnaná so sagitálnou rovinou a nedochádza k žiadnej rotácii v laterálnom smere. To však nemusí v reálnom nasadení modelu vždy platiť (pacient môže hlavou pri snímaní pohnúť), čo môže cefalogram znehodnotiť pre potreby predikcie landmarkov. Cieľom posledného experimentu bolo overiť schopnosť metrík neistoty detekovať cefalogramy, v ktorých je hlava pacienta laterálne natočená. Keďže dataset laterálne natočených cefalogramov nie je voľne k dispozícii, vytvoril som ho použitím CT snímku lebky. CT objem bol najprv laterálne zrotovaný v rozmedzí od -45 do 45 stupňov v axiálnej rovine. Výsledný objem bol následne premietnutý na sagitálnu rovinu sčítaním hodnôt intenzít prekrývajúcich sa voxelov. Následne som analyzoval koreláciu medzi veľkosťou rotácie a hodnotou metrík neistoty. Všetky tri modely a ich metriky neistoty sú schopné laterálnu rotáciu detekovať. Odpovedajúce korelačné koeficienty sú $\rho = -0.91$ pre Baseline, $\rho = 0.95$ pre Ensemble a $\rho = 0.88$ pre MC-Dropout.

Zhodnotenie výsledkov

Všetky tri analyzované metriky neistoty sa ukázali byť užitočné pri detekcii dát pochádzajúcich z distribúcie vzdialenej od tej tréningovej. Metriky založené na rozptyle predikcií mali konzistentne lepšie výsledky naprieš vykonanými experimentami ako maximum aktivácie heat mapy. Oba rozptyly predikcií vykazovali v experimentoch podobné chovanie, pričom metrika modelu Ensemble mala mierne lepšie výsledky. To je možné vysvetliť väčším množstvom parametrov, ktoré sú dostupné ensemblu 15 modelov oproti jednému CNN modelu založenom na metóde MC dropout. Experiment s podtrénovanými modelmi zároveň ukázal, že užitočnosť všetkých troch metrík stúpa, ak sú evaluované dáta pre model neznáme. To je dôležitý výsledok, pretože v reálnom nasadení v medicínskych systémoch, ktoré pracujú s dátami z rôznych prístrojov, je takáto situácia najpravdepodobnejšia a model by mal byť schopný robustnej reakcie na širokú škálu vstupov.

Prezentovaný výskum by mohol pokračovať rôznymi smermi. Dataset anotovaných cefalogramov pochádzajúci z iného prístroja by bol užitočný pre potvrdenie vykonaných experimentov. Modely by tiež mohli byť natrénované pre lokalizáciu úplne odlišnej sady landmarkov. Ďalej by mohli byť metriky založené na rozptyle predikcií evaluované na odlišnej úlohe akou je napríklad klasifikácia alebo segmentácia.

Deep Learning Model Uncertainty in Medical Image Analysis

Declaration

I hereby declare that this master's thesis was prepared as an original author's work under the supervision of Oldřich Kodym. All the relevant information sources, which were used during preparation of this thesis, are properly cited and included in the list of references.

. . . .
Dušan Drevický
May 20, 2019

Acknowledgements

I would like to thank my supervisor Oldřich Kodym for the time he dedicated to me, his advice and guidance. His suggestions and feedback were always useful and kept me moving in the right direction. I would also like to thank Michal Španěl for providing valuable comments that improved the readability of the text.

Contents

Chapter 1

Introduction

The relatively recent major work of Krizhevsky et al. [18] in 2012 which achieved state-of-the-art results on the ImageNet [38] classification task, has started a wave of successful deep learning applications in various scientific areas [37] that now also extends to the medical field [25].

The shortcoming of deep learning models is that they are usually treated as deterministic functions and provide only point estimates of predictions and model parameters without any associated measure of uncertainty. This may lead to situations in which we cannot tell whether a model is making reasonable predictions or just randomly guessing [7]. This is a crucial disadvantage for medical diagnosis which places heavy emphasis on risk-management. The information about the reliability of model predictions is a central requirement for their incorporation into the health-care diagnostic systems [47]. Deep learning models should thus provide each prediction with an estimate of its uncertainty. This would allow the diagnostic system to distinguish between easy cases which can be handled automatically and difficult ones which may instead be referred to a supervising physician for review [25].

On the other hand, models based on probability and uncertainty have been extensively studied in the Bayesian machine learning community. They provide a probabilistic view that offers confidence bounds when performing decision making [7] but usually come with a prohibitive computational cost. To take advantage of the qualities of deep learning models and still have the option of assessing the uncertainty of their predictions, it has been suggested [9] to recast them as Bayesian models using the popular dropout [13] technique often used for regularization. However, if turned on at test time, it can also be viewed as a way to approximate Bayesian inference by averaging multiple model predictions.

Additionally, while deep model ensembles have long been known to increase performance in terms of predictive accuracy [5] this work also explores a recent non-Bayesian line of research [20] which approaches them as an alternative way for obtaining uncertainty estimates.

Chapter 2 of this thesis provides an overview of deep learning applications in medicine. Chapter 3 first introduces the concept of uncertainty in machine learning more formally and then describes the recent research which aims to augment deep learning models with the ability to estimate it. Chapter 4 designs the solution to a landmark localization task on a dataset of cephalometric images using the Bayesian and non-Bayesian approaches to uncertainty modelling as well as a third uncertainty measure proposed by the author. The implementation details are described in Chapter 5. Chapter 6 evaluates the performance of the trained models and their corresponding uncertainty measures in a set of experiments.

Chapter 2

Deep Learning in Medical Imaging

Convolutional neural networks (CNNs) have been used for image analysis for decades. In spite of their initial success they waned in popularity until the ImageNet competition in 2012 in which a CNN model trained by Krizhevsky et al. [18] achieved state-of-the-art results by a large margin. This was made possible by the efficient use of graphics processing units, data augmentation and novel components of CNN architectures such as rectified linear units or dropout regularization [44].

CNNs have been applied to medical image analysis as far back as 1993 when Lo et al. [40] used them for lung nodule detection. Similarly to the general computer vision field, they did not catch on mainly due to long training times and difficult training [12]. This began changing in 2015 when the number of papers documenting deep learning in medical image analysis began to increase rapidly. Since then, deep learning has become the dominant theme at conferences in this field [27].

2.1 Challenges for Application

Practitioners applying deep learning to medical image analysis frequently face challenges that commonly appear in the medical field. Fortunately, these can often be addressed by implementing existing approaches from the general field of computer vision (such as using data augmentation when there are not enough training examples).

2.1.1 Dataset Size

Deep learning is most effective when applied to a large dataset of images since this allows one to take full advantage of all the parameters of a deep model without overfitting the training data. Models competing in the ImageNet [38] challenge are trained on millions of annotated images. On the other hand, it is not uncommon to have less than a thousand images in a medical dataset. Various strategies have been proposed to prevent overfitting including (i) taking 2D or 3D image patches as input instead of full-sized images in order to reduce input size and the number of required model parameters, (ii) applying data augmentation to expand the dataset, (iii) using models pre-trained on a large amount of natural images as feature extractors with an added classifier layer on top or (iv) fine-tuning an entire model pre-trained on natural images [39].

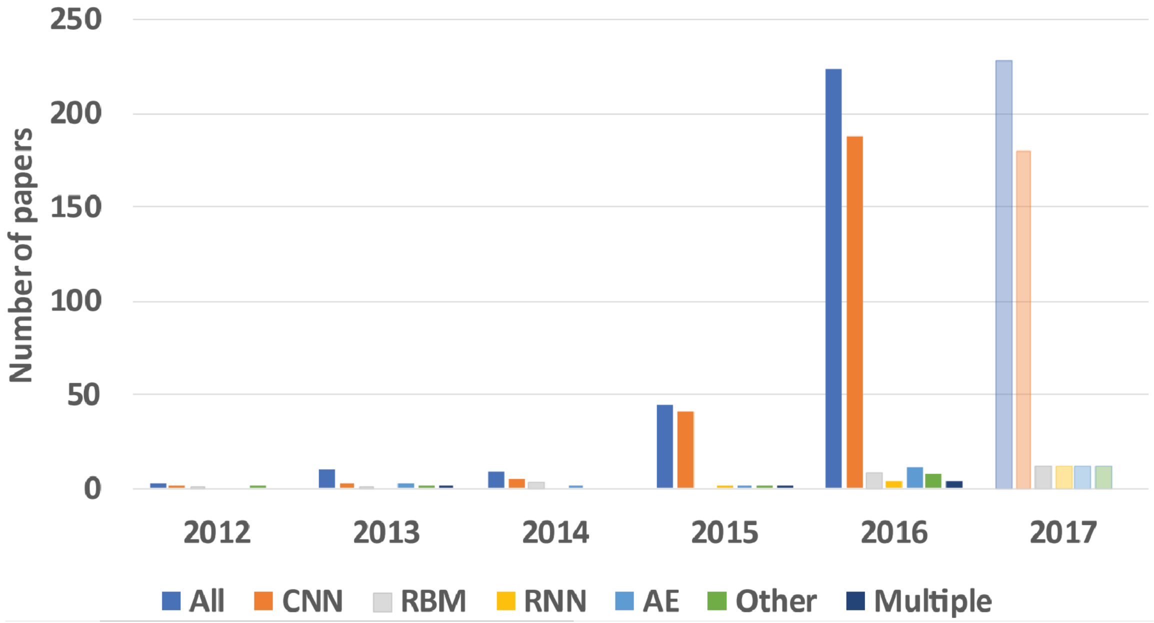


Figure 2.1: Deep learning papers published in the medical imaging field by year. Different color bars correspond to various type of deep learning models used in the papers. The number of papers for 2017 was extrapolated from the papers published in January [27].

2.1.2 Label Availability and Quality

Even in the cases where there are enough images to train a deep model, it may be difficult to acquire the ground truth annotations. Creating them requires expert knowledge and is often a time-consuming process [27]. A possible alternative is to take advantage of crowdsourcing to create non-expert labels [34].

Unfortunately, even expert annotations do not guarantee that the data will be immediately usable for training. Medical image annotations frequently suffer from label noise due to disagreement among the annotators. As an example, a popular lung nodule dataset LIDC-IDRI [40] was annotated by four radiologists. Upon reviewing the annotations, it was discovered that the number of patterns they did not unanimously classify as nodules was three times as large as the number of nodules they agreed on [27].

2.1.3 Within Class Heterogeneity

Classification or segmentation in medical imaging is often treated as a binary task (healthy or unhealthy, object or background). This is an oversimplification since each of these classes is usually heterogeneous. A healthy tissue may contain samples that are completely normal but also several categories of benign findings that may look quite different. This may lead to systems that are able to detect the normal subclasses very well but fail for the rarer ones. Converting the task to a multiclass classification problem is problematic due to the time constraints of the expert annotators [27].

2.1.4 Class Imbalance

Medical image datasets often contain an imbalanced ratio of images from the different classes. It is particularly common that there is a relative shortage of images from the ab-

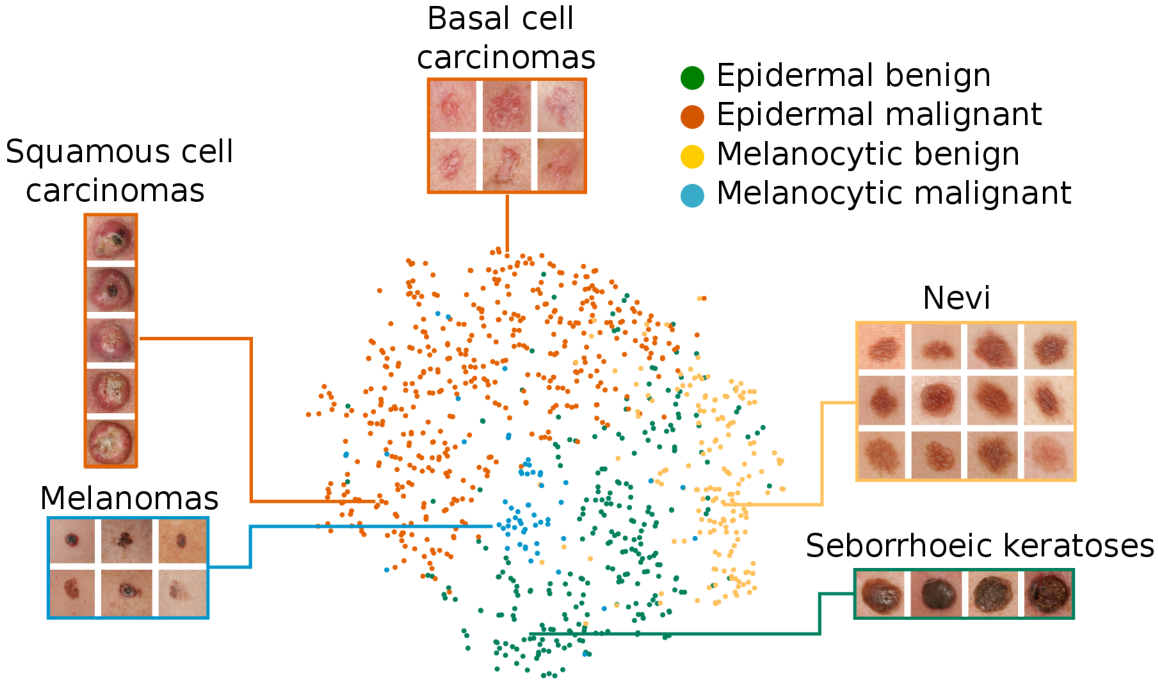


Figure 2.2: t-SNE visualization of skin cancer classes. The picture contains the final CNN layer representations of four different skin diseases. The colored point clusters show how the model groups the diseases [6].

normal class. For example, breast cancer screening has led to the acquisition of large number of mammograms but most of these are normal. Even if suspicious lesions are present, they are mostly benign. This problem is typically addressed using data augmentation to extend the dataset with extra samples from the under-represented class [27].

2.2 Applications Tasks

There is a broad range of different tasks that automatic analysis can help solve in the medical field. The most important of these along with some applications are described below.

2.2.1 Classification

In the medical setting, an image classification task is usually binary which means predicting whether the input image corresponds to a normal (no disease) or abnormal class, but it is also possible to distinguish between multiple classes. The dataset sizes are usually small for this task, which has necessitated the use of transfer learning from networks pre-trained on natural images [27].

Pre-training on natural images may still be beneficial even if there is an abundance of training data available. Esteva et al. [6] achieved performance comparable to dermatologists on a skin cancer classification task by using a Google Inception v3 CNN pre-trained on 1.28 million images from the ImageNet dataset and fine-tuned on 129,450 images of skin lesions (see Figure 2.2).

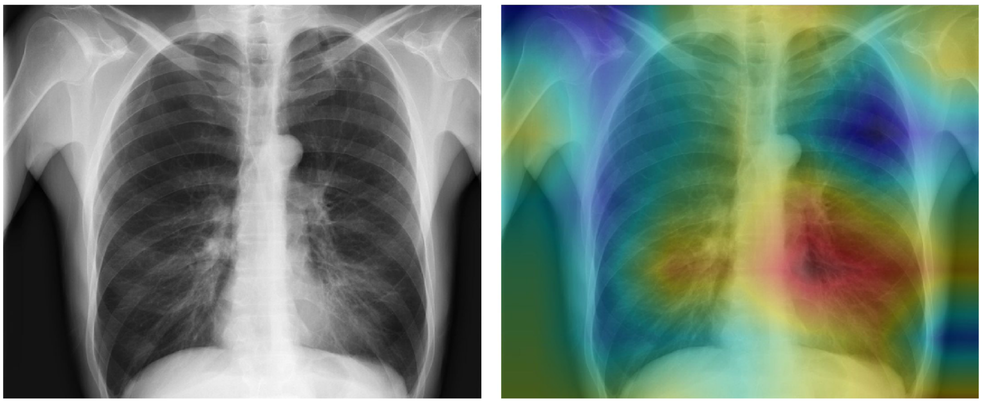


Figure 2.3: CheXNet lung disease detection results. The CheXNet model receives a chest X-Ray on input and outputs the probability of a pathology. In the example above, it correctly detects pneumonia and localizes areas (in red color) which it considers most indicative of the disease [35].

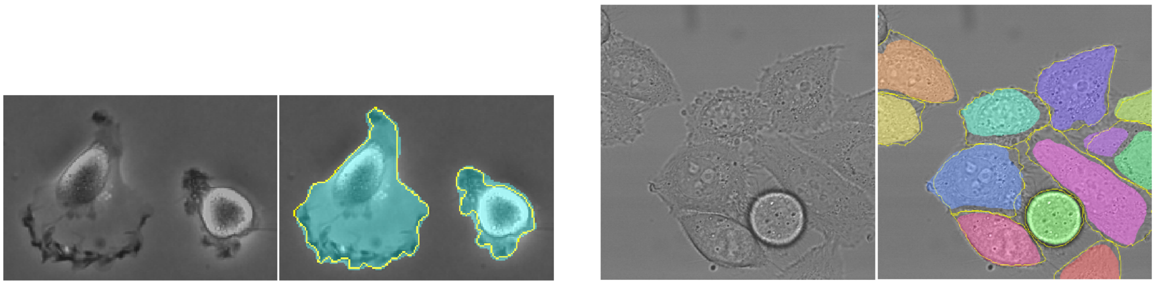


Figure 2.4: U-Net segmentation results. Both sets of pictures contain an input image and a corresponding U-Net output segmentation visualized using a colored mask. The ground truth is marked by a yellow boundary line [36].

2.2.2 Object Detection

Object detection is concerned with localizing a usually unknown number of objects in an image. In the medical setting, these are often pathologies or lesions and finding them is a very important part of the diagnosis process. When done manually by physicians, it is usually a labor-intensive task which has led to an extensive amount of research in this area even before the advent of deep learning. The work has focused on both improving detection accuracy and reducing the time spent by human experts on each case [12].

Architectures used for object detection are frequently similar to or based on architectures used for classification. A recent example is the CheXNet [35] network trained by Rajpurkar et al. which achieved state-of-the art performance comparable to radiologists when detecting abnormalities on chest X-Rays. The 121-layer CNN was trained from scratch on 112,120 chest X-Ray images annotated with up to 14 diseases (see Figure 2.3).

2.2.3 Segmentation

Segmentation is the most common task addressed by deep learning in medical imaging papers. It is usually defined as distinguishing between a set of foreground classes (one or several different organs or substructures) and a background class. This allows for the further analysis of volume and shape of the objects of interest in the data [27].

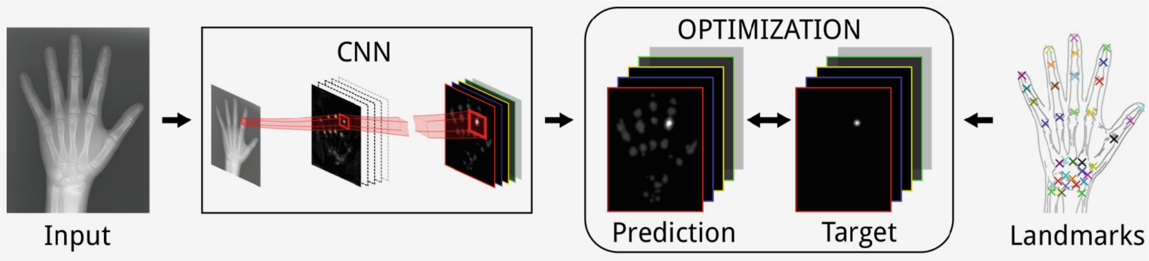


Figure 2.5: Landmark localization by heatmap regression. The CNN is trained to predict a single heatmap for each landmark. The heatmap contains a Gaussian activation at the predicted landmark location [32].

The first deep learning approaches to segmentation utilized neural networks which received patches obtained by sliding a window over the pixels in the input image. An example of this approach is work done by Ciresan et al. [3] which performed pixel-wise segmentation of electron microscopy imagery. One disadvantage of patch-based training approach is that patches overlap and this leads to redundant computation [27].

Current segmentation approaches use some variant of a fully-convolutional network [28] (FCN) which contains only convolutional layers without any fully-connected ones. The main advantage of FCNs is that they take the entire image as input and thus see the full context. The most popular of these in medical imaging is the U-Net [36] architecture proposed by Ronneberger et al. for microscopy image segmentation (see Figure 2.4).

2.2.4 Landmark Localization

Anatomical landmark localization is crucial in medical image analysis both as a frequent pre-processing step for segmentation task and as a part of the clinical process of diagnosis, planning and therapy [27]. The model can either be taught to regress the (x, y) landmark positions directly, but it is also possible to teach it to predict a landmark heatmap as proposed by Pfister et al. [33]. In the latter case, the network is trained on ground truth landmark heatmaps (usually a single plane per landmark) where the landmark position is marked by a Gaussian.

Landmark localization is able to successfully utilize fully-convolutional network architectures often used for segmentation. This usually amounts to changing the number of prediction channels in the final layer to the number of detected landmark heatmaps and modifying the loss function. Payer et al. [32] tested several FCN architectures on two datasets of hand scans achieving state-of-the-art results using their newly proposed architecture.

Chapter 3

Uncertainty in Deep Learning

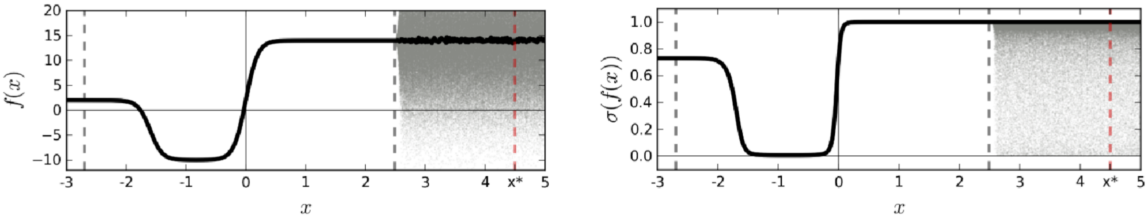
Standard deep learning regression and classification tools do not necessarily capture model uncertainty. In a classification task (in which a model learns to predict the probability of an example belonging to each of multiple classes), the outputs of the softmax function[1] are often interpreted as model confidence which may not be the case by default. A model may produce a high softmax output (suggesting high confidence) and still be be uncertain about the prediction [9]. An example of this behavior can be seen in Figure 3.1.

The usefulness of the raw model predictions for assessing uncertainty varies between different tasks. It also depends on the loss function and whether the model outputs were designed with such a goal in mind. For example, a landmark localization task can be stated in such a way, that a model learns to predict a continuous heatmap for each landmark being detected. The landmark position is then computed from the heatmap by convolving it with a Gaussian kernel of the same size that was used to produce the ground truth heatmaps (which contained a Gaussian at the landmark's location) and finding the position of the maximum activation. It is possible that the value of the maximum activation could in this case be a useful indicator of how certain the model is about its prediction[2].

Even in the latter case however, we might still benefit from using methods that have been designed specifically for assessing uncertainty instead of relying solely on model predictions.

[1] The softmax function is a generalization of the logistic function that is used to "squash" a K-dimensional real-valued vector $\mathbf{z}$ to a K-dimensional vector of real values $\sigma(\mathbf{z})$ where each entry is in the range $(0, 1)$ and all entries add up to 1 [1].

[2] The validity of this hypothesis will be examined as part of the experiments performed in this thesis.



(a) Arbitrary function $f(\mathbf{x})$ as a function of data $\mathbf{x}$ (softmax input)

(b) $\sigma(f(\mathbf{x}))$ as a function of data $\mathbf{x}$ (softmax output)

Figure 3.1: Softmax input and output sketch for a binary classification problem. Training data is contained between the dashed gray lines and function point estimate is given by the solid line. The shaded area indicates function uncertainty. Disregarding uncertainty, the point x^* located far from the training data is classified as class 1 with a confident softmax prediction close to 1.0. [9]

This chapter begins with a discussion of different types of uncertainty used in machine learning. Section 3.2 then describes a recently proposed Bayesian approach to extracting uncertainty estimates from deep learning models utilizing dropout [43]. Section 3.3 is concerned with the same task but uses recently suggested non-Bayesian approach of model ensembles to produce the desired uncertainty estimates. Both approaches are first described theoretically and this is followed by an overview of the practical results achieved by their respective authors. Section 3.4 describes the uncertainty measures that can be used in practice with deep learning models. Finally, Section 3.5 is concerned with different ways of evaluating the quality of uncertainty measures themselves.

3.1 Types of Uncertainty

Two different kinds of uncertainties are commonly considered in modelling.

Aleatoric uncertainty refers to the uncertainty which corresponds to the noise inherent in the process being observed [16]. For example, when observing a sample under a microscope, even if the sample does not change, the configuration of the microscope's camera and the interaction of photons with the sample lead to our inability to capture the same image twice. The uncertainty of this random process is *irreducible* beyond a certain point. Instead of a single value, we might predict a distribution which captures this intrinsic randomness [46].

We can further distinguish between *homoscedastic* aleatoric uncertainty which stays constant for different model inputs and *heteroscedastic* aleatoric uncertainty. For the latter, some inputs to the model may produce noisier outputs than others [16]. When explicitly modeling alaetoric uncertainty in a model, using heteroscedastic uncertainty provides us with more flexibility. We can for example consider an image regression task, in which we are predicting pixel intensity values using a probability distribution (and not just a point estimate as is common in deep learning). A homoscedastic approach would assign the same variance to each probability distribution of pixels in the image while a heteroscedastic approach would provide the model with an option to learn a different tailored variance for each pixel resulting in a greater expressive power of the model [46].

Epistemic uncertainty on the other hand, refers to uncertainty in model parameters or model structure. It captures our ignorance about which model (with which parameters) generated our data. This uncertainty is therefore *reducible* if more data were available to us, which would allow us to specify the model parameters more precisely. Consequently, epistemic uncertainty becomes more important as the amount of collected data decreases [16].

3.2 Bayesian Modelling

Bayesian modelling is concerned with epistemic uncertainty in model parameters. It essentially aims to average the predictions of all possible settings of the model parameters, weighing each setting by its posterior probability given the training data [43]. This is of course much more computationally expensive than a simple parameter point estimate commonly used in deep learning.

This section contains the recent theoretical research conducted primarily by Gal [9][7][8], which allows us to reformulate both fully-connected neural networks and convolutional neural networks with dropout [43] layers as Bayesian models which can model uncertainty. The end result of this theoretical work is that we can obtain practical measures of uncertainty from already existing deep learning architectures, with either no or only slight modifications (namely the addition of dropout layers) [9].

Section 3.2.1 describes the mathematical rationale for using dropout in a neural network as a method of Bayesian inference. Section 3.2.2 extends it to convolutional neural networks as well. Section 3.2.3 follows with an overview of experiments performed by Gal and Ghahramani [9] in order to demonstrate the usefulness of their method on real datasets.

A detailed review of Bayesian inference, variational inference and Bayesian Neural Networks (BNNs) is contained in appendix A.1 and a review of the dropout technique in appendix A.2.

3.2.1 Dropout as Variational Inference in BNNs

The Bayesian model parameters $\mathbf{w}$ of a Bayesian Neural Network (BNN) correspond to the weights in all L network layers. We can therefore define the model parameters as $\mathbf{w} = (\mathbf{W}_i)_{i=1}^{L}$ [8] and the random output of the BNN as $\mathbf{y} = \mathbf{f}^{\mathbf{w}}(\mathbf{x})$.

In order to relate the approximate inference in a BNN to dropout training, we define the approximate variational distribution $q_\theta(\mathbf{W_i})$ of the model weights for every layer i of the network. The weights $\mathbf{W_i}$ are drawn from the approximating distribution as follows:

$$\mathbf{W}_i = \mathbf{M}_i \cdot \text{diag}([\mathbf{z}_{i,j}]_{j=1}^{K_i}) \tag{3.1}$$

$$\mathbf{z}_{i,j} \sim \text{Bernoulli}(p_i) \text{ for } i = 1, ..., L,\ j = 1, ..., K_{i-1} \tag{3.2}$$

with the dimensions of each layer being $K_i \times K_{i-1}$ and the parameters of the variational distribution $q_\theta(\mathbf{w})$ defined as $\theta = \{\mathbf{M}_i, p_i | i \in 1, ..., L\}$. $\mathbf{z}_{i,j}$ are Bernoulli distributed random variables with probabilities p_i and $\mathbf{M}_i$ are weights of the network being optimized. The diag $(\cdot)$ operator maps a vector to a diagonal matrix whose elements on the diagonal are the elements of the matrix. We can look at sampling from the distribution $q_\theta(\mathbf{W_i})$ as applying dropout on the layer i in a network of L layers with weights $(\mathbf{M}_i)_{i=1}^{L}$. $\mathbf{z}_{i,j} = 0$ corresponds to dropping the unit j in layer $i-1$ as an input to layer i. Conversely, $\mathbf{z}_{i,j} = 1$ corresponds to keeping the unit active as an input to the next layer [8]. The dropout probabilities p_i can either be fixed to a certain value (as is commonly done when using dropout) or learned [42].

We have thus approximated the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ of the BNN weights $\mathbf{w}$ given some dataset $\{\mathbf{X}, \mathbf{Y}\}$ with a variational distribution $q_\theta(\mathbf{w})$ based on dropout. We can now use it to approximate the expectation of the BNN random output $\mathbf{y} = \mathbf{f}^{\mathbf{w}}(\mathbf{x})$ under the posterior distribution of the BNN weights. The dropout distribution is still difficult to marginalize but we can easily sample from it using a Monte Carlo (MC) approach [42]

$$\mathbb{E}_{p(\mathbf{w}|\mathbf{X},\mathbf{Y})}[\mathbf{f}^{\mathbf{w}}(\mathbf{x})] = \int p(\mathbf{w}|\mathbf{X}, \mathbf{Y})\mathbf{f}^{\mathbf{w}}(\mathbf{x})d\mathbf{w} \tag{3.3}$$

$$\approx \int q_\theta(\mathbf{w})\mathbf{f}^{\mathbf{w}}(\mathbf{x})d\mathbf{w} \tag{3.4}$$

$$\approx \frac{1}{T}\sum_{i=1}^{T} \mathbf{f}^{\mathbf{w_i}}(\mathbf{x}),\ \mathbf{w}_{i..T} \sim q_\theta(\mathbf{w}) \tag{3.5}$$

This approach to computing the prediction of a network containing dropout layers is called *MC dropout*. In practice, this amounts to computing the mean of T stochastic forward passes through the network [9]. Note that this is approach differs from the one traditionally used for dropout deep learning models called *weight averaging* [43] and does not lead to the same networks predictions. The predictive uncertainty over a prediction is obtained by computing the sample variance of the T stochastic forward passes. This approach works with existing dropout models without modification and the forward passes needed to compute MC dropout can be done concurrently, leading to a constant running time comparable with weight averaging [9].

Based on the preceding theoretical groundwork, implementing a Bayesian neural network with Bernoulli approximate variational inference amounts to adding dropout layers after each weight layer in a neural network. The dropout layers are applied during training, and test time predictions are computed using equation 3.5.

3.2.2 Bayesian Convolutional Neural Networks

BNNs model all of their layers with a probability distribution. All of these have to be integrated over when computing the posterior distribution. On the other hand, when dropout is used in a CNN, it is often applied only after the fully-connected layers. This strategy is equivalent to integrating only the fully-connected layers and taking point estimates of the parameter values of the convolutional filters. To produce a Bayesian CNN, dropout should be applied after both every fully-connected and convolutional layer [8].

To integrate over its filters, it is possible to reformulate the convolution as a linear operation (a matrix product). We also place a prior distribution over the filters in a manner similar to the one used for BNNs. The distribution then randomly zeroes the filters for different patches of the tensor used as convolution input. A Bayesian CNN can thus be implemented by applying a dropout layer after each convolutional layer [3][8].

It is noteworthy that dropout applied after convolutional layers may perform poorly at test time when combined with the weight averaging approach of evaluating predictions. According to the experiments performed by Gal et al. [8], applying MC dropout at test time instead performs significantly better.

3.2.3 Related Results

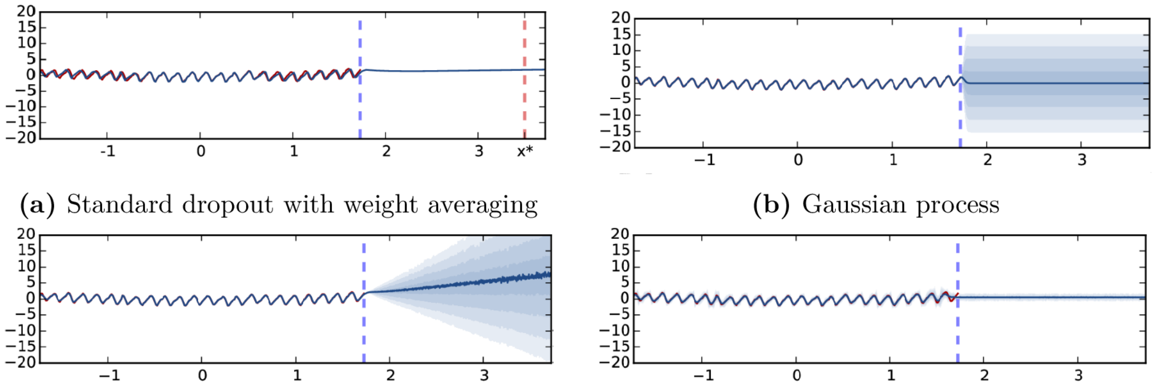
Gal and Ghahramani [9] followed their theoretical work with experiments on real datasets for both regression and classification tasks.

The performance of dropout networks for regression tasks is assessed on a dataset of atmospheric CO_2 concentrations containing about 200 data points. The authors train model networks with 4 or 5 hidden layers and 1024 hidden units, either ReLU or TanH non-linearities and dropout probabilities of 0.1 or 0.2. A Gaussian process with a squared exponential covariance function is evaluated as well. The results are depicted in Figure 3.2. None of the models were able to capture the periodicity of the data and do not predict a correct value for an input far from the training distribution. However, standard dropout with weight averaging still provides a confident (but insensible) prediction whereas the other models provide an insensible prediction along with an estimate of its uncertainty.

It is notable that the uncertainty of the MC dropout model with ReLUs keeps increasing further away with data while the uncertainty of the model with TanH non-linearities is bounded. The authors deduce that this behavior is related to the fact that ReLU non-linearity does not saturate while TanH does.

An additional experiment is performed to evaluate the ability of MC dropout to express model uncertainty in a classification task. The authors train a LeNet [22] convolutional

[3] Refer to the mentioned paper for a more in depth derivation of this result.



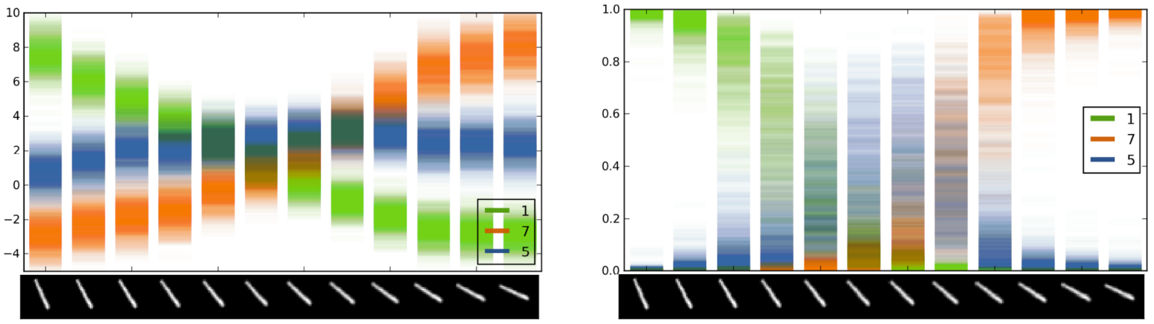
(a) Standard dropout with weight averaging

(b) Gaussian process

(c) MC dropout with ReLU non-linearities

(d) MC dropout with TanH non-linearities

Figure 3.2: Predictive mean and uncertainties on the CO$_2$ regression dataset for different models. The function observed based on training data is in red and left of the dashed blue line. The predictive mean of the models plus/minus two standard deviations (eight for 3.2d) is in blue. Each shade of blue corresponds to half a standard deviation. Point x^* marked with a red dashed line appears far from the training data. Dropout model with weight averaging confidently predicts an insensible value and provides no uncertainty information. The other models also predict insensible values but with the added information that they are uncertain about their prediction. Uncertainty was estimated using 1000 forward passes for MC dropout [9].



(a) Softmax input scatter

(b) Softmax output scatter

Figure 3.3: A scatter plot of 100 forward passes of the softmax layer input and output for dropout LeNet. The x-axis shows the rotated image of the digit 1 that was received by the network on input. The y-axis shows: **(a)** the computed class scores used as softmax input and **(b)** the corresponding softmax output. The model predicts the digits [1 1 1 1 1 5 5 7 7 7 7 7]. Only the predictions for the three digits with the highest class scores are shown [9].

network on the full MNIST [23] dataset and apply dropout before the last fully-connected layer with probability 0.5. The model is then evaluated on a continuously rotated image of the digit 1 as shown in Figure 3.3. It predicts the digits [1 1 1 1 1 5 5 7 7 7 7 7] for the 12 images.

The model always predicts the digit which has the largest class score (softmax input). When looking at the plot of the softmax input values in Figure 3.3b, if the uncertainty envelope of a class is far away from those of the other classes, then we can observe that it is classified with high certainty as measured by the variance of the predictions of all the forward passes in 4.1a. This happens for the three left-most and three right-most images. If on the other hand, the uncertainty envelope of the class scores intersects the envelopes of the other classes, then the uncertainty in the model predictions (softmax outputs) is much greater as determined by the same metric. Note that even in this case, the softmax output for a particular forward pass can be arbitrarily close to 1 and is a poor indicator of uncertainty. This is the case for one of the middle digits which is classified as a 5.

3.3 Ensemble Modelling

An ensemble of models is a set of models whose individual predictions are combined in some way (usually by a weighted or unweighted average) to produce the final ensemble prediction. It is a well established result that an ensemble of models improves predictive performance in comparison with the individual ensemble members, and a large amount of research has explored various methods of combining the individual models [5].

In general, there are two classes of ensemble models: *randomization*-based ensembles in which the individual members may be trained in parallel without any interaction, and *boosting*-based ensembles in which the individual members are trained sequentially. To decorrelate the predictions of the ensemble members it is also possible to use a *bagging* strategy in which each model is trained using a different random subset of the training set [20]. However, it has been observed [24] that for ensembles of deep models optimized for predictive accuracy, random initialization of member models and training on the entire dataset independently is superior to the bagging strategy.

3.3.1 Deep Ensemble Modelling

Using ensembles has recently been proposed by Lakshminarayanan et al. [20] as a viable alternative to Bayesian modelling (in practice implemented using MC dropout) for obtaining epistemic (model parameter) uncertainty estimates from deep learning models. Aside from its Bayesian interpretation which was explored in Section 3.2.1, dropout may also be interpreted as an ensemble model combination [43] where the final network predictions are the average of the predictions of an ensemble of NNs sharing the same parameters.

Lakshminarayanan et al. suggest that this view may be more plausible (and also valid for estimating predictive uncertainty) especially if the dropout probabilities are chosen arbitrarily, and are not learned along with the model weights, since any reasonable approximation to the Bayesian posterior distribution must be based on the observed training data [20].

The authors suggest training an ensemble of deep networks using a proper scoring rule (see section A.3 for a definition) as the training criterion. They also suggest using adversarial training [11] to improve the results. But since they consider it optional to the main method, and it is mostly orthogonal to uncertainty modelling, I will not discuss it here.

The expected advantage of using a proper scoring rule as a loss function is that it should lead to predictions that are well-calibrated and honest (i.e., not overly confident). The predictions themselves should then be a good indicator of model uncertainty. Many loss functions commonly used for training NNs are already proper scoring rules. Popular examples include the score function $S(p_\theta, (y, \mathbf{x})) = \log p_\theta(y|\mathbf{x})$ used for likelihood maximization or the cross entropy loss function used for multiclass classification [20].

The usual approach utilizing NNs for regression problems consists of predicting a single value say $\mu(\mathbf{x})$ and optimizing the model parameters to minimize the mean squared error (MSE) on the training set defined as $\sum_{n=1}^{N}(y_n - \mu(\mathbf{x}_n))^2$. The disadvantage of MSE is that it does not capture predictive uncertainty. Similar to [30], the authors instead use a network which outputs two values in the final layer (unique ones for each input): the mean $\mu(\mathbf{x})$ and the variance $\sigma^2(\mathbf{x}) > 0$[4]. The observed value is then treated as a sample[5] from a Gaussian[6] distribution with these parameters and used to minimize the negative log-likelihood (NLL) cost:

$$-\log p_\theta(y_n|\mathbf{x}_n) = \frac{\log \sigma_\theta^2(\mathbf{x})}{2} + \frac{(y - \mu_\theta(\mathbf{x}))^2}{2\sigma_\theta^2(\mathbf{x})} + \frac{\log(2\pi)}{2} \tag{3.6}$$

Each network in the ensemble (suggested ensemble size is $M = 5$) is randomly initialized and trained using the entire randomly shuffled training set. The ensemble is treated as a uniformly-weighted mixture model with predictions combined as $p(y|\mathbf{x}) = M^{-1}\sum_{m=1}^{m} p_{\theta m}(y|\mathbf{x}, \theta_m)$. For classification this corresponds to a simple average of individual predicted probabilities and for regression, the prediction is a mixture of Gaussians, which the authors further approximate by a single Gaussian with mean and variance equal to those of the mixture. For a mixture of M Gaussians $M^{-1}\sum \mathcal{N}(\mu_{\theta m}(\mathbf{x}), \sigma_{\theta m}^2(\mathbf{x}))$ the mean and variance are given by:

$$\mu_*(\mathbf{x}) = M^{-1}\sum_m \mu_{\theta m}(\mathbf{x}) \tag{3.7}$$

$$\sigma_*^2 = M^{-1}\sum_m(\sigma_{\theta m}^2(\mathbf{x}) + \mu_{\theta m}^2(\mathbf{x})) - \mu_*^2(\mathbf{x}) \tag{3.8}$$

3.3.2 Related Results

Lakshminarayanan et al. [20] evaluated the ensemble strategy for uncertainty modelling first on a regression task using a one-dimensional toy dataset. It consists of 20 samples drawn as $y = x^3 + \epsilon$ where $\epsilon \sim \mathcal{N}(0, 3^2)$ and a single layer architecture of 100 hidden units. A commonly used heuristic for obtaining approximate measure of uncertainty is to train an ensemble of NNs (minimizing MSE), obtain several point predictions and compute their empirical variance. The authors compare the performance of this approach with learning variance by minimizing NLL for a single such network and for an ensemble. The results are in Figure 3.4 and suggest that networks trained with NLL provide us with better predictive uncertainty and that using an ensemble improves performance, which is especially apparent as the predictions move further away from the observed training data.

[4] The positivity constraint on variance is enforced by passing the corresponding NN output through the *softplus* function $\log(1 + \exp(\cdot))$.

[5] Aside from using an ensemble of models to capture the epistemic uncertainty in model parameters, the described method now also captures the aleatoric uncertainty corresponding to the noise in the data.

[6] A more complex distribution can be used if the Gaussian is too restrictive.

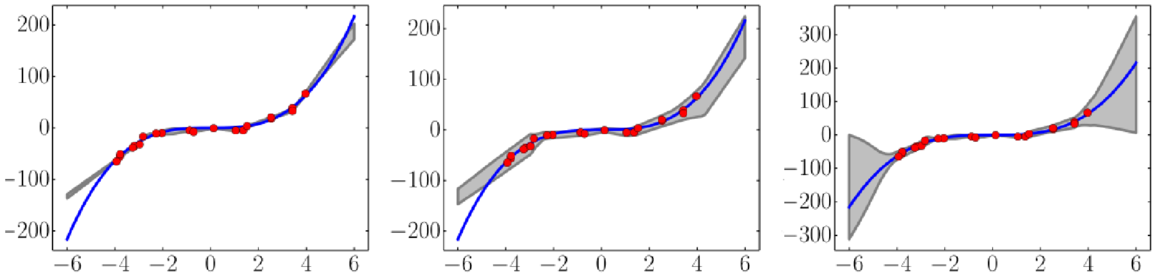


Figure 3.4: Results on the $y = x^3 + \epsilon$ toy dataset. The blue line is the ground truth, the red points are the observed training data and the gray lines are the predicted mean along with three standard deviations. The left plot shows the empirical variance of 5 NNs trained using MSE, the center plot the output of a a single NN trained using NLL and the right plot the performance of 5 NNs trained using NLL [20].

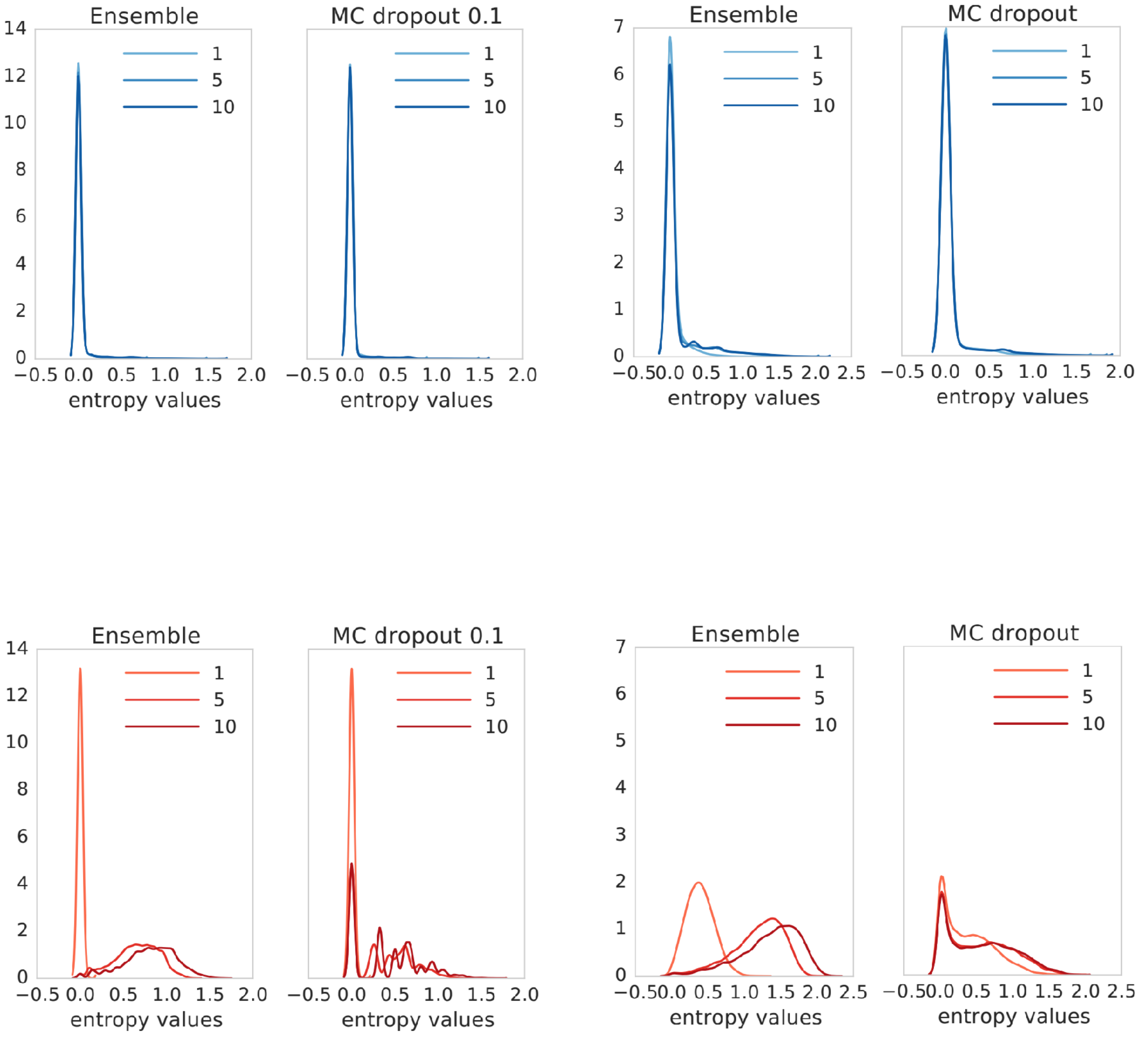
A second experiment on a classification task evaluates the model uncertainty on out-of-distribution examples from classes unseen during training. For a model to be reliable, it should not produce overconfident predictions when the test data is very different from the training data. It should be uncertain about its predictions instead. To evaluate whether models trained using NLL and composed into an ensemble have this property, Lakshminarayanan et al. train fully-connected networks with 3 hidden layers and 200 units per layer, using ReLU and batch-normalization. These are compared with an MC dropout model using the same architecture augmented with $p = 0.1$ dropout layers after each non-linearity. The models are trained on the standard MNIST train split. They are then evaluated on MNIST test split but also on the test split of the NotMNIST[7] dataset which contains letters instead of digits as in MNIST. A similar experiment is conducted by training on the SVHN dataset containing images of digits, and testing on the CIFAR-10 dataset containing images of ten different objects (cars, horses, etc.).

The quality of uncertainty estimates is evaluated using the entropy of the predictive distribution (see Figure 3.5). For known classes observed during training, the proposed method and MC dropout both have low entropy and are confident about the predictions as expected. For the unknown classes, the entropy of the deep ensembles is higher compared with MC dropout suggesting that the proposed method is superior for handling unseen test examples. In particular, MC dropout produces overly confident predictions for some of the unknown classes as indicated by the entropy mode centered around zero.

Building on the MNIST experiment, the authors evaluate whether the trained models are well-calibrated (see Section A.3), which means that their predictions can be trusted if they have high confidence and they are truly uncertain when the predictions have low confidence (confidence here refers to the value of the largest predicted probability by the model i.e., the probability of the class that the model predicts).

The models are again evaluated on both the known classes from MNIST and unknown classes from NotMNIST. A well-calibrated network should have low confidence for predictions on out-of-distribution NotMNIST examples. Test examples for which the model predictions are below the confidence threshold $0 \leq \tau \leq 1$ are filtered out, and the accuracy on the remaining test examples (for which the confidence is above the threshold τ) is plotted in Figure 3.6. We would expect a model to have higher accuracy for larger values of τ so the curve should be monotonically increasing.

[7] See http://yaroslavvb.blogspot.co.uk/2011/09/notmnist-dataset.html



(a) MNIST-NotMNIST

(b) SVHN-CIFAR

Figure 3.5: Model predictive entropy. Histogram of predictive entropy for test examples from known classes (in blue) and unknown classes (in red) for different values of M (ensemble size for the networks trained using NLL and number of samples for the MC dropout model). Both models produce confident prediction for the known classes. The ensemble of NLL networks is uncertain when predicting for unknown classes while MC dropout is erroneously overconfident for some of them [20].

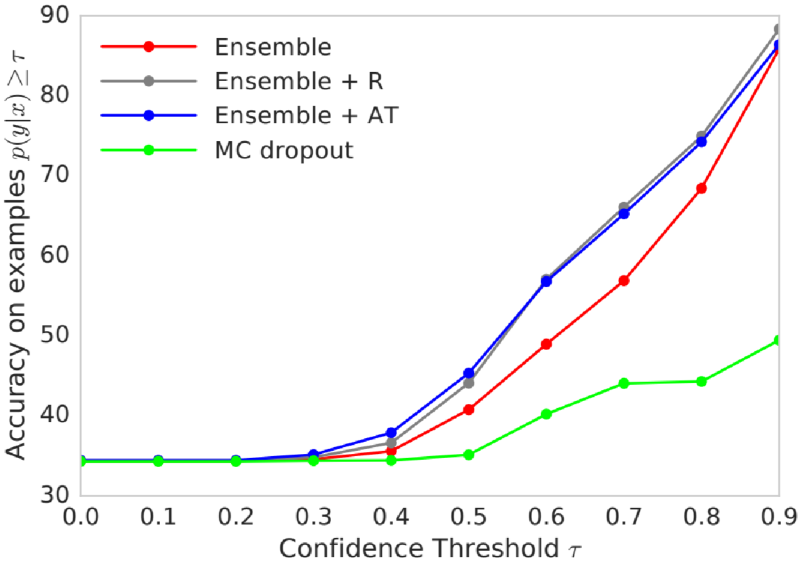


Figure 3.6: Accuracy vs Confidence curves. Models are trained on MNIST and evaluated both on the MNIST test set and on the unseen NotMNIST test set. Red, gray and blue lines correspond to an ensemble of networks trained using the proposed method (with additional data augmentation and adversarial training for gray and blue respectively) and green to MC dropout model. In comparison with the other models, MC dropout produces overconfident predictions as evidenced by low accuracy even for high values of τ [20].

3.4 Uncertainty Measures

In this section, I describe measures that can be used to evaluate deep learning model uncertainty.

3.4.1 Raw Model Predictions

Using the raw model predictions is the most straightforward way of extracting information about the model's uncertainty. For some tasks such as classification or segmentation the model outputs can be interpreted as probabilities, so it seems natural to relate them to model's uncertainty about its prediction. But as we have seen in Figure 3.1, a model may provide a very confident prediction with probably close to 1.0 even for an example which is far from the data distribution it has seen during training. Consequently, the numerical values of the predictions themselves can be misleading.

We also need to take into account that for most regression problems (e.g. predicting pixel intensity values) the model predictions cannot be interpreted as providing any uncertainty information at all, and other measures must be used instead.

3.4.2 Model Prediction Variance

This uncertainty measure requires that we either have a model which can make stochastic prediction samples (such as an MC dropout based model described in Section 3.2.1) or an ensemble of deterministic models with different parameters. In both cases, we essentially compute the predictions of multiple models (we can consider a single MC dropout sample to be a prediction from a single network from the space of all possible networks created by randomly dropping some of the units) for a single point of data. Since each model has different parameters (either due to different units being dropped or due to random initialization at the start of the training) we can assume that they will not make the same kind of mistakes. Consequently, if prediction variance is high, we may assume that each model is uncertain and just guessing, with each guess being dependent on its particular parameters. If on the other hand the variance is low and the models agree, we may conclude that they have a reason to be certain about their predictions.

3.5 Evaluating the Quality of Uncertainty Measures

There are several ways of assessing whether an uncertainty measure provides us with useful information. In this section, I describe three options for performing such an analysis. The test set filtering and out-of-distribution data detection methods were both used by the Deep Ensemble [20] authors when comparing the uncertainty estimates of their ensemble with an MC dropout model. I propose to also use the uncertainty-performance correlation which lends itself well to the landmark localization problem which I explore in the practical part of this thesis.

3.5.1 Correlation between Uncertainty and Performance Measures

This is a straightforward way of evaluating an uncertainty measure used for a regression task. We simply compute the correlation coefficient between the uncertainty and performance measures. Ideally, the correlation coefficient would have a high value (either positive or negative depending on the uncertainty measure) indicating that the model performance is tightly coupled with the uncertainty measure.

3.5.2 Test Set Filtering

This strategy evaluates the quality of an uncertainty measure by using the accuracy (or a different metric) of the model on the test set as the criterion. For each example in the test set, we compute the model prediction as well as the value of the analyzed uncertainty measure. We can then plot the achieved accuracy on the test set when taking into account only the examples for which the model uncertainty measure was below some threshold τ (the filtering is done for different values of τ). If the accuracy on the filtered test set increases as we decrease the uncertainty threshold, then the analyzed measure provides useful information about the uncertainty of the model predictions.

Section 3.3.2 describes an experiment evaluating whether a trained Deep Ensemble model is well-calibrated (see Figure 3.6). Note that the example on the figure differs from the explanation above in that the evaluated uncertainty measure is model confidence (which actually corresponds to model *certainty*) so the accuracy and the measure should increase together. Calibration has a definition related to probability (see Section A.3) but the same experimental procedure in which we filter the test set based on uncertainty measure thresholds generalizes well even to a regression problem.

Both this method and correlation answer a similar question but the advantage of test set filtering is that we can also determine what percentage of the data would be retained at specific uncertainty thresholds. This may be useful in practical applications. As an example, consider a system which automatically classifies medical images when the model uncertainty on the incoming data is below a threshold value, and asks for supervision only when it encounters a data point for which the model's uncertainty is above the threshold.

In such a case, the uncertainty threshold would have to be determined in advance (preferably on held-out test data) while taking into account two criteria.

Firstly, we want the system to perform well enough (which usually amounts to requiring some minimum performance on the accuracy measure) in the cases where it does not ask for supervision. This requires only images with relatively low uncertainty to be classified automatically. On the other hand, setting the uncertainty threshold excessively low would result in too many requests for supervision, which would negate the entire purpose of the system. Since test set filtering provides performance measure values and percentage of retained data at specific uncertainty thresholds, it allows us to balance both criteria much better than correlation.

3.5.3 Out-of-distribution Data Detection

Another desirable attribute of an uncertainty measure is its ability to detect situations when the model encounters data that are far from the distribution of the training data. Intuitively, we should be less trustful of a model's prediction when the evaluated data point differs from the data that the model has seen during training. The value of the uncertainty measure for the data point should thus be proportional to its distance from the training data distribution.

Section 3.3.2 describes the comparison between a Deep Ensemble and MC dropout model's ability to detect out-of-distribution data. Both were trained on the MNIST dataset and their uncertainty estimates were evaluated for the MNIST test set as well as on the NotMNIST test set which contains letters instead of digits. See Figure 3.5 for details.

Chapter 4

Experimental Task Design

I used a landmark localization problem to test and compare the two approaches to modelling uncertainty in deep learning described in Chapter 3. Both the Bayesian modelling approach utilizing dropout with MC sampling, and the deep ensemble approach are implemented using the same model architecture (with and without dropout layers respectively) which is also described in this chapter along with a brief characterization of the dataset used for the experiments.

4.1 Cephalometric Landmark Localization

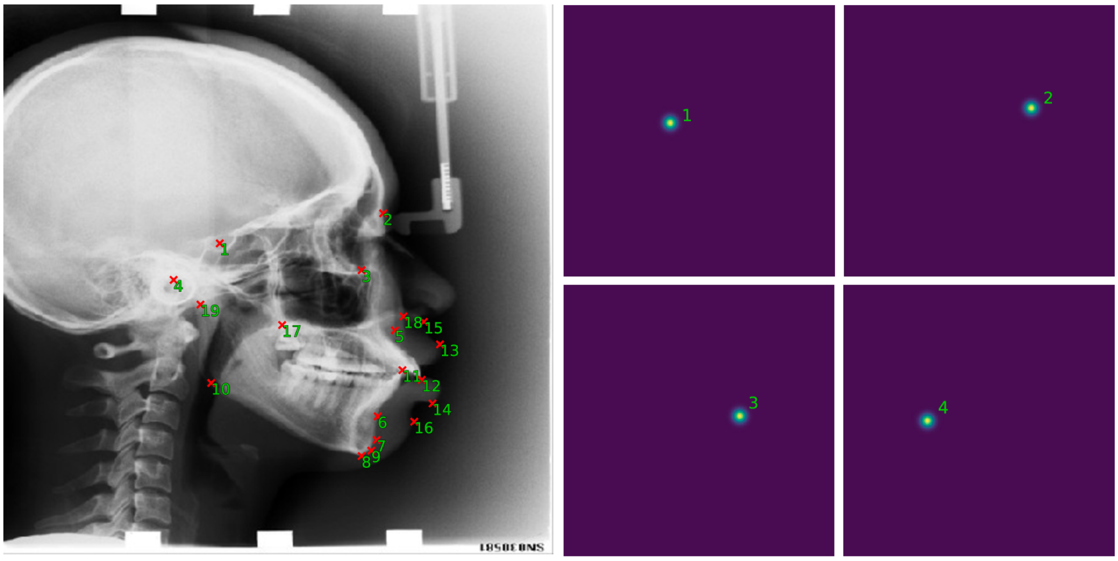
Cephalometric analysis provides clinicians with the interpretation of the bony, dental and soft tissue structures in patients' dental X-ray images. The images resulting from the analysis contain relationships between key points (landmarks) in the radiogram. They are then used for treatment planning, clinical diagnosis, classification of anatomical abnormalities and for surgery. This procedure is time-consuming and subjective if performed by experts. Automatic landmark localization helps to alleviate both of these problems [45]. An additional uncertainty estimate associated with the predicted landmark position could be very helpful for a physician using the landmarks for further clinical work.

4.2 Dataset

The dataset used for the landmark localization experiments was released as a part of the 2015 Grand Challenge in Dental X-ray Image Analysis [45]. It consists of 400 lateral cephalograms from 400 subjects. All cephalograms were acquired in the same format and from an identical scanning machine. The resolution of the images is 1935 x 2400 pixels with a pixel spacing of 0.1 mm. Two clinical orthodontists (a senior with fifteen years of experience and a junior one with six years of experience) provided ground truth manual annotations of 19 cephalometric landmark positions [26]. For simplicity and consistency, only the ground truth from the senior physician was used for accuracy evaluation.

The authors of the challenge split the dataset into three non-overlapping sets of images: **train** set contains 150 images (the only part of the data that models could see before evaluation), **test1** containing 150 images and **test2** containing 100 images.

A single example from the dataset consists of a cephalogram and positions of the 19 annotated landmarks. I chose to implement the approach to landmark localization suggested by Pfister et al. [33] in which the landmark positions are not regressed directly as a pair



(a) Input image with visualized landmarks **(b)** Heatmaps for first four landmarks

Figure 4.1: Cephalogram with ground truth labels. Each image in the dataset contains 19 annotated landmarks and a ground truth heatmap with the same dimensions as the image is made for each of them, by creating a Gaussian at the landmark's position.

of real coordinates but the model learns to regress a separate heatmap for each landmark instead. For each training example, the CNN receives a single-channel gray-scale image rescaled to $d \times d$ dimensions. The corresponding ground truth is a $19 \times d \times d$ volume of heatmaps. Each heatmap corresponds to a single landmark and contains a Gaussian with a fixed variance and amplitude centered on the landmark position as annotated by the physician. The output of the CNN is a $19 \times d \times d$ volume of predicted heatmaps. As a post-processing step, each heatmap is convolved with a Gaussian filter of the same variance as was used when creating the ground truth heatmap, and the maximum activation is chosen as the final predicted landmark position.

4.3 Model Architecture

The model design closely follows the U-Net [36] architecture with some modifications (see Figure 4.2). It consists of a down-sampling path followed by a symmetric up-sampling path. Down-sampling the input to a low resolution allows the network to learn global context which contains the information about the relative positions of the landmarks.

The channel dimension of the input image is first expanded using a double convolution which allows the network to model richer features. A max pooling layer then halves the resolution of the feature map in the width and height dimensions. Each down-sampling level then follows a similar pattern: the input feature map is passed through a double convolution, first of which increases the number of its channels by a factor of two, and is then passed through a max pooling layer.

The up-sampling path consists of applying a transposed convolution to a lower level feature map which halves its channel dimension. The stored feature map from the corresponding down-sampling level is concatenated to this result and passed through a double convolution. This pattern repeats for each level in the up-sampling path. The final convo-

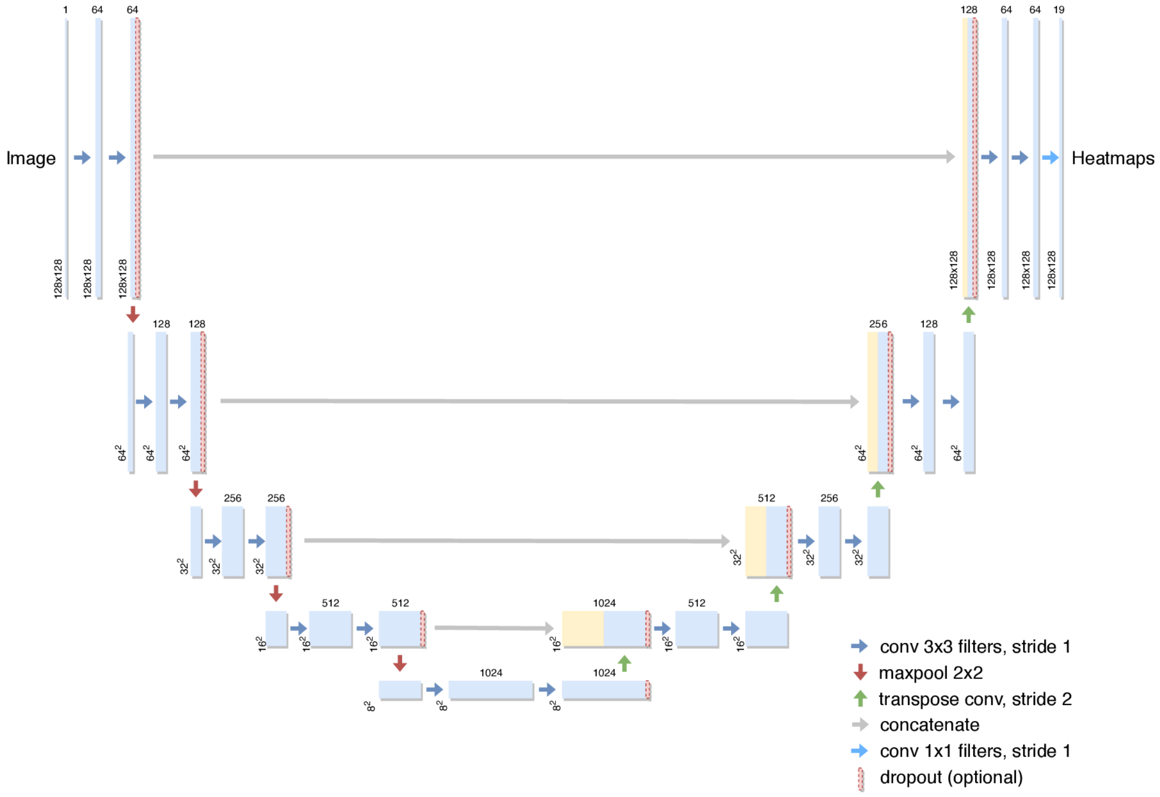


Figure 4.2: Proposed model architecture. Sizes of the feature maps (blue boxes) correspond to an input image with dimensions 128 × 128. Number of channels is at the top of the box and width/height dimensions at the bottom left of the box. Yellow boxes are the feature maps copied from the down-sampling part. Red boxes represents an optional dropout layer applied after the corresponding convolutional layer. Arrows represent different operations.

lutional layer at the top-most level uses 1×1 filters to produce the landmark heatmaps as the model predictions.

A batch-normalization layer [14] is applied after every convolutional layer's ReLU activation to speed up training. If the model uses dropout layers, then they are included just before max pooling in the down-sampling path and right after transposed convolution and concatenation operation in the up-sampling path.

4.4 Uncertainty Measures

Three different uncertainty measures described in Section 3.4 are evaluated and compared in this work. This section briefly describes how they can be applied to a model trained for landmark localization and proposes several experiments for the purpose of their comparison.

4.4.1 Proposed Uncertainty Measures

Firstly, it is determined whether the maximum activation in the predicted landmark heatmap convolved with a Gaussian kernel provides information about model uncertainty. I hypothesize that if a model is uncertain in its prediction of a landmark position, the activation will be lower than in the case when the model is certain. Secondly, a model with dropout layers is trained and I assess whether the variance of its predictions obtained by MC sampling provides usable uncertainty estimates. Finally, an ensemble of multiple models without dropout is trained and the variance of their predictions is evaluated in a similar way.

4.4.2 Proposed Experiments

The following experiments are performed and evaluated in this work:

1. **Correlation between Uncertainty and Performance Measures**
   This experiment determines whether a relationship exists between the radial error for the predicted landmark and the corresponding uncertainty measure value

2. **Test Set Filtering**
   This experiment determines whether the performance metrics improve as we remove data for which the model uncertainty is high.

3. **Elastically Distorted Out-of-distribution Data**
   Elastic distortions were applied to the test set to determine whether the uncertainty measures increase together with increasing distortion strength.

4. **Laterally Rotated Out-of-distribution Data**
   A dataset of laterally rotated cephalograms was created and the experiment evalues whether the uncertainty measures increase together with the increasing lateral rotation of the patient's head.

Additionally, experiments 1 and 2 were also performed using under-trained models whose training was stopped before convergence. This was useful since the fully-trained models turned out to be too accurate on the test data (which was similar to the training data) and allowed us to compare the performance of the uncertainty measures both for familiar and unfamiliar data (the unfamiliarity with data was simulated by reducing model performance since a different test dataset was not available).

Chapter 5

Implementation

This chapter contains the implementation details related to data pre-processing and the training procedure.

5.1 Training Procedure

The input to the CNN is a single channel gray-scale image. While the proposed architecture is a fully-convolutional network which means that it can take arbitrary size images as inputs, all models were trained exclusively on 128x128 size images which allowed for faster training times and easier experimentation. This led to some loss of performance in terms of accuracy compared with models trained on larger images but since the goal of this thesis is the exploration of uncertainty measures the compromise seemed acceptable.

Models along with the training process were implemented using the PyTorch [31] framework and training ran on a Tesla P100 GPU with 16GB of memory.

5.1.1 Training Data

The data split follows the one used in the original 2015 challenge [45] (see also Section 4.2) with the training set containing 150 images, so that the performance of the models could be compared with the ones participating in the challenge. It was further subdivided into a training and validation split using a ratio of 85:15.

5.1.2 Data Augmentation

Data augmentation was used to increase the size and variability of the relatively small training set (no data augmentation was applied to images used for testing). For an input image of dimensions $d \times d$ these consisted of:

- **Scale**: Sampled from the range [0.95, 1.05]
- **Horizontal flip**: Applied with probability equal to 0.5.
- **Rotation**: Sampled from the range [-5, 5] degrees.
- **Translation**: Both vertical and horizontal, sampled from the range $[-0.03d, 0.03d]$.

All augmentations were applied both to images and ground truth heatmaps. The augmentation range was had to be restricted because some of the training images contain

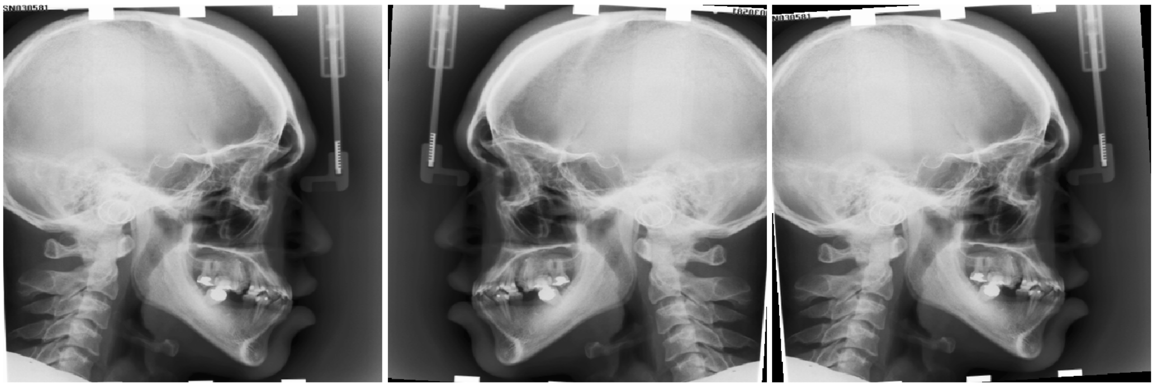


Figure 5.1: Data augmentations. A random horizontal flip and an affine transformation consisting of a random rotation, translation and scale was applied to each training image and its corresponding labels. Three different samples for the same training example are displayed above.

landmarks which are very close to the edge of the image. Too strong a transformation (translation for example) can make them disappear from both the image and the heatmap which would be undesirable.

5.1.3 Loss Function

The MSE loss was used to train the models for heatmap regression:

$$\text{MSE} = \frac{1}{m}\sum_{i=1}^{m}(y - \hat{y})^2 \tag{5.1}$$

where y are the ground truth heatmaps, $\hat{y}$ are the model predictions and m is batch size. Since the ground truth heatmaps contain non-zero values only in a small region around the landmark position where the Gaussian is active, the neural networks focused on predicting a zero-filled heatmap and ignored the Gaussian if its amplitude was set to 1. This led to very slow convergence of the training process. This is presumably because predicting the non-zero Gaussian at the landmark position does not contribute significantly towards lowering the MSE loss. To alleviate this, the Gaussian's amplitude was increased which consequently increased the loss function's gradient in the area around the landmark position and thus improved convergence. For training images of size 128 × 128, the best-performing Gaussian had an amplitude of 1000 and standard deviation of 5. Note that the settings of these parameteres should be cross-validated for when changing the training image size.

Some initial experiments using the negative log-likelihood loss function as suggested for deep ensembles in Section 3.3.1 were also performed. However, the models failed to converge to a meaningful heatmap prediction using this loss function so the MSE which provided consistent performance was used instead for all of the trained models.

5.1.4 Training Parameters

The Adam [17] optimizer with a batch size of 32 was used for training. The initial learning rate was set to 10^{-3} and weight decay of 10^{-4} was used to reduce overfitting. The model was evaluated on the validation set after each epoch and if performance in terms of validation loss did not improve for 10 consecutive epochs, the learning rate was decreased by a factor of 10. The training was stopped if the validation loss plateaued for 30 consecutive epochs. Only the model weights with best performance on the validation set were kept.

Chapter 6

Experiments and Results

This chapter contains the description of the trained models, evaluation of their performance on the landmark localization task along with a comparison with the state-of-the-art published model on the same dataset. The main part of this chapter documents different experiments analyzing the behavior of the proposed uncertainty measures in various scenarios.

6.1 Trained Models

Three models using the common architecture and training procedure described in Sections 4.3 and 5.1 respectively are evaluated in this chapter.

The **Baseline** model does not include any dropout layers and uses the maximum heatmap activation as its uncertainty measure. Additionally, 15 instances of the Baseline model were trained independently from scratch to form the **Ensemble** model which uses its members' prediction variance for uncertainty estimation.

The third model contains dropout layers at the end of each down-sampling level and at the beginning of each up-sampling level as shown in Figure 4.2. Although this contradicts the suggestion of Gal et al. [8] who originally proposed the inclusion of dropout after each convolution, that requirement is not necessary in practice and the placement of dropout layers ends up being a part of hyper-parameter search (see for example the work by Kendall et al. [15]). The probability of a unit being dropped is set to $p = 0.4$ uniformly for all of the dropout layers and was chosen based on performance results during cross-validation. **MC-Dropout** refers to a version of this model evaluated using 15 samples and the MC dropout scheme described in Section 3.2.1. It uses MC sample prediction variance to estimate uncertainty.

6.2 Landmark Localization Evaluation

Before assessing the quality of the uncertainty measures, the model performance was evaluated on the landmark localization task using the same metrics as were used in the 2015 Grand Challenge in Dental X-ray Image Analysis [45]. The trained models can thus be compared with the ones which participated in the competition.

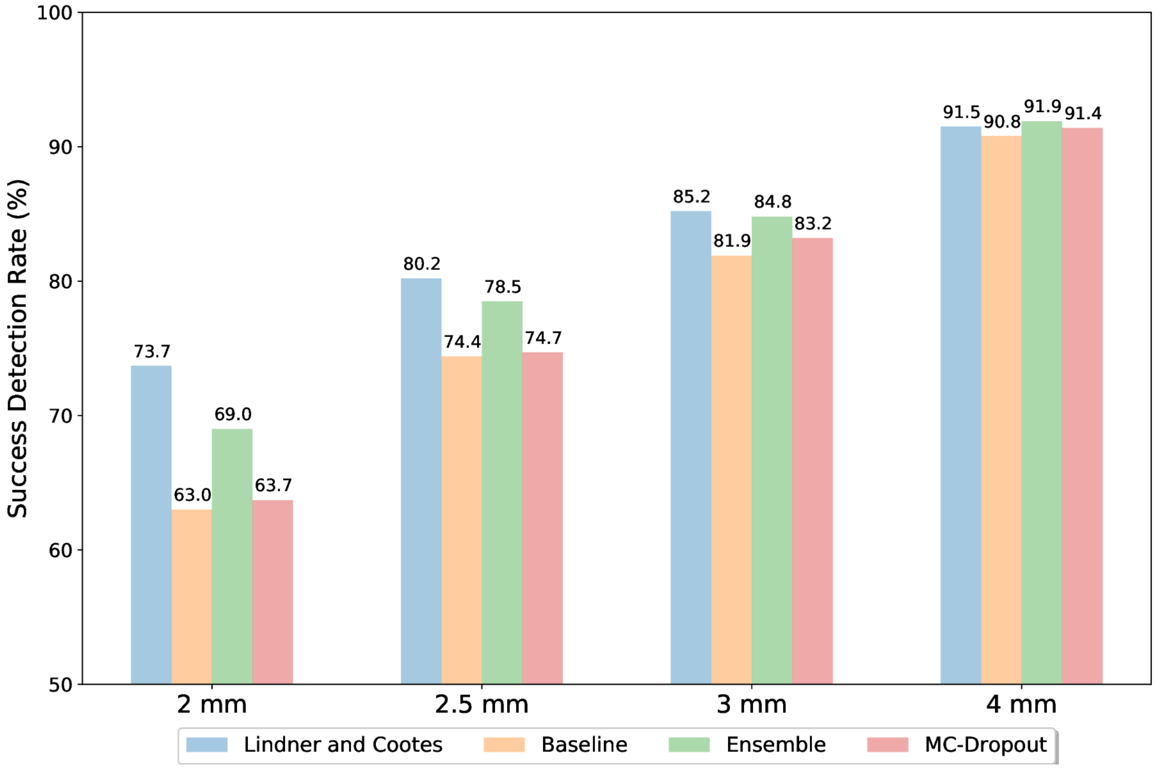


Figure 6.1: Success detection rates (SDRs). Comparison between the proposed models and the best-performing method from the 2015 competition by Lindner and Cootes [45]. All four have been evaluated on the test1 split of the dataset.

6.2.1 Performance Measures

The radial error R is simply the Euclidean distance defined as $R = \sqrt{\Delta x^2 - \Delta y^2}$ where Δx and Δy are the distances between the predicted and actual landmark position in the x and y direction respectively. The **mean radial error** (MRE) and the associated **standard deviation** (STD) are defined as

$$\text{MRE} = \frac{\sum_{i=1}^{N} R_i}{N} \tag{6.1}$$

$$\text{STD} = \sqrt{\frac{\sum_{i=1}^{N} (R_i - \text{MRE})^2}{N}} \tag{6.2}$$

where N is the total number of predicted landmarks.

Landmark's position is denoted as a single pixel in the ground truth annotations. For a landmark to be considered successfully detected, the distance between the predicted and annotated positions must be below z mm. The **success detection rate** (SDR) p_z with precision less than z mm is defined as

$$p_z = \frac{\#\{j : ||L_d(j) - L_a(j)|| < z\}}{\#\Omega} \times 100\% \tag{6.3}$$

where L_d, L_a are the locations of the detected and annotated landmarks respectively, z corresponds to the four precision measurements used in the 2015 challenge evaluation

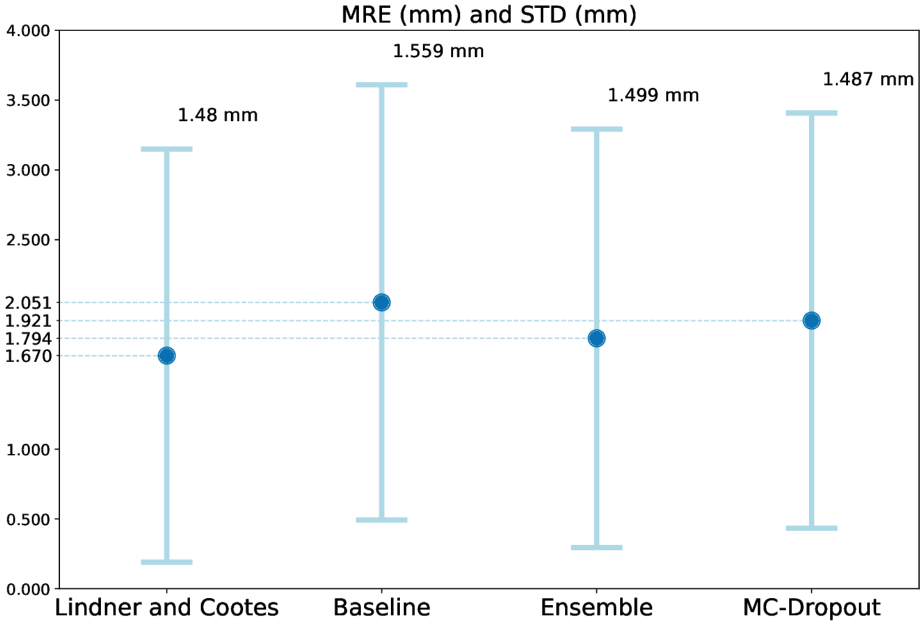


Figure 6.2: Mean radial errors (MREs) and standard deviations (STDs). Comparison between the proposed models and the best-performing method from the 2015 competition by Lindner and Cootes [45]. All four have been evaluated on the test1 split of the dataset.

(2 mm, 2.5 mm, 3 mm and 4 mm). The numerator contains the number of successfully detected landmarks and the denominator the total number of landmarks.

6.2.2 Model Performance

The three models were evaluated on the test1 split of the cephalometry dataset. Their performance is compared with the best-performing model (proposed by Lindner and Cootes) participating in the 2015 Grand Challenge in Dental X-ray Image Analysis [45]. Figures 6.1 and 6.2 compare the success detection rates and the MRE with the associated STD of the three models respectively.

The MC-Dropout model matches the performance of Lindner and Cootes in SDR closely in the 3 and 4 mm precision ranges but lags behind in the 2.5 and 2 mm ranges. The Ensemble model outperforms MC-Dropout in all of the SDR precision ranges and reaches significantly better detection rate in the 2.5 and 2 mm ranges (but still lags behind Lindner and Cootes). Using the MRE metric, Lindner and Cootes' model again outperforms the proposed models while Ensemble performs better than MC-Dropout. The Ensemble consistently outperforms a single Baseline model.

The performance drop of the proposed models in the 2.5 and 2 mm SDR ranges is expected, since they were trained using images sub-sampled to 128×128 from the original 1935×2400 dimensions due to computational limitations. A single pixel in the original image corresponds to 0.1 mm, while a single pixel in the sub-sampled image (and thus also in the heatmap predicted by the model which has the same dimensions) to roughly 1.7 mm.

The predicted landmark position must refer to a single pixel in the original dimensions so the model prediction which is restricted to the $[0, 128)$ range, must be re-scaled to them. However, the input images do not provide the model with enough information to predict the landmark position with such accuracy. This leads to a low SDR in the discussed

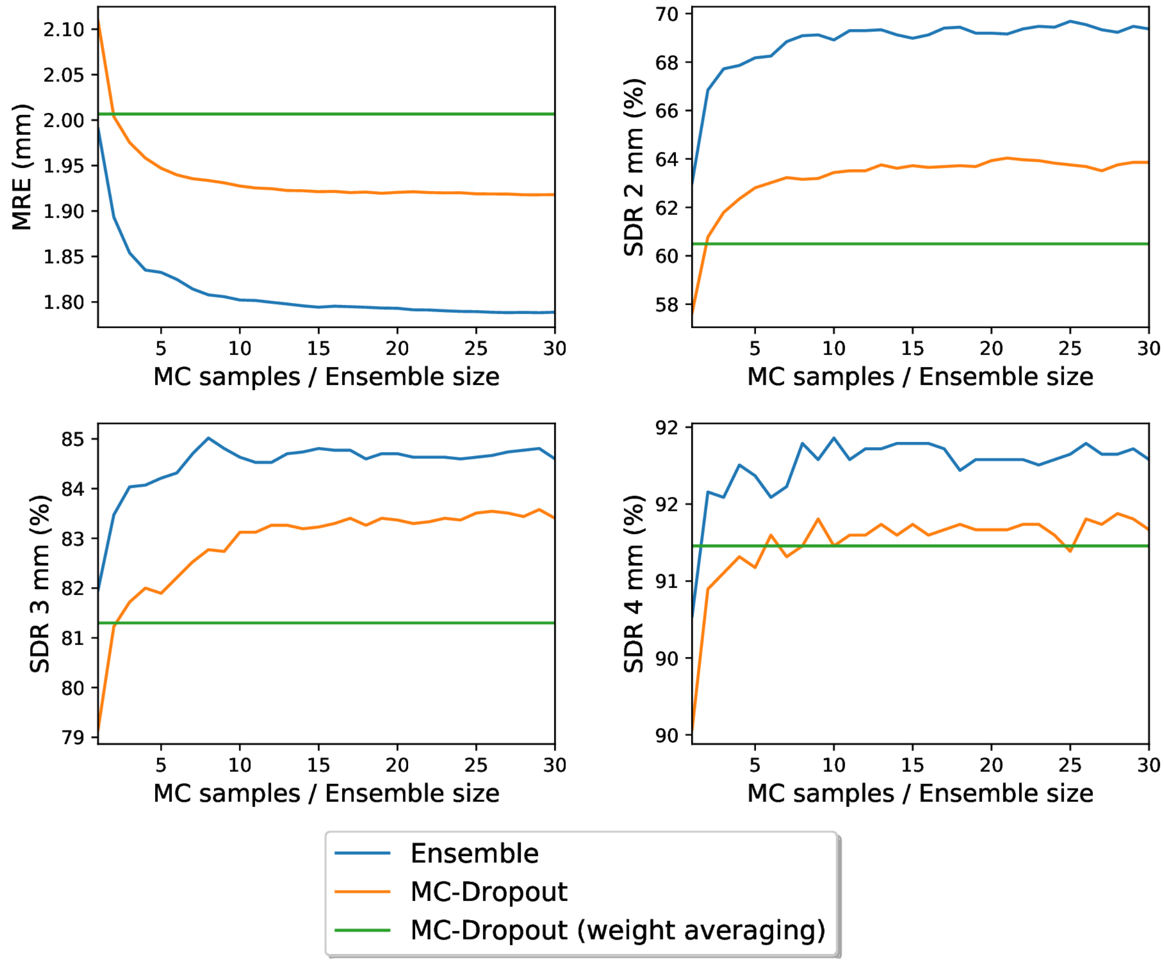


Figure 6.3: Model performance when varying ensemble size and dropout sample count. Ensemble size and sample count vary for the Ensemble and MC-Dropout models respectively. MC-Dropout is additionally evaluated using the commonly used weight averaging method (see Section 3.2.1). Increasing the MC sample count and ensemble size up to 15-20 samples/models increases performance on all metrics. Moreover, MC-Dropout with MC sampling starts outperforming a single Baseline model (i.e., Ensemble of size 1) and MC-Dropout with weight averaging when using at least 5 samples.

precision ranges. It also accounts for the relative drop in performance of the proposed models compared with Lindner and Cootes since the low SDR in the 2.5 and 2 mm ranges leads to a greater total MRE.

The better overall performance of Ensemble compared with MC-Dropout is attributable to the fact that an ensemble of 15 networks trained independently provides a greater variability of model parameters than 15 samples from a single dropout CNN. Ensemble can therefore generalize better to unobserved test data.

Figure 6.3 compares the performance of the Ensemble and MC-Dropout models on the test data for varying ensemble size and number of MC samples respectively. MC-Dropout's performance using the traditional weight averaging evaluation strategy (which uses a single prediction with disabled dropout at test time) is computed as well. Increasing the number of MC samples for MC-Dropout and ensemble size for Ensemble improves performance on all metrics, but a point of diminishing returns is reached after about 15-20 samples/ensemble members. Notably, MC-Dropout with MC sampling begins to outperform or match both a single Baseline model (i.e., Ensemble of size 1) and MC-Dropout with weight averaging in all metrics when using roughly 5 samples or more.

6.3 Uncertainty Measure Evaluation

The three uncertainty measures for the landmark localization task are evaluated using the methods described in Section 3.5. The raw model predictions from the Baseline network, Ensemble members' prediction variance and the MC-Dropout sample prediction variance are used. These uncertainty measures were proposed in Section 3.4. That description is further elaborated on here since their implementation is specific to the landmark localization task.

6.3.1 Uncertainty Measures

The Baseline model uses the raw predictions for uncertainty estimation. All models were trained to regress heatmaps containing a Gaussian activation at the position of the detected landmark (see Figure 4.1 and Section 5.1.3 describing the loss function). The heatmaps are convolved with a Gaussian kernel to compute the predicted landmark position. I hypothesized that the maximum heatmap activation after the convolution may indicate the model's uncertainty in its prediction, with higher values indicating higher certainty. For the purpose of analysis in the following experiments, this dimensionless quantity was normalized to a unit range. The upper bound of one for normalization was chosen based on the maximum value of this uncertainty measure observed for all of the landmarks in the test set.

The two other uncertainty measures used are both prediction variances. For the landmark localization task, I propose to compute prediction variance of a vector $\mathbf{y}$ containing prediction samples as the mean Euclidean distance between the prediction samples y_i and the prediction mean $\hat{y}$:

$$\hat{y} = \frac{1}{T} \sum_{i=1}^{T} y_i \tag{6.4}$$

$$\mathrm{Var}(\mathbf{y}) = \frac{1}{T} \sum_{i=1}^{T} \| y_i - \hat{y} \| \tag{6.5}$$

Note that the prediction mean $\hat{y}$ is also used as the landmark location predicted by the Ensemble and MC-Dropout models. The uncertainty measures along with model predictions for some of the test set images are visualized in Figure B.5.

6.3.2 Correlation between Uncertainty and Performance Measures

For each case of landmark prediction the trained models produce a landmark position and also an uncertainty measure value. For the evaluated test set images, the predicted landmark position is compared with the ground truth to produce a radial error in detection.

By correlating the individual uncertainty measure values with the respective radial errors for each landmark, we can determine whether a relationship exists between the uncertainty measure and the error. A strong correlation would indicate that the uncertainty measure is useful (i.e., that the model knows what it does not know). The results of this correlation analysis for the proposed models are shown in Figure 6.4.

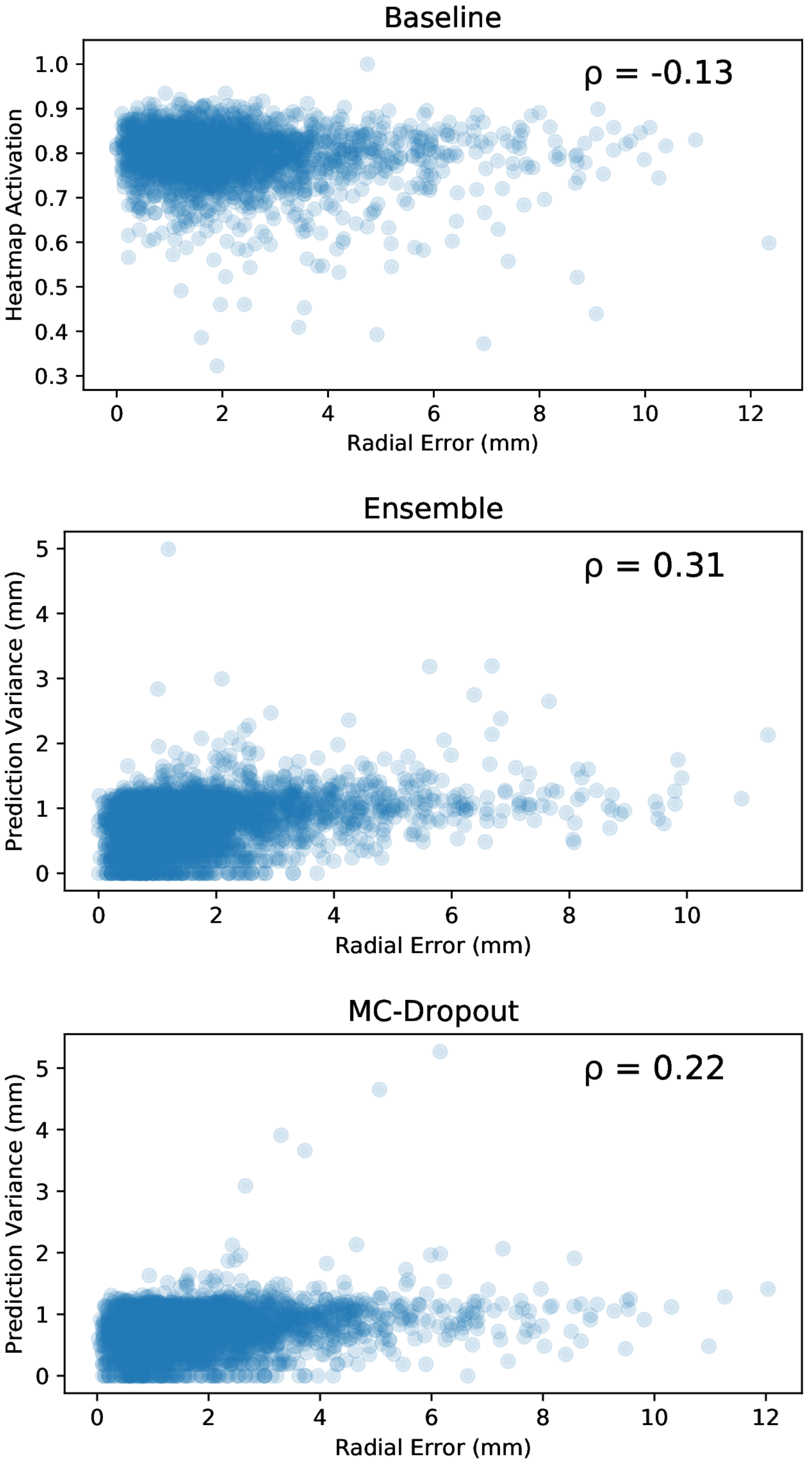


Figure 6.4: Correlation between radial error in landmark detection and observed uncertainty measure value for the landmark. ρ denotes the Pearson correlation coefficient. Each of the three models was evaluated on the test1 data split containing 2850 landmarks. For each detected landmark we observe a pair of values comprising radial error (when compared with the ground truth annotation) and model uncertainty (each model uses a different uncertainty measure). See Figure 6.8 for a comparison with under-trained models' performance.

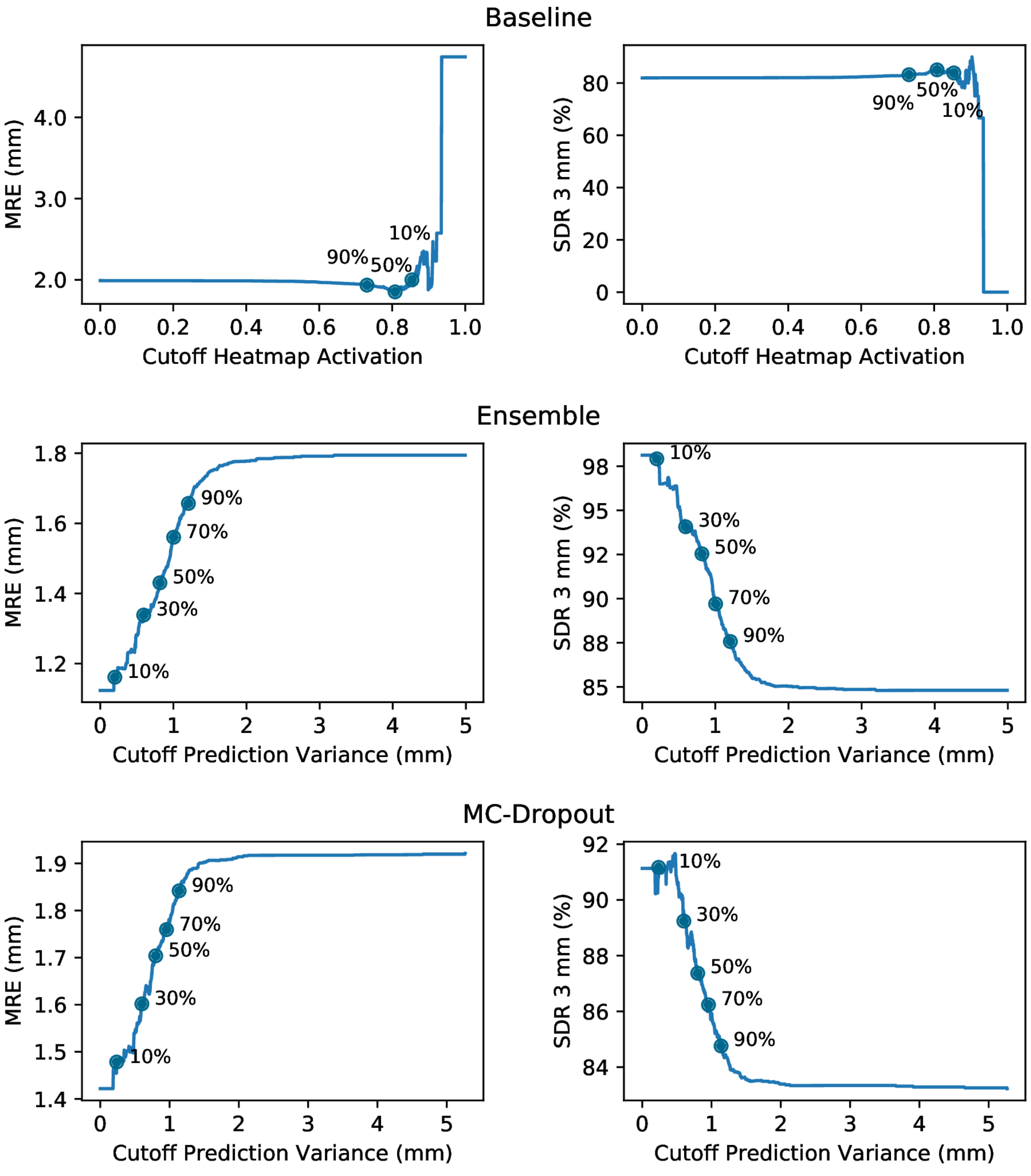


Figure 6.5: Test set filtering using different uncertainty thresholds. The models' performance is evaluated at different thresholds of their uncertainty measure using only a subset of the test data for which the model's uncertainty falls below the specified threshold (or above it in the case of heatmap activations). Graphs also contain the percentage of test data retained for evaluation for specific uncertainty thresholds. Both prediction variances provide a reasonable assessment of model uncertainty while filtering using the maximum heatmap activations does not. See Figure 6.9 for a comparison with under-trained models' performance.

The maximum heatmap activation performed poorly with a Pearson correlation coefficient $\rho = -0.13$ (a negative ρ indicates usefulness for this measure since we assume that the model uncertainty *decreases* as the activation value increases). However, since ρ is close to zero, the correlation is not very significant. The Ensemble's prediction variance attained $\rho = 0.31$ and the MC-Dropout's prediction variance a $\rho = 0.22$ which suggest a greater degree of usefulness for these measures.

6.3.3 Test Set Filtering

The three uncertainty measures are next evaluated using the test set filtering strategy (see Section 3.5.2 for a detailed description). When applying this strategy, we evaluate the model on some performance metric but only use that subset of the test data, for which the model's uncertainty is below a certain threshold. If the uncertainty measure is useful, decreasing the threshold should lead to a better model performance on the retained data. The results of the analysis along with the percentage of retained data for specific uncertainty thresholds are depicted in Figure 6.5. The MRE and SDR in the 3 mm precision range are used as the evaluated performance metrics (the results were very similar for different SDR precision ranges).

The best performing uncertainty measure using this evaluation strategy is Ensemble's prediction variance with an MRE of 1.16 mm and SDR of 98% at the 10% data retention point, and MRE of 1.66 mm and SDR of 88% at the 90% data retention point. MC-Dropout's prediction variance attained an MRE of 1.48 mm and SDR of 91% at the 10% data retention point and MRE of 1.84 mm and SDR of 85% at the 90% data retention point. The plotted behavior of both uncertainty measures appears to be very similar. Better performance of the Ensemble model can be explained by a larger variety of predictions since it contains different models trained from scratch, and MC-Dropout model is limited by the capacity of a single neural network.

Maximum heatmap activations of the Baseline model did not provide a lot of useful information using this evaluation strategy. As the value of the threshold increased (so that the maximum heatmap activation for a given landmark had to be above it for it to be kept within the evaluated test set) the performance measures increased very slightly until a certain point after which they started abruptly decreasing. However this decrease occurred at less then 10% data retention point so it may not be of significant importance.

6.3.4 Elastically Distorted Out-of-distribution Data

Elastic distortion [41] was applied to the entire test set to evaluate the ability of the uncertainty measures to detect out-of-distribution data examples. Forty versions of the test set were created in total, and each copy had an elastic distortion of progressively stronger magnitude applied to it. Figure B.1 shows a test image transformed with an elastic distortion of varying magnitude.

Each model's predictions and uncertainty estimates for every version of the distorted test set was then computed. Figure 6.6 shows the correlation between the mean uncertainty measure value for all landmark position predictions for a given version of the test set, and the elastic distortion magnitude applied to that version of the test set. The analysis shows that a strong correlation exists between the mean value of each uncertainty measure and the strength of the elastic distortion.

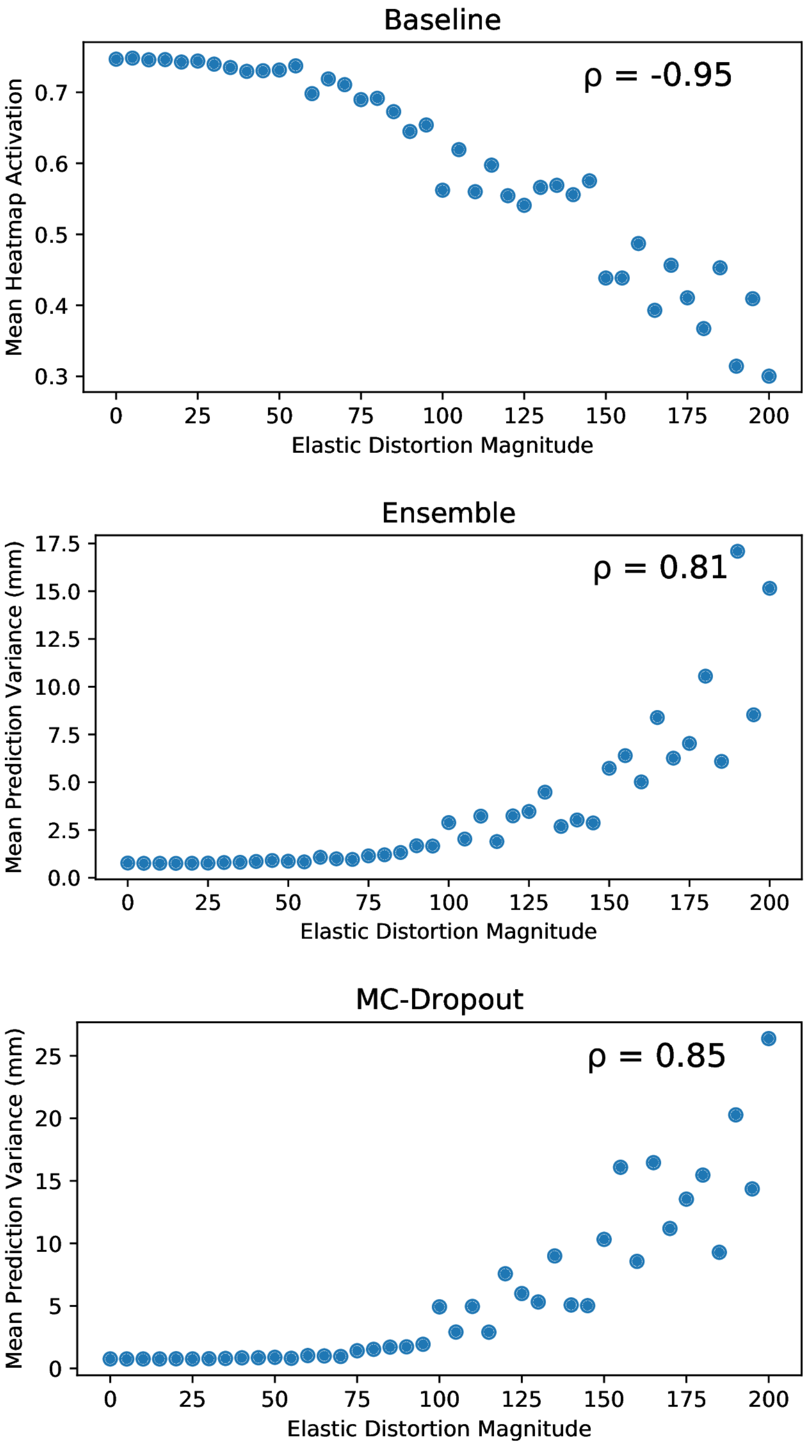


Figure 6.6: Correlation between elastic distortion magnitude and mean uncertainty measure on the distorted test set. Each of the models along with its uncertainty measure was evaluated on forty versions of the test set modified by elastic distortions of increasing magnitude. The mean uncertainty measure value on the entire test set was then correlated with the magnitude of the elastic distortion applied to the test set. Each uncertainty measure is able to detect out-of-distribution test examples.

The maximum heatmap activations of the Baseline model (which performed poorly in the test set filtering experiments) show the strongest Pearson correlation coefficient $\rho = -0.95$. The MC-Dropout and Ensemble show ρ of 0.85 and 0.81 respectively. This suggests that each of the measures could reliably be used to detect out-of-distribution data examples. Figure B.2 contains a visualization of model predictions and uncertainty measure values for a test image elastically distorted with different magnitudes.

6.3.5 Laterally Rotated Out-of-distribution Data

When creating a cephalogram, we would ideally want the patient's head to be aligned perfectly with the sagittal plane so that there is no rotation in either of the lateral directions. This is not always the case however and the lateral rotation of the skull distorts the resulting image and may even lead to some of the landmarks overlapping. A model should be able to detect these data examples and possibly alert the physician that they cannot be analyzed automatically.

Since a dataset of laterally rotated cephalograms is not publicly available, volumetric CT scan of a skull was used to create one. The skull volume was first laterally rotated by θ degrees in the axial plane. The resulting volume was then projected onto the sagittal plane by summing the intensity values of overlapping voxels. Pixel values in the resulting 2D image were then normalized by dividing by the maximum pixel intensity within the image. The resulting dataset contains 91 images with θ ranging from -45° to 45° including a rotation of 0°.

The models' prediction and uncertainty measure values were then evaluated for each image in the dataset. Figure 6.7 shows the correlation analysis between the mean uncertainty value for a given image (computed as the mean of uncertainty estimates for all of the landmarks predicted for the image) and the magnitude of rotation corresponding to that image. The results are similar to those depicted in Figure 6.6 for the experiment with the elastically distorted test set. All of the uncertainty measures show a very strong correlation between the two sets of values and could be used to detect the misaligned patient data. Figure B.4 contains a visualization of model predictions and uncertainty measure values for the skull projection rotated by different magnitudes θ.

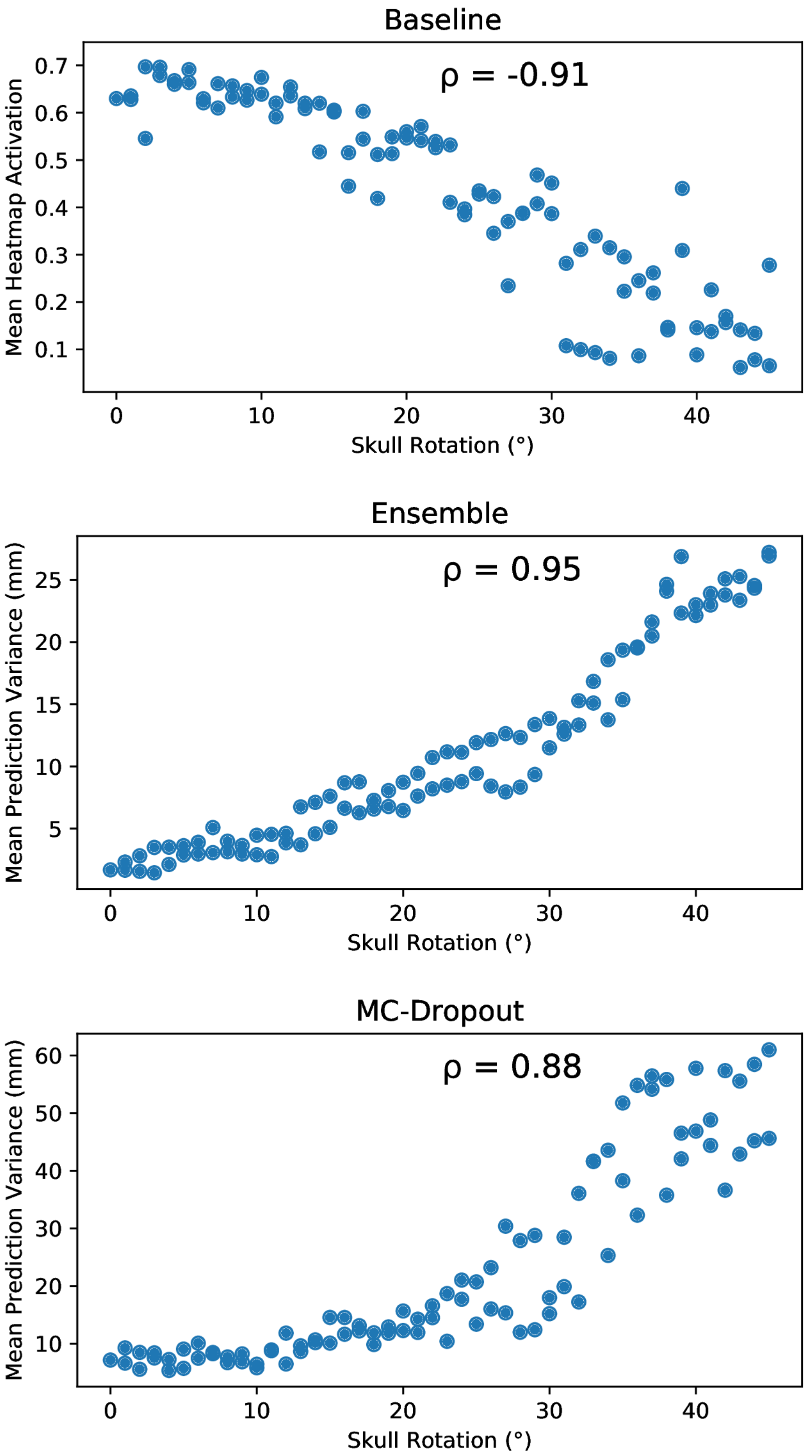


Figure 6.7: Correlation between the mean uncertainty measure for an image and the corresponding skull rotation. Each of the models along with its uncertainty measure was evaluated on 91 images of a skull CT scan projection onto the sagittal plane. The scan was laterally rotated before projection with various magnitudes. The mean uncertainty measure value for an image was then correlated with the rotation magnitude.

	MRE	STD	SDR 2 mm	SDR 2.5 mm	SDR 3 mm	SDR 4 mm
Baseline	2.05 mm	1.56 mm	63.0 %	74.4 %	81.9 %	90.8 %
Ensemble	1.79 mm	1.50 mm	69.0 %	78.5 %	84.8 %	91.9 %
MC-Dropout	1.92 mm	1.49 mm	63.7 %	74.7 %	83.2 %	91.4 %
Baseline-UT	6.43 mm	12.99 mm	19.1 %	27.0 %	36.0 %	53.5 %
Ensemble-UT	4.99 mm	4.83 mm	22.1 %	31.4 %	40.6 %	57.0 %
MC-Dropout-UT	8.30 mm	12.76 mm	15.6 %	21.5 %	29.3 %	42.2 %

Table 6.1: Performance comparison of fully-trained and under-trained models. Models were evaluated on the test1 split of the dataset. The under-trained (UT) models were trained using the same procedure as the fully-trained models but the training was stopped before convergence.

6.3.6 Under-trained Models

The analysis of the quality of the proposed uncertainty measures in Section 6.3.2 and visualized in Figure 6.4 suggested a meaningful correlation between the test set radial error in landmark localization prediction and the corresponding uncertainty measure value for all of them. However, it was not particularly strong for any of the measures, with the maximum attained Pearson correlation coefficient $\rho = 0.31$ for Ensemble's prediction variance and $\rho = -0.13$ close to zero for Baseline's heatmap activations.

The relatively poor performance of the uncertainty measures in the error-uncertainty correlation experiment on one hand, and their solid performance in the out-of-distribution data detection experiments indicate that the unmodified test data used for evaluation in Section 6.2.2 may have been overly similar to the training distribution, and that the fully-trained models performed too well. I therefore investigated the possibility that the uncertainty measures become more useful (and the correlation between performance and uncertainty estimate more apparent) when the model performance is not too high on the data.

Since an annotated test dataset with a different distribution was not available, the drop in model performance it would have caused was simulated by creating under-trained versions of all three models, and evaluating them on the same test set as the fully-trained models.

The under-trained models termed Baseline-UT, Ensemble-UT and MC-Dropout-UT were trained using the same procedure as the fully-trained models but the training was stopped once the MRE metric on the validation data dropped below 8.0 mm. Table 6.1 compares the performance of all models on the test1 split of the dataset and Figure B.6 visualizes the model predictions and uncertainty measure values for some of the test set images and landmarks.

The under-trained models were then used to perform the same correlation and test set filtering experiments as were described for the fully-trained models in Sections 6.3.2 and 6.3.3 respectively. Figure 6.8 depicts the error-uncertainty correlation analysis and Figure 6.9 the results of the test set filtering experiment.

Both MC dropout and ensemble prediction variances showed a very strong correlation with a Pearson correlation coefficient $\rho = 0.86$ and 0.85 respectively (a significant improvement over the fully-trained models' $\rho = 0.22$ and 0.31). The maximum heatmap activation uncertainty measure performed better when utilized by the under-trained model as well, achieving a $\rho = -0.35$ which is an improvement over the fully-trained model's $\rho = -0.13$.

Both prediction variances performed very well in the test set filtering which was the case even when using the fully-trained models. Additionally, the maximum heatmap activation also showed more useful results when compared with its performance when utilized by the fully-trained Baseline model. For Baseline-UT, removal of test set data based on this measure led to both MRE and SDR metrics improvement in the roughly 100% to 20% data retention range followed by a more erratic behavior when less than 10% of the data was retained.

6.3.7 Summary of Experiments

The experiment in Section 6.3.2 analyzed the correlation between the uncertainty estimates and the error in the predicted landmark on the test set. The test set filtering experiment in Section 6.3.3 analyzed whether removing examples with uncertain predictions from the data increased performance. Both uncertainty measures based on prediction variance used by the Ensemble and MC-Dropout models performed relatively well in these experiments, while Baseline's heatmap activation measure performed poorly in both.

Both experiments were repeated in Section 6.3.6 with under-trained models evaluated on the same test set. All of the three uncertainty measures showed a much stronger uncertainty-error correlation and also performed better on the test set filtering experiment. Since the data in the test set is very similar to the training data, the strong improvements for all measures when used by under-trained models suggests that they perform better as the models' familiarity with the data decreases. This might be useful in real-world applications where the system often has to deal with data that is different from the training examples.

The additional two experiments analyzed the uncertainty measures' ability to detect out-of-distribution data. Experiment in Section 6.3.4 used copies of the test dataset altered with different elastic distortion magnitudes. The one described in Section 6.3.5 used a dataset of cephalograms created from a CT scan of a skull, which was laterally rotated by different angles. Both experiments confirmed the ability of all three uncertainty measures to detect examples which differ from the training distribution. This is useful when deciding whether to classify the data points automatically or pass them on to a physician for manual processing.

6.3.8 Further Research

The research conducted here could continue in several directions. A test set with cephalograms captured by a different device would be useful to confirm the conclusions of the performed experiments. The models could also be trained for landmark localization on distinct data entirely (not necessarily from the medical domain) to analyze whether the uncertainty measures show similar behavior across datasets.

The uncertainty measures could additionally be evaluated on another task such as classification or segmentation. This would allow us to determine whether their performance characteristics transfer across to different problems or not. This would of course only be applicable to the task-agnostic prediction variance measures and not to heatmap activations which are specific to landmark localization using heatmap regression. On the other hand, different tasks may allow the use of new uncertainty measures which were not analyzed in this work.

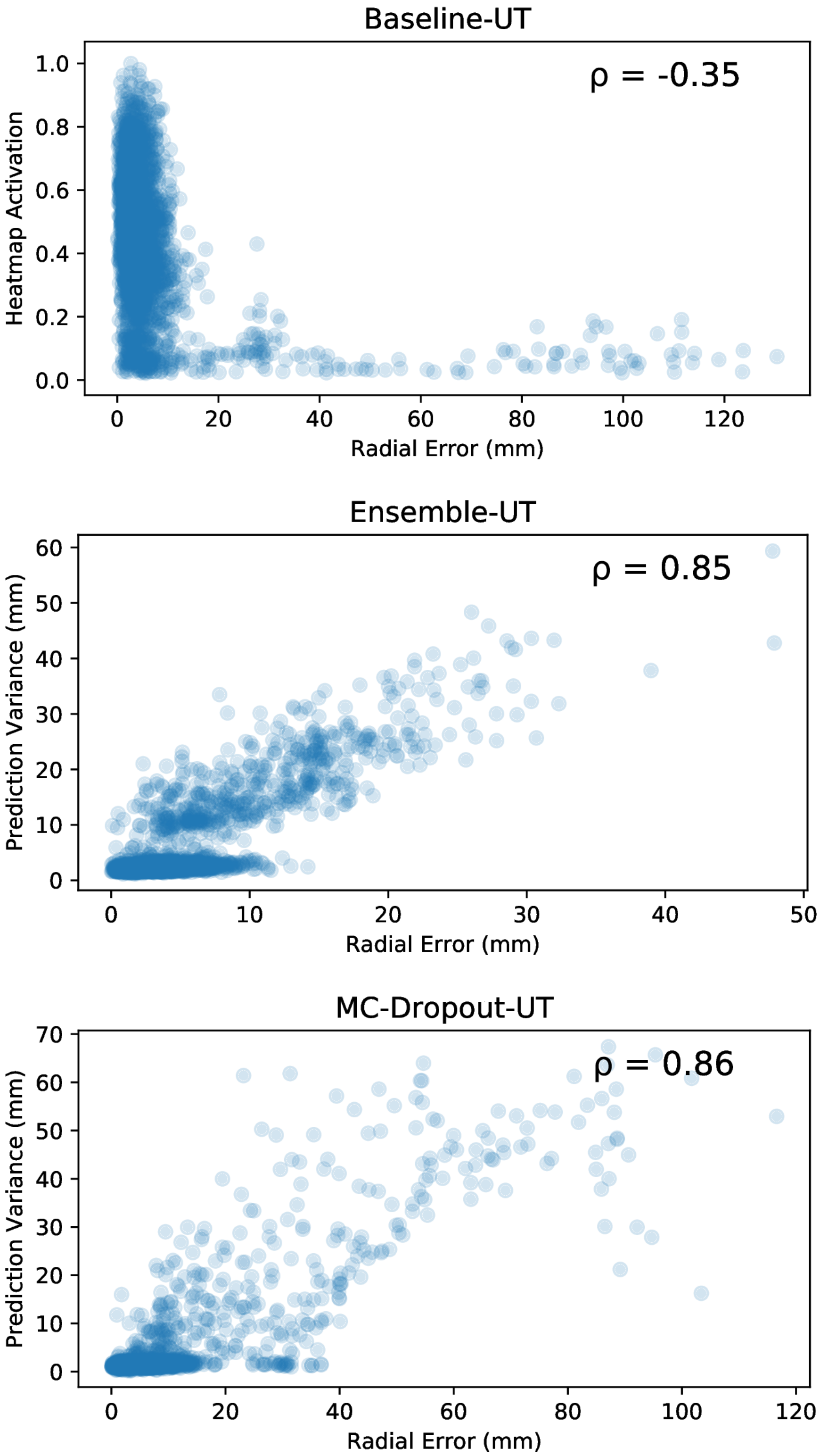


Figure 6.8: Correlation between radial error in landmark detection and observed uncertainty measure value for the landmark for under-trained models. Refer to Figure 6.4 for a more detailed description of the experiment evaluation procedure and for comparison with the fully-trained models. The correlation between the radial error and the respective model uncertainty measures is much stronger for all under-trained models (especially for both prediction variances).

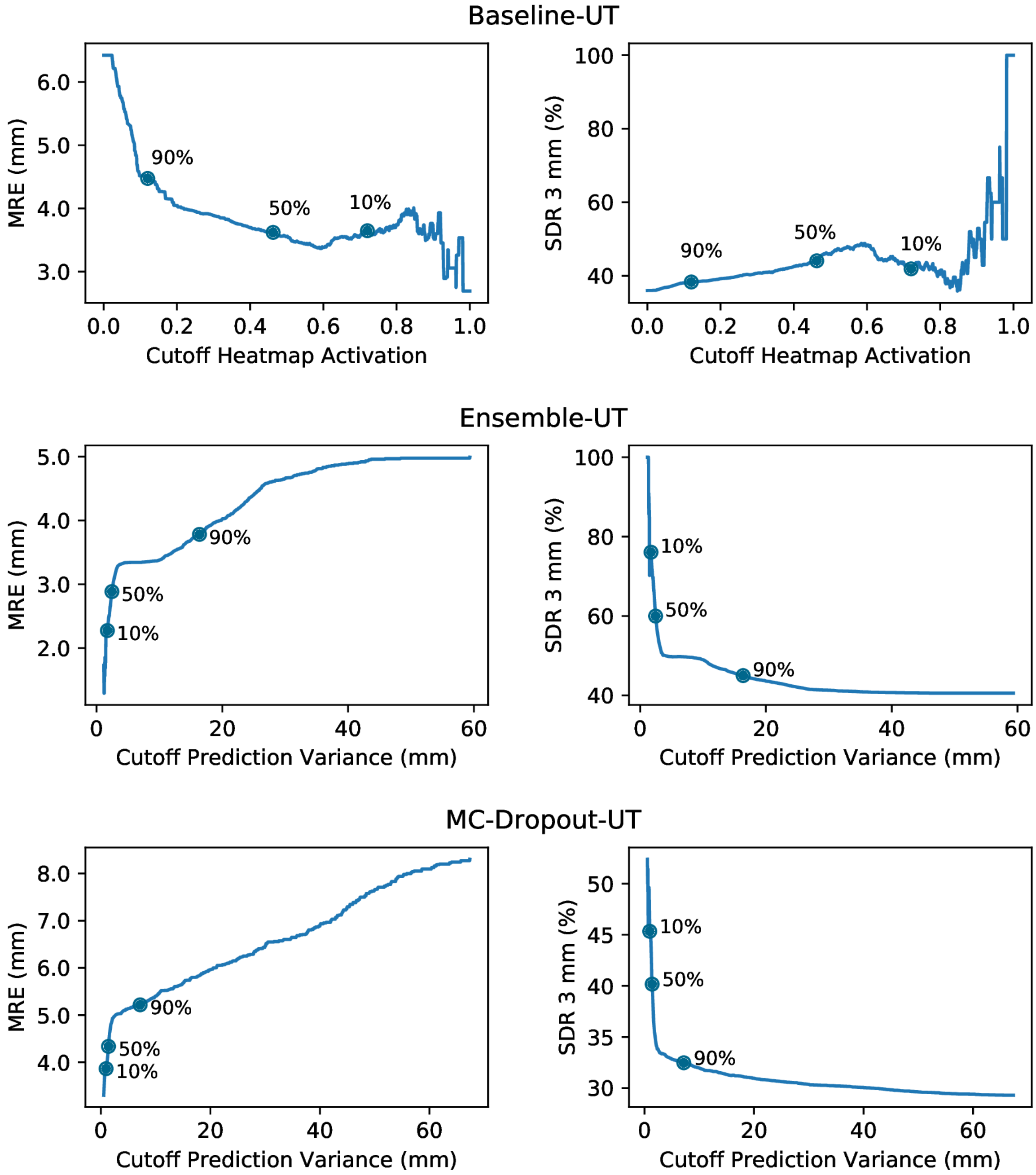


Figure 6.9: Test set filtering using different uncertainty thresholds for under-trained models. Refer to Figure 6.5 for a more detailed description of the experiment evaluation procedure and for comparison with the fully-trained models. For the under-trained versions of the models, all three uncertainty measures provide a reasonable assessment of model uncertainty.

Chapter 7

Conclusion

This thesis dealt with the problem of estimating uncertainty of deep learning model predictions in medical image analysis. A landmark localization task on a dataset of X-ray cephalograms was chosen to explore two prediction variance based approaches from recent research in this field, as well as a third approach to uncertainty estimation based on model heatmap predictions proposed by the author. A CNN architecture was designed for this purpose and used for training three models, each of which provides a prediction along with an uncertainty value.

Baseline is a single CNN without dropout layers which uses the maximum activation of the heatmap prediction regressed for each landmark as its uncertainty measure. Ensemble is an ensemble of 15 Baseline models following the ideas in the work of Lakshminarayanan et al. [20]. MC-Dropout uses the approach proposed primarily by Gal [9][7][8] which recasts CNNs with dropout layers as Bayesian models. Both use the prediction variance of ensemble members or MC dropout samples respectively to estimate model uncertainty. All models performed comparably with the state-of-the art approach on the landmark localization task.

Based on the experimental results, the Ensemble and MC-Dropout prediction variances outperformed the Baseline's heatmap activation measure when evaluated on the test set. The maximum heatmap activation value failed to consistently correlate with the landmark error and also failed to reliably filter the test dataset using its uncertainty estimates. The reliable performance of prediction variance across all experiments suggests that it is superior to the raw model predictions. Both prediction variances showed similar characteristics across experiments and the slightly superior performance of the Ensemble's variance is attributable to a greater number of model parameters. The heatmap activations should thus only be used to estimate uncertainty if computational resources are limited and it is not possible to use ensemble or MC sample prediction variance.

It is noteworthy that the performance of all measures improved when using the undertrained models. All three measures also performed very well in the out-of-distribution data detection experiments with elastically distorted images and laterally rotated skull cephalograms. This suggests that their uncertainty estimates are more reliable when evaluated on data that the models are not overly familiar with. This is an important result for real-world applications in which a model has to robustly deal with a wide range of data.

The research conducted here could be extended by evaluating the uncertainty measures' performance on a different dataset or on a different task entirely as described in Section 6.3.8.

Bibliography

[1] Bishop, C. M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Berlin, Heidelberg: Springer-Verlag. 2006. ISBN 0387310738.

[2] Blei, D. M.; Kucukelbir, A.; McAuliffe, J. D.: Variational Inference: A Review for Statisticians. *Journal of the American Statistical Association*. vol. 112, no. 518. 2017: pp. 859–877. doi:10.1080/01621459.2017.1285773.
https://doi.org/10.1080/01621459.2017.1285773.
Retrieved from: https://doi.org/10.1080/01621459.2017.1285773

[3] Ciresan, D.; Giusti, A.; Gambardella, L. M.; et al.: Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images. In *Advances in Neural Information Processing Systems 25*, edited by F. Pereira; C. J. C. Burges; L. Bottou; K. Q. Weinberger. Curran Associates, Inc.. 2012. pp. 2843–2851.
Retrieved from: http://papers.nips.cc/paper/4741-deep-neural-networks-segment-neuronal-membranes-in-electron-microscopy-images.pdf

[4] Dawid, A. P.: The Well-Calibrated Bayesian. *Journal of the American Statistical Association*. vol. 77, no. 379. 1982: pp. 605–610. ISSN 01621459.
Retrieved from: http://www.jstor.org/stable/2287720

[5] Dietterich, T. G.: Ensemble Methods in Machine Learning. In *Multiple Classifier Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg. 2000. ISBN 978-3-540-45014-6. pp. 1–15.

[6] Esteva, A.; Kuprel, B.; Novoa, R. A.; et al.: Dermatologist-level classification of skin cancer with deep neural networks. *Nature*. vol. 542. Jan 2017: pp. 115 EP –.
Retrieved from: https://doi.org/10.1038/nature21056

[7] Gal, Y.: *Uncertainty in Deep Learning*. PhD. Thesis. University of Cambridge. 2016.

[8] Gal, Y.; Ghahramani, Z.: Bayesian Convolutional Neural Networks with Bernoulli Approximate Variational Inference. 06 2015.

[9] Gal, Y.; Ghahramani, Z.: Dropout As a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*. ICML'16. JMLR.org. 2016. pp. 1050–1059.
Retrieved from: http://dl.acm.org/citation.cfm?id=3045390.3045502

[10] Gneiting, T.; Raftery, A. E.: Strictly Proper Scoring Rules, Prediction, and Estimation. *Journal of the American Statistical Association*. vol. 102, no. 477. 2007:

pp. 359–378. doi:10.1198/016214506000001437.
https://doi.org/10.1198/016214506000001437.
Retrieved from: https://doi.org/10.1198/016214506000001437

[11] Goodfellow, I. J.; Shlens, J.; Szegedy, C.: Explaining and Harnessing Adversarial Examples. *CoRR*. vol. abs/1412.6572. 2014.

[12] Greenspan, H.; van Ginneken, B.; Summers, R. M.: Guest Editorial Deep Learning in Medical Imaging: Overview and Future Promise of an Exciting New Technique. *IEEE Transactions on Medical Imaging*. vol. 35, no. 5. May 2016: pp. 1153–1159. ISSN 0278-0062. doi:10.1109/TMI.2016.2553401.

[13] Hinton, G. E.; Srivastava, N.; Krizhevsky, A.; et al.: Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*. vol. abs/1207.0580. 2012. 1207.0580.
Retrieved from: http://arxiv.org/abs/1207.0580

[14] Ioffe, S.; Szegedy, C.: Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR*. vol. abs/1502.03167. 2015. 1502.03167.
Retrieved from: http://arxiv.org/abs/1502.03167

[15] Kendall, A.; Badrinarayanan, V.; Cipolla, R.: Bayesian SegNet: Model Uncertainty in Deep Convolutional Encoder-Decoder Architectures for Scene Understanding. *CoRR*. vol. abs/1511.02680. 2015. 1511.02680.
Retrieved from: http://arxiv.org/abs/1511.02680

[16] Kendall, A.; Gal, Y.: What Uncertainties Do We Need in Bayesian Deep Learning for Computer Vision? In *Advances in Neural Information Processing Systems 30 (NIPS)*. 2017.

[17] Kingma, D. P.; Ba, J.: Adam: A Method for Stochastic Optimization. *CoRR*. vol. abs/1412.6980. 2014. 1412.6980.
Retrieved from: http://arxiv.org/abs/1412.6980

[18] Krizhevsky, A.; Sutskever, I.; Hinton, G. E.: ImageNet Classification with Deep Convolutional Neural Networks. In *Proceedings of the 25th International Conference on Neural Information Processing Systems - Volume 1*. NIPS'12. USA: Curran Associates Inc.. 2012. pp. 1097–1105.
Retrieved from: http://dl.acm.org/citation.cfm?id=2999134.2999257

[19] Kullback, S.; Leibler, R. A.: On Information and Sufficiency. *Ann. Math. Statist.*. vol. 22, no. 1. 1951: pp. 79–86.

[20] Lakshminarayanan, B.; Pritzel, A.; Blundell, C.: Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems 30*, edited by I. Guyon; U. V. Luxburg; S. Bengio; H. Wallach; R. Fergus; S. Vishwanathan; R. Garnett. Curran Associates, Inc.. 2017. pp. 6402–6413.
Retrieved from: http://papers.nips.cc/paper/7219-simple-and-scalable-predictive-uncertainty-estimation-using-deep-ensembles.pdf

[21] Lakshminarayanan, B.; Pritzel, A.; Blundell, C.: Simple and Scalable Predictive Uncertainty Estimation using Deep Ensembles. In *Advances in Neural Information Processing Systems 30*, edited by I. Guyon; U. V. Luxburg; S. Bengio; H. Wallach; R. Fergus; S. Vishwanathan; R. Garnett. Curran Associates, Inc.. 2017. pp. 6402–6413.
Retrieved from: http://papers.nips.cc/paper/7219-simple-and-scalable-predictive-uncertainty-estimation-using-deep-ensembles.pdf

[22] Lecun, Y.; Bottou, L.; Bengio, Y.; et al.: Gradient-based learning applied to document recognition. *Proceedings of the IEEE*. vol. 86, no. 11. Nov 1998: pp. 2278–2324. ISSN 0018-9219. doi:10.1109/5.726791.

[23] LeCun, Y.; Cortes, C.: MNIST handwritten digit database. 2010.
Retrieved from: http://yann.lecun.com/exdb/mnist/

[24] Lee, S.; Purushwalkam, S.; Cogswell, M.; et al.: Why M Heads are Better than One: Training a Diverse Ensemble of Deep Networks. *CoRR*. vol. abs/1511.06314. 2015.

[25] Leibig, C.; Allken, V.; Ayhan, M. S.; et al.: Leveraging uncertainty information from deep neural networks for disease detection. *Scientific Reports*. vol. 7. 12 2017. doi:10.1038/s41598-017-17876-z.

[26] Lindner, C.; Wang, C.-W.; Huang, C.-T.; et al.: Fully Automatic System for Accurate Localisation and Analysis of Cephalometric Landmarks in Lateral Cephalograms. *Scientific Reports*. vol. 6. 09 2016: page 33581. doi:10.1038/srep33581.

[27] Litjens, G. J. S.; Kooi, T.; Bejnordi, B. E.; et al.: A survey on deep learning in medical image analysis. *Medical image analysis*. vol. 42. 2017: pp. 60–88.

[28] Long, J.; Shelhamer, E.; Darrell, T.: Fully convolutional networks for semantic segmentation. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2015. ISSN 1063-6919. pp. 3431–3440. doi:10.1109/CVPR.2015.7298965.

[29] Neal, R. M.: *Bayesian Learning for Neural Networks*. Berlin, Heidelberg: Springer-Verlag. 1996. ISBN 0387947248.

[30] Nix, D. A.; Weigend, A. S.: Estimating the mean and variance of the target probability distribution. In *Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94)*, vol. 1. June 1994. pp. 55–60 vol.1. doi:10.1109/ICNN.1994.374138.

[31] Paszke, A.; Gross, S.; Chintala, S.; et al.: Automatic differentiation in PyTorch. In *NIPS-W*. 2017.

[32] Payer, C.; Stern, D.; Bischof, H.; et al.: Regressing Heatmaps for Multiple Landmark Localization Using CNNs. In *MICCAI*. 2016.

[33] Pfister, T.; Charles, J.; Zisserman, A.: Flowing ConvNets for Human Pose Estimation in Videos. *CoRR*. vol. abs/1506.02897. 2015. 1506.02897.
Retrieved from: http://arxiv.org/abs/1506.02897

[34] Rajchl, M.; Lee, M. C. H.; Schrans, F.; et al.: Learning under Distributed Weak Supervision. *CoRR*. vol. abs/1606.01100. 2016.

[35] Rajpurkar, P.; Irvin, J.; Zhu, K.; et al.: CheXNet: Radiologist-Level Pneumonia Detection on Chest X-Rays with Deep Learning. *CoRR*. vol. abs/1711.05225. 2017. 1711.05225.
Retrieved from: http://arxiv.org/abs/1711.05225

[36] Ronneberger, O.; Fischer, P.; Brox, T.: U-Net: Convolutional Networks for Biomedical Image Segmentation. In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, edited by N. Navab; J. Hornegger; W. M. Wells; A. F. Frangi. Cham: Springer International Publishing. 2015. pp. 234–241.

[37] Rusk, N.: Deep learning. *Nature Methods*. vol. 13. 12 2015: pp. 35–35. doi:10.1038/nmeth.3707.

[38] Russakovsky, O.; Deng, J.; Su, H.; et al.: ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*. vol. 115, no. 3. 2015: pp. 211–252. doi:10.1007/s11263-015-0816-y.

[39] Shen, D.; Wu, G.; Suk, H.-I.: Deep Learning in Medical Image Analysis. *Annu Rev Biomed Eng*. vol. 19. Jun 2017: pp. 221–248. ISSN 1545-4274. doi:10.1146/annurev-bioeng-071516-044442. 28301734[pmid].
Retrieved from: https://www.ncbi.nlm.nih.gov/pubmed/28301734

[40] Shih-Chung Benedict Lo, M. T. F. S. K. M., Jyh-Shyan Lin: Computer-assisted diagnosis of lung nodule detection using artificial convoultion neural network. 1993. doi:10.1117/12.154572.
Retrieved from: https://doi.org/10.1117/12.154572

[41] Simard, P. Y.; Steinkraus, D.; Platt, J. C.: Best practices for convolutional neural networks applied to visual document analysis. In *Seventh International Conference on Document Analysis and Recognition, 2003. Proceedings.*. Aug 2003. pp. 958–963. doi:10.1109/ICDAR.2003.1227801.

[42] Smith, L.; Gal, Y.: Understanding Measures of Uncertainty for Adversarial Example Detection. 03 2018.

[43] Srivastava, N.; Hinton, G.; Krizhevsky, A.; et al.: Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*. vol. 15. 2014: pp. 1929–1958.
Retrieved from: http://jmlr.org/papers/v15/srivastava14a.html

[44] Tajbakhsh, N.; Shin, J. Y.; Gurudu, S. R.; et al.: Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning? *IEEE Transactions on Medical Imaging*. vol. 35, no. 5. May 2016: pp. 1299–1312. ISSN 0278-0062. doi:10.1109/TMI.2016.2535302.

[45] Wang, C.-W.; Huang, C.-T.; Lee, J.-H.; et al.: A benchmark for comparison of dental radiography analysis algorithms. *Medical Image Analysis*. vol. 31. 2016: pp. 63 – 76. ISSN 1361-8415. doi:https://doi.org/10.1016/j.media.2016.02.004.

Retrieved from:
http://www.sciencedirect.com/science/article/pii/S1361841516000190

[46] Weigert, M.; Schmidt, U.; Boothe, T.; et al.: Content-Aware Image Restoration: Pushing the Limits of Fluorescence Microscopy. *bioRxiv*. 2017. doi:10.1101/236463. https://www.biorxiv.org/content/early/2017/12/19/236463.full.pdf.
Retrieved from: https://www.biorxiv.org/content/early/2017/12/19/236463

[47] Widdowson, S.; Taylor, D.: The management of grading quality: good practice in the quality assurance of grading. Tech Report. 2016.
Retrieved from:
https://assets.publishing.service.gov.uk/government/uploads/system/uploads/attachment_data/file/512832/The_Management_of_Grading.pdf

Appendix A

Review of Useful Concepts

A.1 Bayesian Modelling

This section contains the review of the theoretical aspects of Bayesian modelling which may be useful for understanding how the dropout technique can be used as an approximation to Bayesian inference in deep learning models as described in section 3.2.1 and on-wards.

A.1.1 Bayesian Inference

Considering training inputs $\mathbf{X} = \{\mathbf{x}_1, ..., \mathbf{x}_N\}$ and their corresponding outputs (labels) $\mathbf{Y} = \{\mathbf{y}_1, ..., \mathbf{y}_N\}$, in Bayesian (parametric) modelling we want to find the parameters $\mathbf{w}$ of a function $\mathbf{y} = \mathbf{f}^{\mathbf{w}}(\mathbf{x})$ that are likely to have generated the outputs $\mathbf{Y}$ from the inputs $\mathbf{X}$. The Bayesian approach suggests putting a *prior* distribution $p(\mathbf{w})$ over the space of possible model parameters. It represents our prior belief about which parameters are likely to have generated the data before we actually observe any data values. To assess how likely some particular value of parameters $\mathbf{w}$ was to generate the outputs we define a *likelihood* distribution $p(\mathbf{y}|\mathbf{x}, \mathbf{w})$ which is used to generate the outputs from the inputs given a value of $\mathbf{w}$ [7].

The final goal is then the calculation of the *posterior* distribution over the space of parameters $\mathbf{w}$. This distribution captures our updated belief about which parameters are most likely to have generated the data *after* we have observed the data values. Using the Bayes' theorem we get:

$$p(\mathbf{w}|\mathbf{X}, \mathbf{Y}) = \frac{p(\mathbf{Y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})}{p(\mathbf{Y}|\mathbf{X})} \tag{A.1}$$

Having defined this distribution, we can now predict the output $\mathbf{y}^*$ for a new input $\mathbf{x}^*$ by integrating

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) = \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})p(\mathbf{w}|\mathbf{X}, \mathbf{Y})d\mathbf{w} \tag{A.2}$$

which is a process we call (Bayesian) inference [7][1]. To compute the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ we must be able to evaluate the denominator, also called *model evidence*:

[1] Note that in deep learning the term inference usually refers to predicting an output of a model at test time. In Bayesian modelling it refers to integration over the model parameters which can also be done during training (by approximate inference which is a process of approximating this integral) [7].

$$(\mathbf{Y}|\mathbf{X}) = \int p(\mathbf{Y}|\mathbf{X}, \mathbf{w})p(\mathbf{w})d\mathbf{w} \tag{A.3}$$

By using the sum rule, we have marginalized the likelihood over $\mathbf{w}$. Marginalization is a core concept in Bayesian modelling. Ideally, we would like to marginalize over all uncertain quantities. We especially want to average over all possible model parameters $\mathbf{w}$ weighted by their prior plausibility $p(\mathbf{w})$. This can be done analytically for some simple models such as Bayesian linear regression but analytic solutions to marginalization do not exist for more interesting models. In such cases we have to turn to approximations [7].

A.1.2 Variational Inference

Variational inference is a method which aims to approximate an otherwise difficult-to-compute probability density through optimization. In the case of Bayesian modelling we approximate $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$. The method first chooses a family of functions H and then attempts to find a member of this family $q_\theta(\mathbf{w})$ parametrized by θ, that is as close to the probability density being approximated as possible. When choosing H we want to choose a family which is flexible enough to capture the approximated distribution closely but also take into account that computational cost increases with greater complexity of H [2]. Closeness between the distributions is measured using the Kullback-Leibler (KL) divergence [19]:

$$\mathrm{KL}(q_\theta(\mathbf{w}) \parallel p(\mathbf{w}|\mathbf{X}, \mathbf{Y})) = \int q_\theta(\mathbf{w}) \log \frac{q_\theta}{p(\mathbf{w}|\mathbf{X}, \mathbf{Y})} d\mathbf{w} \tag{A.4}$$

KL divergence measures how one probability distribution differs from a second, reference probability distribution. Minimizing it w.r.t parameters θ of the variational distribution $q_\theta(\mathbf{w})$ allows us to approximately restate the predictive distribution as

$$p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{X}, \mathbf{Y}) \approx \int p(\mathbf{y}^*|\mathbf{x}^*, \mathbf{w})q_\theta^*(\mathbf{w}|\mathbf{X}, \mathbf{Y})d\mathbf{w} := q_\theta^*(\mathbf{y}^*|\mathbf{x}^*) \tag{A.5}$$

where q_θ^* is the minimum of the optimization objective [7].

Minimization of KL divergence is equivalent to maximizing the *evidence lower bound* w.r.t θ which is a more commonly used objective:

$$\mathcal{L}_{VI}(\theta) = q_\theta(\mathbf{w}) \log p(\mathbf{Y}|\mathbf{X}, \mathbf{w})d\mathbf{w} - \mathrm{KL}(q_\theta(\mathbf{w}) \parallel p(\mathbf{w}|\mathbf{X}, \mathbf{Y})) \tag{A.6}$$

By maximizing the first term of the equation above we encourage the distribution $q_\theta(\mathbf{w})$ to explain the data well, while the KL divergence term forces it to remain as close as possible to the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ which we are trying to approximate [7].

Instead of optimizing over point estimates as in deep learning the optimization in variational inference is performed over distributions. We can thus preserve the advantages of Bayesian modelling and end up with probabilistic models which also capture uncertainty. We now have the power to approximate posterior distributions of different classes of models that we would be unable to solve analytically for. Certain disadvantages still remain however. In particular, the method has difficulty scaling to large amounts of data and is unable to adapt to some complex models [7].

A.1.3 Bayesian Neural Networks (BNNs)

Bayesian Neural Networks [29] are probabilistic models that replace the deterministic network's point estimates of weight parameters with distributions over these parameters. We do not optimize the network weights directly but instead average over all possible weights by marginalization. To define them in the language of Bayesian modelling, let us denote the random output of the BNN as $\mathbf{y} = \mathbf{f}^{\mathbf{w}}(\mathbf{x})$. In the case of BNNs the model parameters correspond to the network weights in all L layers so we have $\mathbf{w} = (\mathbf{W}_i)_{i=1}^{L}$ [8].

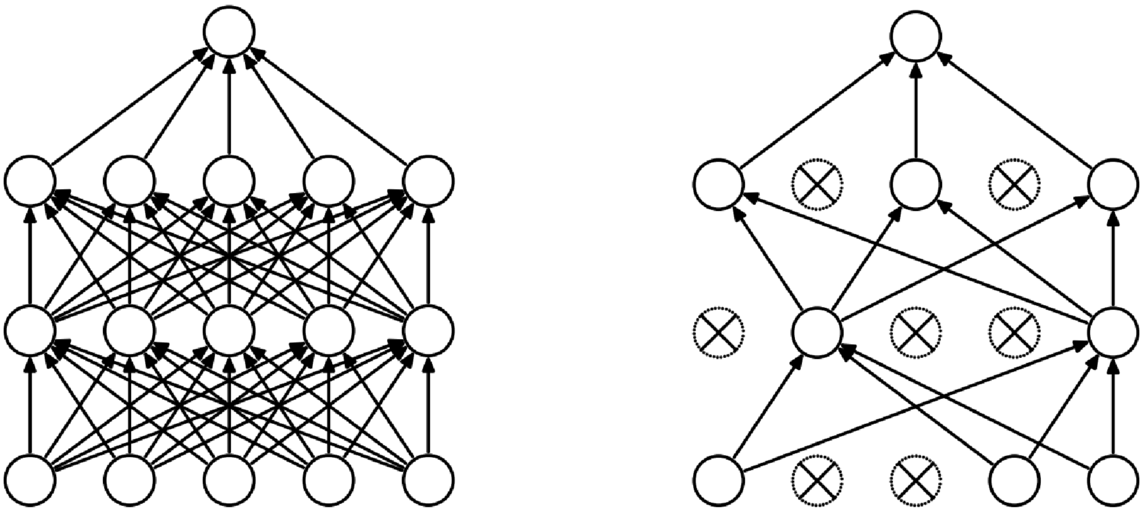
We can then define the model likelihood as $p(\mathbf{y}|\mathbf{f}^{\mathbf{w}}(\mathbf{x})$. When used for regression tasks, the likelihood can be defined as a Gaussian with mean determined by the model output as $p(\mathbf{y}|\mathbf{f}^{\mathbf{w}}(\mathbf{x}) = \mathcal{N}(\mathbf{f}^{\mathbf{w}}, \sigma^2)$ where σ is an observational noise scalar. For classification the model outputs are often squashed through the softmax function and the resulting probabilty vector is then sampled from: $p(\mathbf{y}|\mathbf{f}^{\mathbf{w}}(\mathbf{x}) = \text{Softmax}(\mathbf{f}^{\mathbf{w}})$ [16].

Given the dataset $\mathbf{X}, \mathbf{Y}$, Bayesian inference computes the posterior distribution $p(\mathbf{w}|\mathbf{X}, \mathbf{Y})$ which specifies the plausible model weights having observed the data. Since like most more complex models, BNNs are too complex to perform inference in directly, they need to be approximated by variational inference [16].

A.2 Dropout

This section contains the review of the dropout [43] technique which is used by Gal and Ghahramani [9] as a way of computing approximate Bayesian inference in Bayesian Neural Networks. Their work is described in part 3.2.1 of the main text.

Dropout is a technique originally introduced for addressing the problem of overfitting[2] in deep neural networks. Its central idea is to randomly drop units along with their connections in a particular layer during training with probability p.



(a) Standard Neural Net

(b) Neural Net after dropout

Figure A.1: Dropout network model. **(a)** A standard neural net **(b)** A particular realization of dropout applied to the three layers of the same neural net [43].

Since each time after its application, different units are dropped, it essentially samples from an exponential number of "thinned" networks during training. This should lead to the network learning only the important aspects of the task and data and not fitting the pa-

[2] A model overfits if it adjusts its parameters too closely to the training data and consequently fails to generalize its predictions to unseen data.

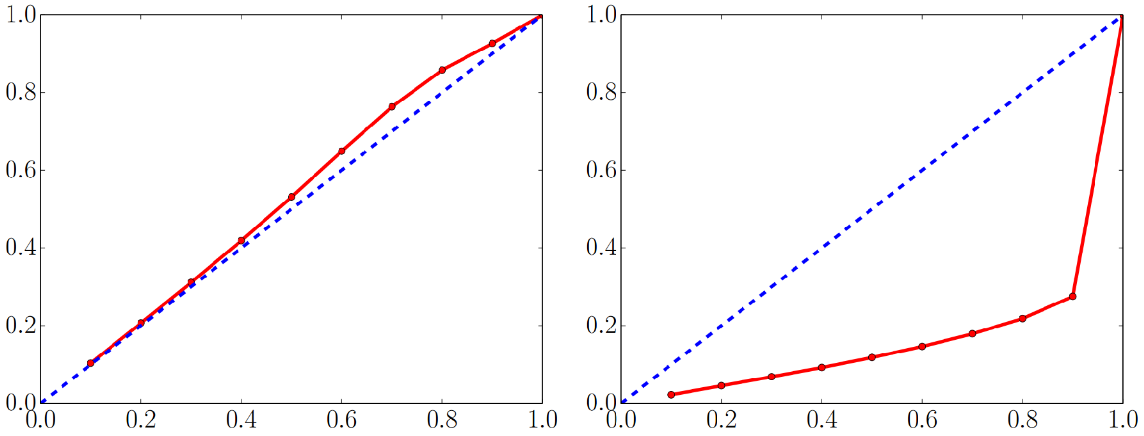


Figure A.2: Model calibration. x-axis corresponds to the expected fraction of an event happening and the y-axis to the actual fraction. The model on the left is much better calibrated than the model on the right. Blue line corresponds to an output of an ideally calibrated model. [20].

rameters to accomodate the complicated relationships in the training data that result from sampling noise and will not be present in the test data. Note that during training, neurons and connections are dropped randomly and applying dropout will result in a different realization (almost) every time.

Since it is unfeasible to take an exponential number of samples during test time, the authors of the dropout paper suggest an approximate method called *weight averaging*. Dropout is no longer applied at test time and all network neurons are present during inference. If the neuron was dropped with probability p during training, then all the weights leading from this neuron are scaled by p during test time. This strategy ensures that for any hidden neuron the expected output (corresponding to the distribution used to drop neurons during training) is equal to the actual output [43].

A.3 Calibration

Calibration is a quantity measuring the discrepancy between subjective forecasts and empirical long-run frequencies of an event [21]. For example, consider a forecaster that sequentially assigns probabilities to events. The forecaster is *well-calibrated* if in the long-term, the proportion of events to which he assigns a probability of 30 percent that actually occur are really 30 percent [4].

Calibration can be assessed using scoring rules. Scoring rules allow us to measure the quality of predictive uncertainty. They assign a numerical score to a predictive distribution based on the prediction and the actual event that materialized and reward probability distributions with better calibrated predictions [10].

Let $\mathbf{x}$ be a vector of input features, y the corresponding label and $p_\theta(y|\mathbf{x})$ a predictive distribution with parameters θ over the labels. We consider scoring rules for which a higher numerical score is better. A scoring rule $S(p_\theta, (y, \mathbf{x}))$ is a function evaluating the quality of the predictive distribution $p_\theta(y|\mathbf{x})$ relative to an event $y|\mathbf{x} \sim q(y|\mathbf{x})$ with $q(y|\mathbf{x})$ denoting the true underlying distribution over the $(y, \mathbf{x})$ tuples. The expected score is then $S(p_\theta, q) = \int q(y, \mathbf{x}) S(p_\theta, (y, \mathbf{x})) dy d\mathbf{x}$. We also define a *proper scoring rule* as one for which $S(p_\theta, q) \leq S(q, q)$ with equality occurring only if $p_\theta(y|\mathbf{x}) = q(y|\mathbf{x})$, for all tuples $(y, \mathbf{x})$ [20].

Proper scoring rules can be used as training criteria for machine learning models and the advantages of this approach are described in part 3.3.1.

Appendix B

Additional Figures

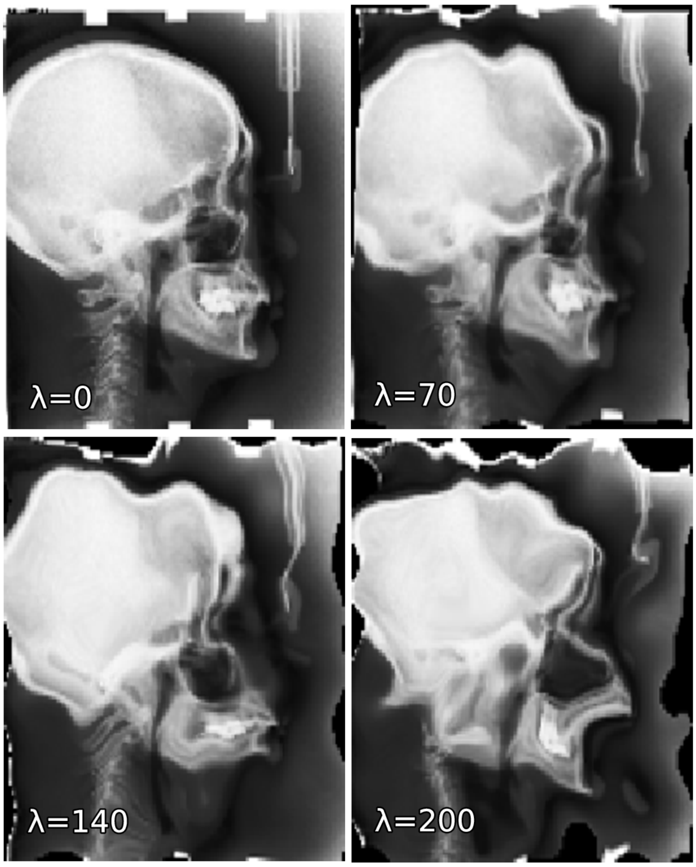


Figure B.1: Elastic distortions applied to a test image. Elastic distortions [41] of increasing magnitude were applied to the entire test set to produce out-of-distribution images. Figure depicts a single test image distorted with different magnitudes λ (the values can be related to figure 6.6).

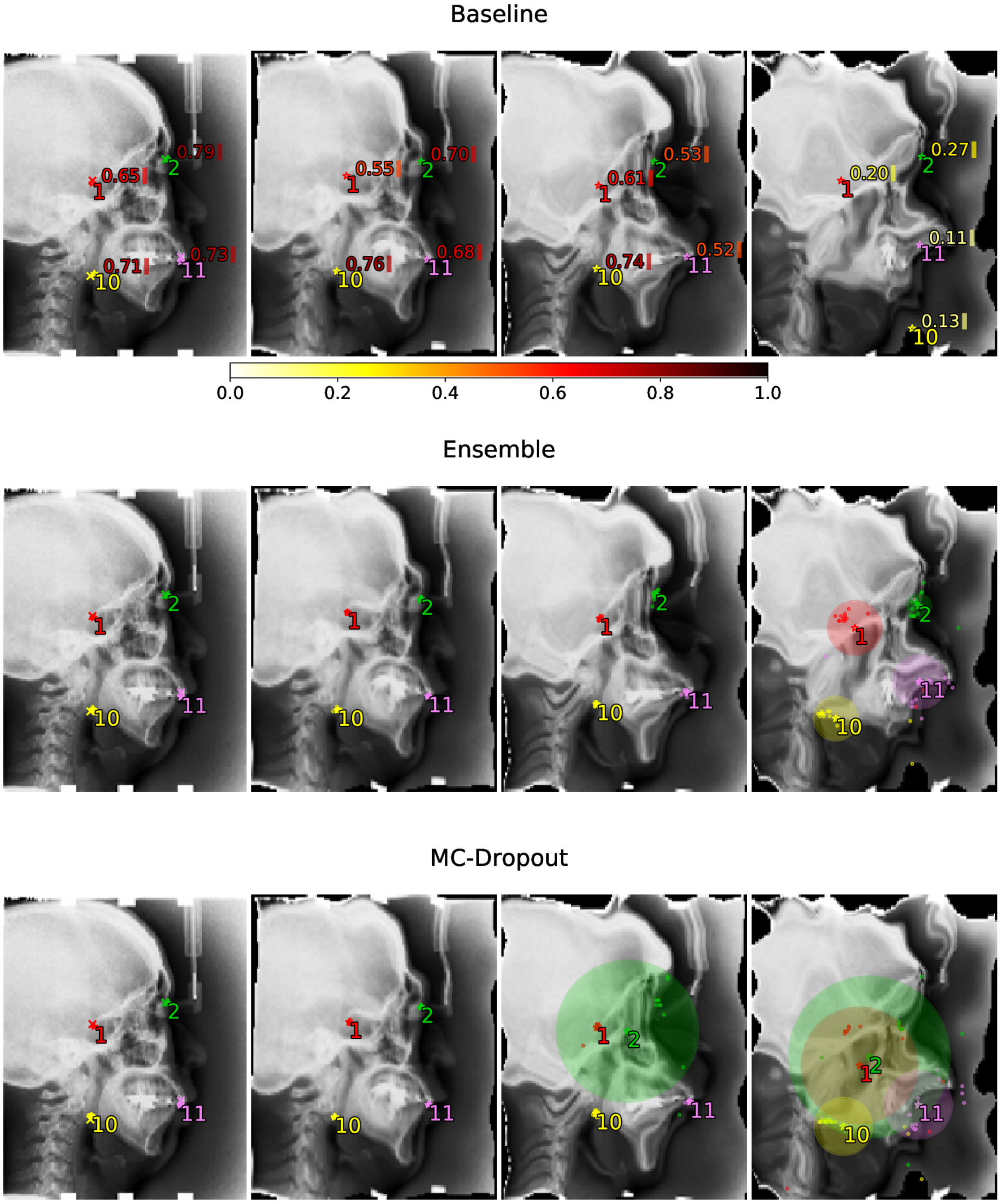


Figure B.2: Fully-trained models' predictions and uncertainty measures on the distorted test sets. Models were evaluated on the same image transformed with elastic distortion magnitudes equal to 0, 70, 140 and 200 increasing from left to right. Predictions and model uncertainty are visualized for landmarks 1, 2, 10 and 11 in each image. Star marks the predicted landmark position and the left-most (undistorted) images also contain the ground truth position marked by a cross. Prediction samples are visualized as dots and prediction variance by a circle for the MC-Dropout and Ensemble models. The maximum heatmap activation (normalized to unit range) is shown for the Baseline model.

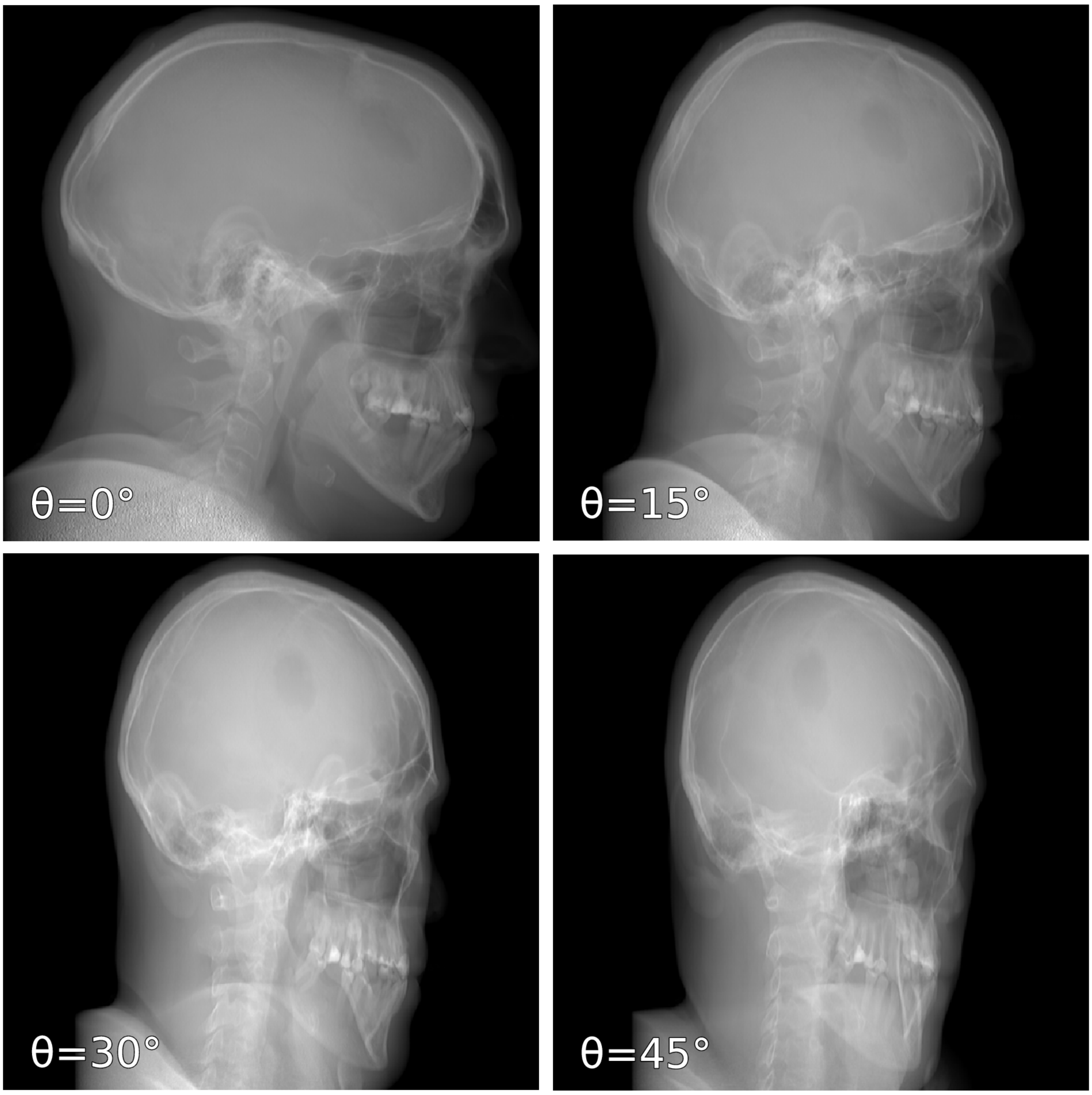


Figure B.3: Lateral projections of a rotated CT scan of a skull. The skull volume was first rotated in the axial plane by θ degrees and then projected onto a single slice in the sagittal plane.

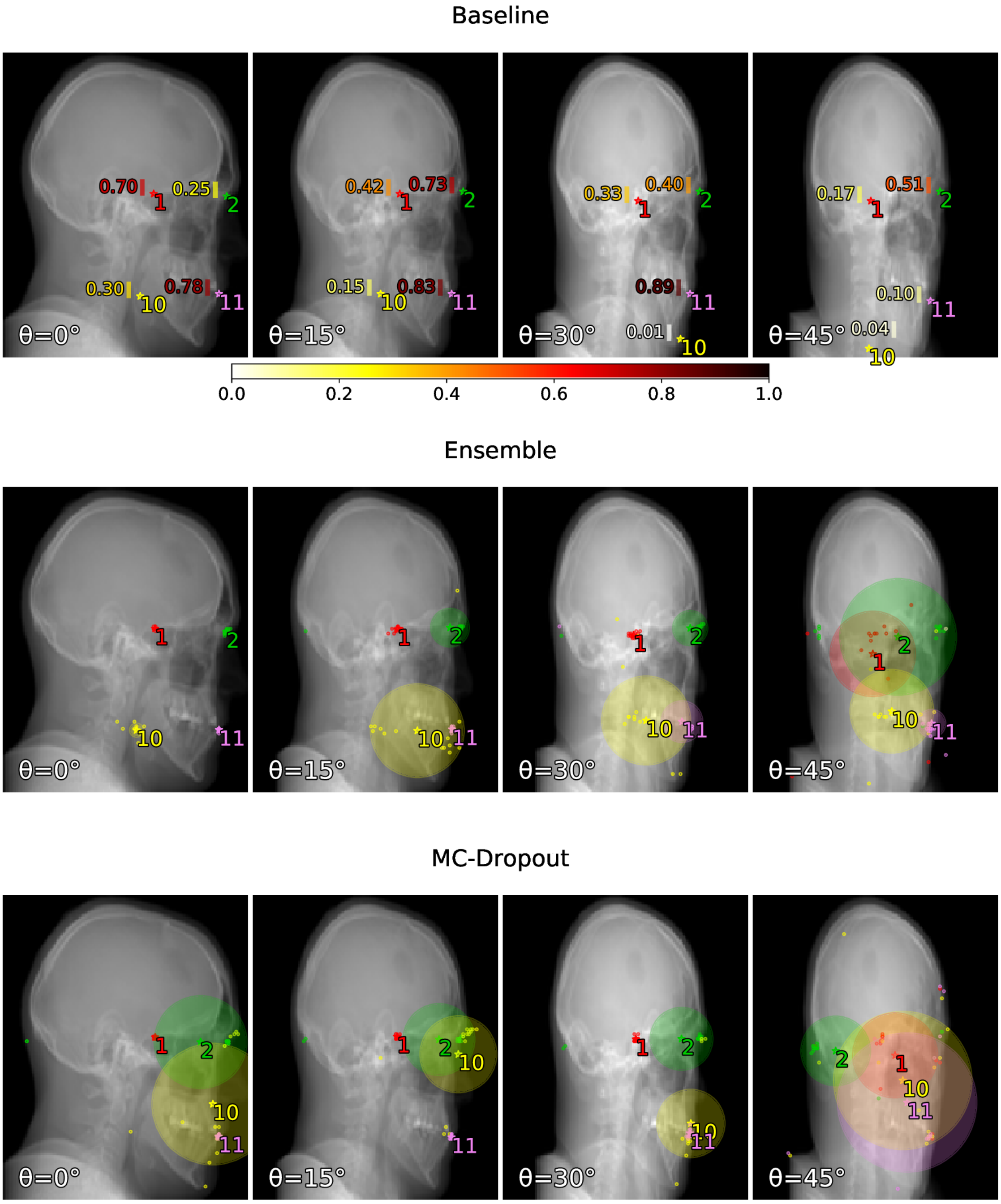


Figure B.4: Fully-trained models' predictions and uncertainty measures on the rotated skull dataset. Predictions and model uncertainty are visualized for landmarks 1, 2, 10 and 11 in each image. Star marks the predicted landmark position while the ground truth is not available. Prediction samples are visualized as dots and prediction variance by a circle for the MC-Dropout and Ensemble models. The maximum heatmap activation (normalized to unit range) is shown for the Baseline model.

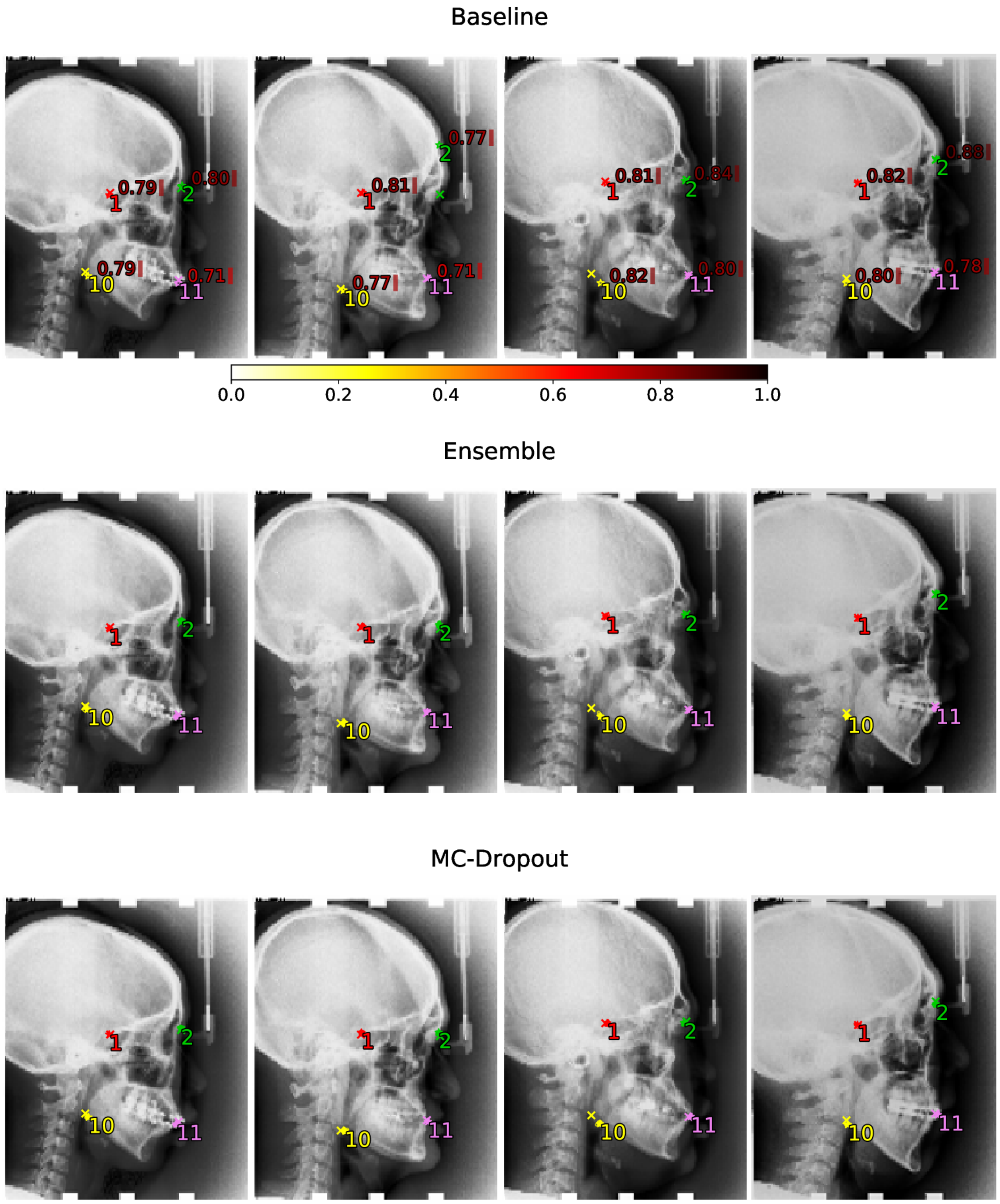


Figure B.5: Fully-trained models' predictions and uncertainty measures on the test set. Predictions and model uncertainty are visualized for landmarks 1, 2, 10 and 11 in each image. Star marks the predicted landmark position and cross the ground truth position. Prediction samples are visualized as dots and prediction variance by a circle for the Ensemble and MC-Dropout models. Both are hardly visible due to low prediction variance. See figure B.6 for a comparison with under-trained models' predictions. The maximum of the heatmap activation (normalized to unit range) is shown for the Baseline model.

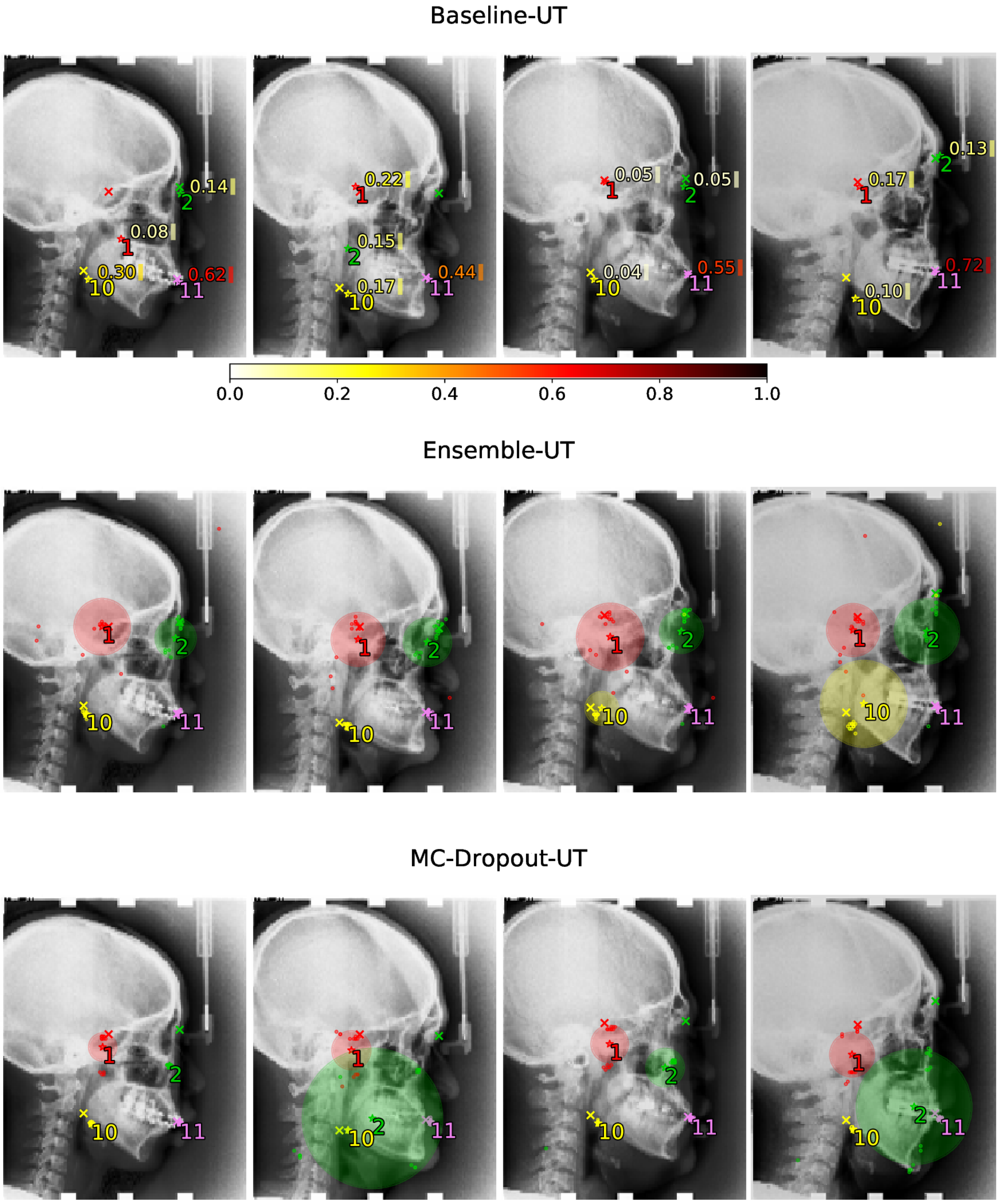


Figure B.6: Under-trained models' predictions and uncertainty measures on the test set. Predictions and model uncertainty are visualized for landmarks 1, 2, 10 and 11 in each image. Star marks the predicted landmark position and cross the ground truth position. Prediction samples are visualized as dots and prediction variance by a circle for the Ensemble and MC-Dropout models. The maximum of the heatmap activation (normalized to unit range) is shown for the Baseline model. See figure B.5 for comparison with the fully-trained models' performance.

Appendix C

CD Content

The CD contains the following directories and files[1]:

`README.md`	Contains code description.
`models`	Source files with CNN architecture.
`utilities`	Source files with utility functions.
`prepare_dataset.ipynb`	Contains steps to download the dataset.
`train.py`	Training script.
`generate_predictions.py`	Script for generating model predictions.
`evaluate.py`	Script for evaluating model performance.
`train_ensemble.sh`	Pre-made script for training the Ensemble model.
`train_mc_dropout.sh`	Pre-made script for training the MC-Dropout model.
`eval_baseline.sh`	Pre-made script for evaluating the Baseline model.
`eval_ensemble.sh`	Pre-made script for evaluating the Ensemble model.
`eval_mc_dropout.sh`	Pre-made script for evaluating the MC-Dropout model.
`thesis`	Thesis text.
`thesis_source`	Thesis source code made in LaTeX.

[1] This content is also available at https://bitbucket.org/ddrevicky/deep-learning-uncertainty

Appendix D

Manual

Downloading the Dataset

Follow the `prepare_dataset.ipynb` notebook to download and preprocess the data.

Model Training and Evaluation

Use the following scripts to evaluate the performance of the models on the landmark localization task.

Ensemble and Baseline

To train 15 independent Baseline models to form an Ensemble as described in the thesis run `train_ensemble.sh`. To generate predictions for all ensemble members and then evaluate the full Ensemble run `eval_ensemble.sh`. Once this is done and the predictions are generated you can also evaluate a single Baseline member by running `eval_baseline.sh`.

MC-Dropout

Run `train_mc_dropout.sh` to train an MC-Dropout model as described in the thesis. To first generate predictions on the test set using 15 samples and then evaluate the performance of the model run `eval_mc_dropout.sh`.

Deep Learning Model Uncertainty in Medical Image Analysis

Dušan Drevický Advisor: Oldřich Kodym

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ABSTRACT

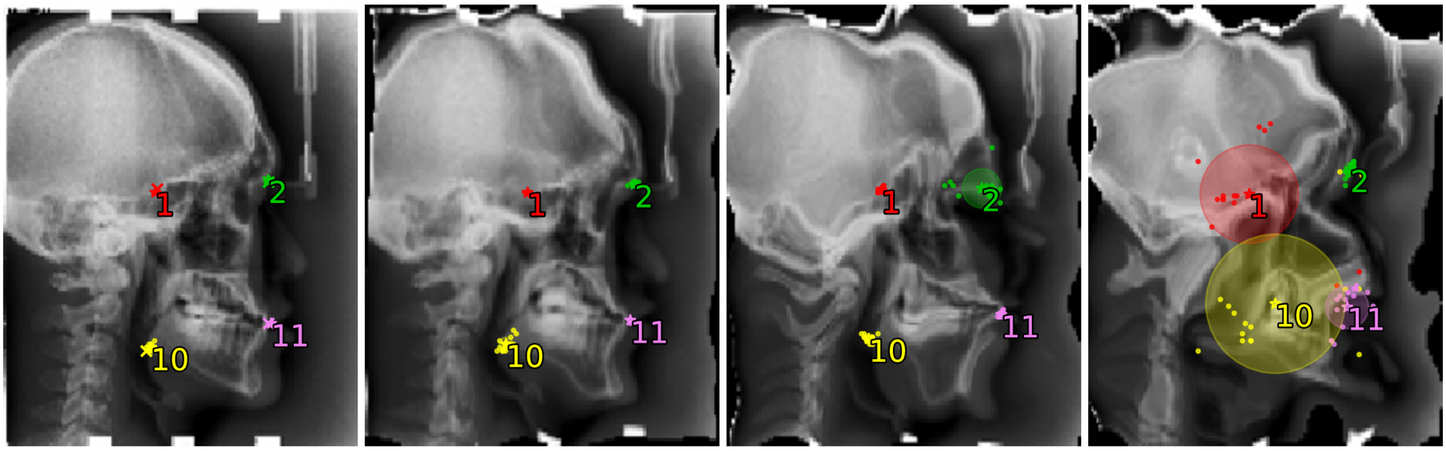
While **Deep Learning** models achieve state of the art results in image analysis, they do not provide reliable information about the certainty of their predictions by default. This shortcoming is especially important in **medicine** where **mistakes are costly** and we would prefer to know whether the model predictions can be trusted or not for a given patient.

This work implements **three uncertainty measures** on cephalometric landmark localization task formulated as heatmap regression. In addition to outputting the landmark positions, the models also estimate the **uncertainty of their predictions**. This allows us to **detect data** for which the models would perform poorly and handle them appropriately.

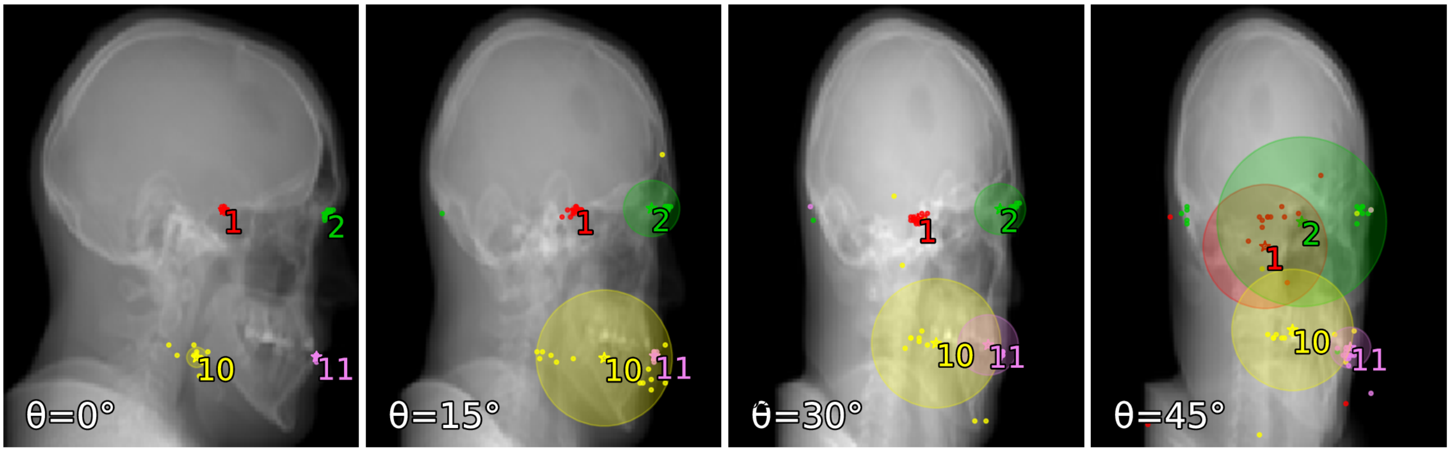
METHODS

- **Maximum activation value** in the heatmap predicted for each landmark by a CNN without dropout.
- **The Monte Carlo sample prediction variance** of a CNN with multiple dropout layers. Dropout is also applied at test time.
- **The ensemble prediction variance** of several CNNs without dropout. **The best performing method.**

RESULTS



Ensemble prediction variance on an increasingly **distorted cephalogram.** Cross marks ground truth landmark positions, star the mean predicted position and dots correspond to prediction samples. Uncertainty increases together with distortion strength.



Ensemble prediction variance on an increasingly **laterally rotated cephalogram**. Ground truth is not available. Uncertainty increases together with rotation magnitude.