Uncertainty Quantification and Estimation on Medical Imaging Classification Tasks

Sidi Yang

A Thesis
In the Department
of
Computer Science and Software Engineering

Presented in Partial Fulfillment of the Requirements
For the Degree of
Master of Computer Science at
Concordia University
Montreal, Quebec, Canada

December 2020

Concordia University
School of Graduate Studies

This is to certify that the thesis prepared

By: **Sidi Yang**

Entitled: **Uncertainty Quantification and Estimation on Medical Imaging Classification Tasks**

and submitted in partial fulfillment of the requirements for the degree of

Master of Computer Science

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the final Examining Committee:

_____ Chair
Dr. Adam Krzyzak

_____ Examiner
Dr. Adam Krzyzak

_____ Examiner
Dr. Tristan Glatard

_____ Supervisor
Dr. Thomas Fevens

Approved _____
Dr. Lata Narayanan, Chair of Department

November 2020 _____
Dr. Mourad Debbabi,
Dean Faculty of Engineering and Computer Science

Abstract

Uncertainty Quantification and Estimation on Medical Imaging Classification Tasks

Sidi Yang

This thesis presents an uncertainty quantification (UQ) system on medical classification imaging tasks and its practical use. Deep Neural Networks have shown tremendous success in numerous AI-related fields, for example, object detection, recognition, and health care. However, despite Deep Neural Networks exhibiting remarkable performance, we usually can not guarantee the modelling predictions to be absolutely correct. Therefore, estimation and quantification of uncertainty have become an essential parameter in Deep Learning practical applications, especially in medical imaging. Measuring uncertainty can help with better decision making, early diagnosis, and a variety of tasks.

In this thesis, we explore uncertainty quantification (UQ) approaches and propose an uncertainty estimation system for general medical imaging classification tasks. In experiments, we apply the UQ system for three medical imaging databases, including All-IDB2 (an acute lymphoblastic leukemia database), SARS-CoV2 (a coronavirus disease 2019 database) and BreaKHis (a breast cancer histopathological imaging database). Besides, we discuss how to apply UQ methods to obtain more information on the database and its modelling. We can capture the samples with the uncertainty values and predict the most uncertain category. We also discover that we can receive more accurate results than initial modelling results by removing a percentage of data with higher uncertainty results. In summary, we find great potential for UQ research on complex medical classification tasks and consider it to become probably one of the future's essential research directions.

Acknowledgments

First and foremost, I would like to express my sincere appreciation to my supervisor, Dr. Thomas Fevens. With his support and encouragement, I can finish this thesis for the Master's Degree. I would like to thank the Examining Committee for reviewing the previous edition thesis and their constructive comments.

Also, I am grateful for Surgical Innovation Program. It has been an excellent and professional experience at the hospital and working with a cross-disciplinary team about the innovation process for creating surgical devices. I would like to thank the NSERC-CREATE program and Concordia University for the financial support for my study.

Furthermore, I would extend my thanks to my colleagues at Imagia. The internship this summer at Imagia has been an incredible experience for my future career path. I also express my thanks to my lab colleagues at Deep Learning for Medical and Visual Processing (DL-MVP) lab. I have learned such a lot from our lab meetings in the last two years. Further thanks to my friends for supporting me for all ups and downs during this research.

Finally, I would like to dedicate this thesis to my family. With your unconditional love, I can always get encouraged and move forward. I will always love you.

2020 has been a tough year for us all. I wish everyone safe and the best.

Contents

List of Figures viii

List of Tables x

1 Introduction 1

1.1 Motivation . . . 1
1.2 Contributions . . . 2
1.3 Outline . . . 3

2 Background 4

2.1 Neural Networks . . . 4
2.1.1 Convolutional Neural Networks . . . 5
2.1.2 Bayesian Neural Networks . . . 8
2.1.2.1 Bayesian Probabilistic Modeling . . . 8
2.1.2.2 Sampling . . . 9
2.1.2.3 Variational Inference . . . 9
2.2 Uncertainty Quantification . . . 10
2.2.1 Source of Uncertainty . . . 11
2.2.2 Uncertainty Categories . . . 12
2.3 Uncertainty Quantification Approaches . . . 12
2.3.1 Monte-Carlo Dropout Approach . . . 12
2.3.1.1 Monte-Carlo Sampling for Probability . . . 13
2.3.1.2 Monte-Carlo Dropout as a Bayesian Approximation in Neural Networks . . . 14
2.3.2 Deep Ensembles Approach . . . 15
2.3.3 Ensemble MC Dropout Approach . . . 17

2.4 Uncertainty Computation Measures 18
2.4.1 Prediction Variance 18
2.4.2 Predictive Entropy 19

3 Methodology and Experiment Setup 21
3.1 Experiment Workflow 21
3.2 Databases 23
3.2.1 ALL-IDB Database 23
3.2.2 SARS-CoV2 CT-scan Database 24
3.2.3 BreaKHis Database 26
3.3 Transfer Learning 27
3.4 Inception Network Models 29
3.4.1 Inception-V1 29
3.4.2 Inception-V3 31
3.4.3 ResNet 32
3.4.3.1 Residual Block 33
3.4.3.2 ResNet152V2 34
3.4.4 Inception-ResNet-V2 34

4 Experiments and Results 36
4.1 Data Argumentation and Preparation 37
4.2 Uncertainty Quantification on ALL-IDB2 38
4.2.1 Capturing Uncertainty via Three UQ Approaches 38
4.2.2 Evaluation and Association between Uncertainty and Accurateness 41
4.3 Uncertainty Quantification on SARS-CoV2 41
4.3.1 Finding the Most Uncertain Images in SARS-CoV2 43
4.3.2 Uncertainty Distribution Quantification on Data Retention . . 46
4.4 Uncertainty Quantification on BreaKHis 49
4.4.1 Classification of Eight Categories on BreaKHis 49
4.4.2 Uncertainty Exploration at Four Magnifications 52
4.4.3 Finding the Most Uncertain Category 56

5 Conclusion and Future Work 61

Bibliography **63**

List of Figures

1 An example of max-pooling layer [100] 7
2 The Convolutional Neural Networks (CNNs) architecture 7
3 Standard NNs vs. NNs with MC dropout architecture 15
4 Experimental workflow 22
5 ALL-IDB2 example images 24
6 SARS-CoV-2 example images 25
7 An example of BreakHis slide image with different magnifications from the same patient 26
8 Eight subtypes example images from BreakHis database at 40X magnification factor 27
9 Transfer Learning architecture 28
10 Inception module in GoogLeNet [86] 30
11 Inception-V1 architecture [86]. 30
12 Grid-size reduction in Inception-V3 module [88] 32
13 Inception-V3 architecture [88] 32
14 Residual Learning: a building block [31] 33
15 Inception modules in Inception-ResNet-V2 model 35
16 Inception-ResNet-V2 architecture compressed view [3]. 35
17 Automatic diagnosis system [23] 46
18 Accuracy vs retained data distribution of three models via three UQ approaches on validation/test sets. The caption of each plot is named as {*model, dataset*}. 48
19 Example of images at four magnifications with predicted results and true labels presented 51

20 Accuracy vs retained data distribution on **validation** sets at four magnifications. The captions are named in the form of {*Magnification, Dataset*}. Each graph contains six plots presenting the combination of three UQ approaches and two models by different colors. Each plot is named by the way of {*UQ Approach, Model*}, which can be found at the corner of each graph. 54
21 Accuracy vs retained data distribution on **test** sets at four magnifications. The captions are named in the form of {*Magnification, Dataset*}. Each graph contains six plots presenting the combination of three UQ approaches and two models by different colors. Each plot is named by the way of {*UQ Approach, Model*}, which can be found at the corner of each graph. 55
22 Removing percentage based on uncertainty predictions (left) vs. uncertainty of true labels (right) at magnification 40X and 100X 59
23 Removing percentage based on uncertainty predictions (left) vs. uncertainty of true labels (right) at magnification 200X and 400X . . . 60

List of Tables

1 SARS-CoV-2 CT-scan database description 25
2 BreaKHis database description 27
3 ALL-IDB2 database train/validation/test composition description . . 38
4 Hyper-parameter configuration in ALL-IDB2 experiment 39
5 Uncertainty mean value of samples in the validation/test sets by three UQ approaches on ALL-IDB2 database 41
6 Uncertainty mean value (*Unc*) and sample size (*Num*) of wrong-predicted (*Wrong-pred*) and right-predicted (*Right-pred*) samples on validation/test sets by MC dropout, deep ensemble and ensemble MC dropout UQ methods on ResNet152V2 (*ResNet*) , Inception-V3 (*IncepV3*) and Inception-ResNet-V2 (*IncepRes*) models. 42
7 SARS-CoV2 train/validation/test sets composition description 43
8 Ten most uncertain samples in SARS-CoV2 database 44
9 Confusion matrices of the original test set (top) and new test set with ten most uncertain images removed (bottom) 45
10 BreaKHis detailed description table with eight sub-category and four magnifications 50
11 Accuracy on validation/test sets by MC dropout (*MCD*), Deep ensembles (*Deep Ens*) and Ensemble MC dropout (*Ensemble MCD*) UQ approaches with Inception-V3 (*IncepV3*) and Inception-ResNet-V2 (*IncepRes*) models at four magnifications (*40X, 100X, 200X and 400X*) 52
12 Comparison of the accuracy performance of the eight-class classification on BreaKHis with the previous work. For the first six references, 100% of testing data was used. 56

13 The evaluation metrics computed from best result of our work at each magnification factor and compare with the previous work. For the first three references, 100% of testing data was used. 57

Chapter 1

Introduction

1.1 Motivation

Deep Learning (DL) methods have made a revolution in many fields during the past decade, including bioinformatics [64] [90], natural language processing [18] [106], and autonomous driving [66] [93]. Despite their success, DL methods still have restrictions due to inadequate predictions on some tasks, such as overconfidence on out-of-distribution data.

A simple example is to feed dogs and cats' images to a classifier for training; however, another category of birds' images also mixes into the test set. The model will still return predictions, but with low accuracy, since it does not learn birds' features from the training process. Our task is to let the classifier tell us that it does not know enough information about the birds' images. As the classifier only learns cats and dogs' knowledge, it cannot recognize birds with good results. In a word, we need a way to know what our model does or does not know.

Therefore, reliability and safety have become a crucial concentration lately. Overconfidence can sometimes cause unintended and harmful behaviours [5]. Uncertainty Quantification (UQ) has been addressed as a vital role in numerous tasks, especially with high-safety requirements. In particular, here are various uncertainty quantification applications in practical:

- Decision making and object detection stability detection in medical diagnosis [7] [24]
- Forecasting, as a realistic example, prediction on impending financial expenses

of customers in digital payments [13]

- Guide exploration (concentrate on uncertain fields) to assume deterministic dynamics [37]
- Removal from noisy data based on uncertainty distribution to improve the existing system
- Applications on AI safety to prevent problems such as oversight, reward hacking and other negative side effects [5]
- Automating mechanical technology security concerns [69]

Our goal is to improve and optimize existing systems by estimating and quantifying uncertainty. In this research, we consider accuracy as one of the essential evaluation components as same as in practical. Accordingly, we combined Transfer Learning methods and uncertainty quantification approaches to promise reasonable efficient predictions and feature extractions on medical classification tasks.

1.2 Contributions

This thesis focuses on exploring uncertainty on classification tasks on medical imaging to optimize automatic diagnosis systems. The experiments consist of **two binary classification** tasks on an acute lymphoblastic leukemia database [53], a coronavirus disease 2019 (COVID-19) database [81] and **a multi-classification** study on a breast cancer histopathology imaging database [82].

We first explore and compare the uncertainty quantification (UQ) measures, including Monte-Carlo dropout, deep ensembles and ensemble MC dropout on three databases. Then we propose three UQ applications which would be beneficial for optimization and development in reality, which includes:

- finding the most uncertain samples;
- predicting the most uncertain category;
- detecting statistics inside a database and improving modelling using UQ results in a multi-classification problem.

1.3 Outline

In structure, this thesis is organized into five parts:

- Chapter 1 is the introduction of this thesis. We discuss our motivation and list the thesis outline.
- Chapter 2 concentrates on background description. We introduce Neural Networks, Uncertain Quantification (UQ), and its corresponding measurement approaches and computation methods.
- Chapter 3 describes the detailed experimental design and implementation modelling methodology. We also introduce three databases utilized in the experiments in this chapter.
- Chapter 4 presents procedures, results, and analysis of three experiments in specific.
- Chapter 5 concludes the thesis research and discusses the possibilities of future work.

Chapter 2

Background

This chapter introduces the thesis's general background, including Neural Networks (NNs) in section 2.1, Uncertainty Quantification (UQ) in section 2.2, Uncertainty Quantification approaches in section 2.3 and uncertainty computation methods in section 2.4.

2.1 Neural Networks

Before embarking on quantifying the uncertainty, it is crucial to understand what the Neural Networks (NNs) are. A neural network is an artificial neural network, which consists of artificial neurons or nodes [36]. Marvin Minsky and Seymour Papert proposed the first research of machine learning associated with the neural network in 1969 [65]. In the 1970s, Kunihiko Fukushima developed the neocognitron, the original convolutional neural networks architecture [91]. In 1989, Yann LeCun et al. combined neural networks with back-propagation theories and implemented the handwritten digits recognition [20]. Over recent years, many techniques including support vector machines [19], gradient-based learning [56], AlexNet [50] have been proposed, which led the evolution of Deep Learning. The neural network research stagnated and achieved remarkable performance and applications. For instance, Convolutional NNs (CNNs) [55] and Residual NNs (RNNs) [31] achieve outstanding performance in various areas, especially in recent years.

This section addresses two types of NNs used in our uncertainty estimation experiments — Convolutional Neural Networks (CNNs) and Bayesian Neural Networks

(BNNs).

2.1.1 Convolutional Neural Networks

Deep Learning (DL), as a branch of Machine learning, has shown outstanding performance on a wide variety of applications, for example, automatic speech recognition [34], image restoration [57] [60] and medical image analysis [59] [78] etc. We can find a definition of Deep Learning from the book *"Deep Learning"* published by MIT in 2016 [28] :

> *"Deep learning is a particular kind of machine learning that achieves great power and flexibility by learning to represent the world as a nested hierarchy of concepts, with each concept defined in relation to simpler concepts, and more abstract representations computed in terms of less abstract ones."*

The neural network is one of the most famous architectures which assists the development of DL. One of the most popular neural network architectures is the Convolutional Neural Networks (CNNs) in Computer Vision [104], which was introduced by Y. LeCun and Y. Bengio [55] in 1995. Over the last few years, CNN has become a typical architecture with tremendous contributions in various applications in real-life, including satellite imaging recognition [2], classifying hand-written characters [17], and Natural Language Processing (NLP) field [18].

CNN is composed of a series of neurons carrying learnable weights and biases. Each neuron receives inputs, performs a dot product and alternatively follows with a non-linearity [85]. It consists of three types of basic layers:

- The **convolution** layer. Convolutional layers extract features in the CNN architecture. In mathematics, **convolution** is defined as an *operation* on two functions f and g to produce a new function which describes how one function is modified from the other one. The computing process and corresponding results are **convolution**.

  The convolutional layer in a CNN architecture works the same way. It is an *operation* to use a matrix (named as *filter* or *kernel*) on another matrix. Specifically, the *operation* is to multiply the values of the image matrix and kernel

matrix within the corresponding *cells*. The *cells* generate with the span of the filter matrix. We apply this operation to all the cells covered in the image matrix. Then the last step of the convolution layer is to add them together into another matrix as the output.

Meanwhile, some critical operations during convolution are:

- The **padding** function adds another row and column at the sides of the input image with value 0, which can also name as **Zero Padding**. These "0"s will not add any extra information but can account for the previously less-accounted values into the output. The padding function can help with the model learning completely from the training samples and reducing bias.
- The **striding** function applies during convolution. Instead of shifting the filter one-by-one at a time, the striding function can increase the size to 2 or 3 each time. The striding function can efficiently reduce the calculation cost as well as the size of the output matrix.
- The **ReLU** (Rectified Linear Unit) activation function. Its mathematical definition is:

$$f(x) = \begin{cases} 0, & \text{for } x < 0 \\ x, & \text{for } x \geq 0 \end{cases}$$

  The ReLU activation functions are applied to the output matrix after the convolution operations. The primary advantages of using ReLU are to reduce likelihood of vanishing gradient problem [35] which might stop the NNs for further training and speed up the computation time.

- The **pooling** layer. Pooling layers usually extract a particular value from a set of values. This value could either be the maximum of all the values indicating the **max-pooling**, or the average value referring to the **average-pooling**. The pooling function helps with reducing the size of the output matrix.

  As an example of max-pooling in Figure 1, we take the maximum value of each 2×2 part from the input (left). The output is the maximum value of each block, as illustrated on the right side. We calculate the average value in the average-pooling function instead of picking up the max value in the max-pooling function. For example, the red block result after average-pooling would be $(12 + 20 + 8 + 12)/4 = 13$.

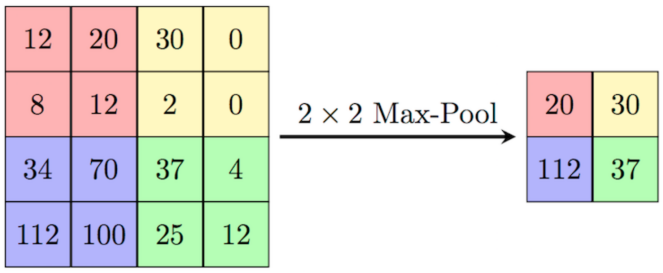


Figure 1: An example of max-pooling layer [100]

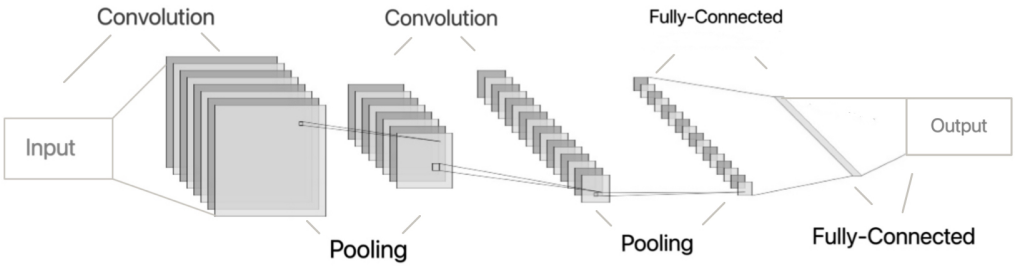


Figure 2: The Convolutional Neural Networks (CNNs) architecture

- The **Fully-Connected (FC)** layer. The FC layer is usually the last layer in a CNN architecture. It has full connections to all the activations from previous layer. In classification tasks, the last FC layer is referred as **Softmax** layer to decide the final result of each input image.

Combining all the layers and functions discussed so far, we present an example of a simple common use CNN of a classification task can be summarized as shown in Figure 2.

The CNN is one of the widely demonstrated architectures in DL. It is the foundation of various state-of-art networks currently proposed. One of the most effective CNN-based architecture in current classification tasks is Inception Networks, which is investigated based on CNNs. Therefore, we choose Inception Networks as a starting point in our experiments. In section 3.3, we discuss Inception Networks architecture and relevant models in detail.

2.1.2 Bayesian Neural Networks

Deep Learning methods (e.g., CNNs) have become powerful tools for researchers and engineers in recent years. However, we still can not always guarantee perfect results. Namely, NNs are overconfident with their predictions occasionally. Thus, our motivation is to explore uncertainty distribution science within NNs. Besides, we often need probabilistic predictions to reflect uncertainty or model confidence in classification tasks. Bayesian Neural Networks (BNNs) fit into this role as a standard methodology of quantifying uncertainty.

A BNN consists of a probabilistic model and NNs. In some deep NNs, we already use probability distributions. For instance, the *Dropout* layer in NNs as stochastic computations, generative modelling with a distribution over x. The objective of BNNs implementation is to combine the strengths of stochastic modelling and neural networks.

2.1.2.1 Bayesian Probabilistic Modeling

First of all, let us review how to evaluate standard NNs modelling, we have a model labelling as $p(y|x, w)$ with y as output, x as input and w as weights. Given a dataset $D = \{(x_n, y_n)\}_{n=1}^{N}$, N is the size of this dataset and n is its index. Most NNs are estimated by the Maximum Likelihood Estimation (MLE) as:

$$-min_w \Sigma_{n=1}^{N} \log p(y|x, w)$$

where $-\log p(y|x, w)$ is the negative log-likelihood. It leads to a squared error in regression or a cross-entropy error in classification. To avoid overfitting, the *regularization* function is summed up into the MLE as a prior $p(w)$:

$$-min_w \Sigma_{n=1}^{N} \log p(y|x, w) - \log p(w)$$

Another useful measure is to apply L2 regularization [68], which can result in weight decay. Overall, the results of both of them are stochastic gradient estimations. The points estimates of weights here are obtained, not as random variables. It will not provide any knowledge underlying uncertainty.

In contrast, Bayesian probabilistic modeling estimates the posterior distribution over weights to capture uncertainty:

$$p(w|D) = \frac{p(D|w)p(w)}{p(D)} = \frac{p(D|w)p(w)}{\int p(D|w)p(w)dw}$$

where w is the weight, D as data, $P(w)$ as the prior probability based on w, and $p(D|w)$ as the likelyhood of the observations. $p(w|D)$ is the posterior probability which can be used for uncertainty estimation. In other words, we measure uncertainty over parameters of a model.

Although posterior probability offers the right perspective of viewing probabilistic science inside modelling to estimate uncertainty, it is challenging to compute the output numerically of multi-dimensional parameter spaces. Therefore, some efficient methods are combined to solve this problem when estimating the uncertainty. The two most common approximation methods in practical are *Sampling* and *Variational Inference*.

2.1.2.2 Sampling

The sampling technique is often used in probability problems, and more broadly, in Deep Learning problems. For many probabilistic problems, especially when inference is intractable, we have to resort them to some form of approximation [9]. It is the regular use of sampling. In other words, samples are drawn randomly from a probability distribution, then approximate to the desired quantity. The Markov Chain Monte Carlo family [74] of algorithms is a general class technique for sampling from random variables. The Monte-Carlo dropout approach for estimating uncertainty is built based on this technique. We will discuss Monte-Carlo Sampling in detail in section 2.3.

2.1.2.3 Variational Inference

The primary concept of the Variational Inference (VI) is to translate the inference of the posterior into an optimization problem, which could either be minimization or maximization. Mathematically, the posterior $p(w|D)$ can be approximated into a $q_\theta(w) = q_\theta(w|D)$ with a known distribution form and parameters θ. As an example, q could be a Gaussian distribution. Then we will have the mean and variance based on this distribution. We can estimate parameter means and variances instead of maximizing a posterior estimation. Hence, we need to minimize the Kullback-Leibler (KL) divergence [43] as:

$$KL(q_\theta(w)|p(w|D)) = E_{q_\theta(w)}\left[\log \frac{q_\theta(w)}{p(w|D)}\right]$$

which can be expanded to:

$$\overbrace{KL(q_\theta(w)|p(w|D))}^{\text{KL divergence}} = -E_{q_\theta(w)}[\log p(w|D)] + E_{q_\theta(w)}[\log q_\theta(w)] + \underbrace{\log(p(D))}_{\text{Constant}}$$

With adding $-E_{q_\theta(w)}[\log p(D|w)]$ to both sides, it can be simplified to:

$$-E_{q_\theta(w)}[\log p(y|x,w)] + KL(q_\theta(w)|p(w))$$

where $-E_{q_\theta(w)}[\log p(D|w)]$ can become $-E_{q_\theta(w)}[\log p(y|x,w)]$ since we assume the data is distributed identically and independently. To minimize this equation, we applied our assumed Gaussian distribution to $q_\theta(w)$ and $p(w)$, the final equation becomes as:

$$-KL(q_\theta(w^{(l)})|p(w^{(l)})) + \Sigma_{l=1}^{L} \log p(y|x,w^{(l)})$$

where l is the index of L. This is the mathematics concept underlying VI approximation, which is the core concept of *MC dropout* uncertainty estimation approach.

After exploring these two approximation methods for estimating posterior probability, let us look into our main topic of this thesis — Uncertainty Quantification.

2.2 Uncertainty Quantification

Deep Learning has permeated our daily lives and several research fields in recent years, especially in medical image processing tasks, such as segmentation, classification, and object detection. Numerous techniques have been developed every day and proven state-of-art achievements. For exploration into the unknown black-box work, uncertainty quantification has become a trendy research field lately in the DL field.

Uncertainty Quantification (UQ) is a science of quantitative depiction and uncertainty reduction in computational and practical applications. When the possibility of wrong decisions decreases by more training data and more complicated models, it would be infeasible to cover all possible cases but usually concentrating on the most crucial problems. Therefore, a model should show its trustworthiness by encountering an unknown circumstance where it can infer knowledge and emphasizes the outcome [52]. This process is uncertain. In general words, we know there exist unknown factors a model does not know and does not learn from training. We are curious to learn what exactly a model does not know.

2.2.1 Source of Uncertainty

There are various sources of uncertainty in mathematical models and practical applications. One way to classify uncertainty sources is [46]:

- **Parameter uncertainty** is the uncertainty from the model parameters, which are the inputs to the experiments but unknown to the experimentalists. As an illustration in a statistical model, we look for approximation methods to infer the results since the real results can not be inferred directly. This operation will generate the parameter uncertainty.

- **Model inadequacy** refers to structural uncertainty due to the lack of knowledge of physics underlying the problem. As we know, the model will lead to particular predictions during training; thus, the prediction results are not equivalent to actual values. The discrepancy is defined as the model inadequacy.

- **Residual variability** is the variance yield with the repeated conditions. The prediction usually generates by the inputs along with the specified conditions. In reality, if these conditions repeat, this procedure will not always generate the same value. This type of variation is identified as the residual variability.

- **Parametric variability** comes from the input parameters unspecified or uncontrolled, and variability of inputs. Parametric variability will generate extra uncertainty in implementation.

- **Observation error** is from the variability of actual experimental observations. Observation error is unavoidable in practical, which adds further uncertainty to the residual variability at the same time. Typically, they are not easy to separate during experiments.

- **Code uncertainty** comes from numerical approximation and errors during the training process. Because effective models are usually complex, code uncertainty is inevitable, especially in DL.

There are general sources of uncertainty in practical applications. We will look into how to categorize uncertainty in deep learning problems in the next section.

2.2.2 Uncertainty Categories

As the UQ research develops, uncertainty can be modelled into two primary categories prominently in medical applications [48]: aleatoric uncertainty and epistemic uncertainty.

- **Epistemic uncertainty**, on the other hand, is the uncertainty inside the model [45]. Hence, epistemic uncertainty can also be considered as model uncertainty. The epistemic uncertainty will decrease with enough training samples. It is resulting from the limitation of knowledge and data of the system. Some common causes could be that the model ignores certain effects; measurement is not accurate.

- **Aleatoric uncertainty**, which is also known as statistical uncertainty or data uncertainty. *Alea* here refers to "dice" in Latin. The aleatoric uncertainty is the uncertainty that exists inside the dataset. It captures noise inherent in the observations [45]—randomness, which arises from the distribution of data. Unlike epistemic uncertainty, aleatoric uncertainty will not decrease with more data provided. It is highly dependent on bias and distribution inside the input data but not the size of training samples.

After exploring what uncertainty, its sources, and the categories are, we will investigate how we quantify uncertainty in modelling in the next section.

2.3 Uncertainty Quantification Approaches

This section will introduce the approaches for estimating uncertainty in our experiments, which include: Monte-Carlo dropout approach in section 2.3.1, deep ensembles approach in 2.3.2, and ensemble Monte-Carlo dropout approach in 2.3.3.

2.3.1 Monte-Carlo Dropout Approach

As we discussed from the mathematical concept in section 2.1.2, the standard DL techniques in medical imaging tasks, such as segmentation and classification, do not capture uncertainty. We want to learn how to estimate and quantify uncertainty in DL models. To solve this problem, Gal and Ghahramani [25] proposed an approach named Monte-Carlo dropout (MC dropout) for estimating uncertainty in 2016.

2.3.1.1 Monte-Carlo Sampling for Probability

In statistics, Monte-Carlo (MC) methods refer to a broad class of computational algorithms depending on repeated **random sampling** to receive numerical results. MC approaches primarily apply in three-class problems [51]: generating draws, optimization, and numerical integration from a probability distribution.

MC methods have several developments in computer science-related fields, for instance:

- **Artificial Intelligence**. Monte-Carlo Tree Search [15] is an effective algorithm for the decision process. It is a framework for Game AI, which aims to find the most promising next moves in a game. The Monte-Carlo Tree Search algorithm broadly applies in practical games, such as classic board games, modern board games, and video games.

- **Applied Statistics**. There are several applications associated with the MC Method in Applied Statistics Science. For example, In our experiments of the Monte-Carlo dropout approach, it can provide random samples from the posterior distribution in the Bayesian inference. Additionally, MC methods can be applied to compare competing statistics between small samples in real data state [33]. Moreover, MC methods can implement hypothesis tests, which are more effective than permutation tests and more accurate than asymptotic distributions.

- **Computer Graphics and Computational Biology**. In the computer graphics field, an algorithm named Ray Tracing [44], also known as Path Tracing, is an MC method which renders three-dimensional-scene images. It integrates over illuminance at a single point on a surface of an object. Global illumination transformation to reality would be faithful by this algorithm. In the computational biology area, the MC method applies in various areas; as an example, Bayesian inference in phylogeny [38], a study in proteins [71] or membranes [63].

In a word, Monte-Carlo Sampling Methods are used in various fields when the problem correlates to probabilistic interpretation. It can provide the foundation for a wide range of practical issues. Some DL methods, including ensemble networks, hyperparameter tuning, and resampling, are related to MC methods.

2.3.1.2 Monte-Carlo Dropout as a Bayesian Approximation in Neural Networks

Before we look into the Monte-Carlo dropout approach, let us review what *dropout* is. Dropout is a technique proposed by Srivastava et al. in 2014 [84]. It is a generic regularization technique to avoid overfitting. The key concept of the *dropout* function is to randomly drop units and their connections from the NNs during training. It exists in most types of NNs as a convenient technique of optimization, including CNNs, Recurrent Neural Networks (RNNs).

The core idea of the MC dropout approach is to propose how to interpret a regular dropout before every weight layer to be used as a Bayesian approximation of the Gaussian process [73], which refers to a probabilistic model. We can consider this probabilistic model as a BNN with the probabilities distributions as the output. The key component in this model to implement MC dropout approach is the *dropout* layer [25]. There are various reasons to make modifications on the dropout layer as the UQ tool, which includes:

- generalization and convenience of dropout in NNs
- forcing the network to learn redundant and independent features
- saving computation cost.

From the mathematical and statistic side, a dropout interpretation can be approximated to a Bayesian process. With a Bayesian model, we can obtain uncertainty values from the posterior distributions of weights. The next step is to approximate this posterior distribution of weights by VI we introduced in section 2.1.2.3. Adapted the variables from section 2.1 to apply in MC dropout approach, (x, y) as (input,output), $p(w|x, y)$ as observed data, w as weights, q_θ as the distribution to approximate to the observed data, we need to minimize the equation:

$$L_\theta = -E_{q_\theta(w)}[\log p(y|x, w)] + KL(q_\theta(w)|p(w|D))$$

The best θ can make q_θ a good approximation for the posterior is given by the θ which maximizes the above equation as the lower bound. The next step is to sample $\hat{w}$ from q, then the expectation $E_{q_\theta(w)}$ can be replaced by $\log p(y|x, \hat{w})$. Then we have:

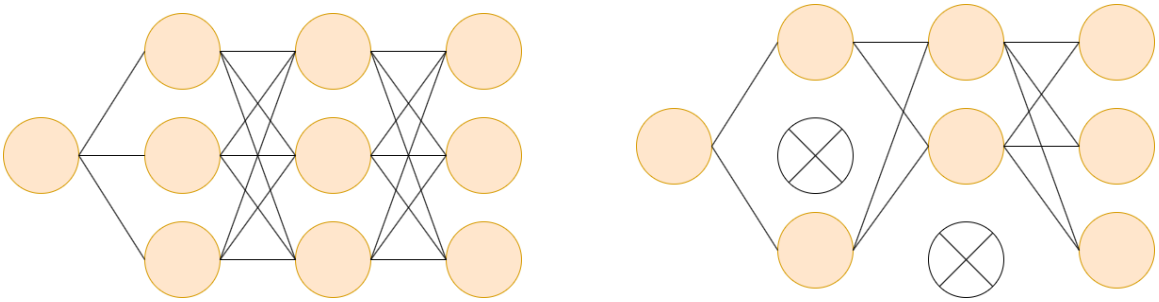
$$\hat{L}_\theta = \log p(y|x, \hat{w}) - KL(q_\theta(w)|p(w|D))$$

$\hat{L}_\theta$ is known as an unbiased estimator of L_θ in the applied Bayesian statistics. Then repeat this process as [25]:

1. sample $\hat{w}$ to $q_\theta(w)$
2. apply one step minimization regarding θ as

$$\hat{L}_\theta = \log p(y|x, \hat{w}) - KL(q_\theta(w)|p(w|D))$$

Another highlight from this paper is that the authors define q_θ as the product of the weight matrix. A diagonal matrix is built with Bernoulli values on the diagonal based on the weight matrix. The sampling technique applies to the elements inside this Bernoulli diagonal matrix. The process is exactly as same as randomly dropping out units on the dropout layer. It is the mathematics foundation of the MC dropout approach.



(a) (b)

Figure 3: (a) Standard NNs (b) Applying two neurons dropout

To be specific in our classification tasks, we receive each image's probability results to decide its category by *Softmax* layer. Due to the random characterization of BNN modelling, we can obtain multiple different predictions for each image in multiple MC sampling procedures. Eventually, we collect the prediction results and apply computation measures to receive each image's uncertainty values. We will describe the whole overflow in detail in section 3.1.

2.3.2 Deep Ensembles Approach

The deep ensembles method was proposed by Google Deep Mind in 2017 [54]. Deep neural networks have shown impressive performance in improving accuracy on recent

medical imaging work. In the last section, we introduce the MC dropout method, which is positively related to BNNs. Unfortunately, the convenient MC dropout approach occasionally comes with a high cost, e.g. Markov Chain Monte-Carlo is expensive. As an alternative method to quantify uncertainty, the deep ensembles method also yields remarkable predictive uncertainty estimations on experiments.

There are three critical steps in the deep ensembles method:

- Proper scoring rules. Setting up a proper scoring rule is the basis of measuring predictive uncertainty [26]. We assign a numerical score to the prediction $p_\theta(y|x)$. In Deep NNs related work, many common loss functions are proper rules. For example, in multi-classification tasks, the cross-entropy loss function is a proper scoring rule. A proper rule is a crucial step in deep ensembles, considering it can measure if a model knows what it should know. As an example, after we obtain a network from a training set, we apply the model to a different type of dataset. The predictive uncertainty results we receive from the new dataset should be much higher compared with what we received from the original training set.

- Use adversarial training to smooth the predictive distributions. Adversarial training is proposed by Szegedy [87] and improved by Goodfellow et al. [27]. This is an optional step in the deep ensembles training process. The adversarial training in deep ensembles intends to smooth predictive distributions.

- Train an *ensemble*. There are various ensembles procedures and algorithms in practical applications, such as decision trees and random forests. Generally, ensembles methods can be classified as two categories: *boosting-based methods* and *randomization-based methods*. Boosting-based methods indicate sequential instances in an ensemble. On the other side, randomization-based methods refer to ensemble instances in parallel without connections, which do not require correlation with a sequence. We concentrate on randomization-based methods in the deep ensembles approach to deal with parallel uncertainty output distributions. In most Deep NNs tasks, the models usually perform better with more data. One of the authors' essential findings is: when the parameters of NNs were randomly initialized, it is sufficient to archive better performance in practice [54].

Mathematically, we donate a training database S with N i.i.d samples, $S = \{x_n, y_n\}_{n=1}^{N}$. Let M donate number of NNs in ensemble with $\{\theta\}_{m=1}^{M}$ and l as the scoring rule. Then the training criterion would be $l(\theta, x, y)$. The overall process of the deep ensembles training procedure summarizes in Algorithm 1 [54]:

Algorithm 1 Pseudocode of deep ensembles method training procedure

1. Let each neural network parameterize a distribution over the outputs, i.e., $p_\theta(y|x)$. Use a proper scoring rule as the training criterion $l(\theta, x, y)$. Recommended default values are M = 5 and ϵ = 1% of the input range of the corresponding dimension. (*e.g., 2.55 if input range is [0,255]*).
2. Initialize $\theta_1, \theta_2, ..., \theta_M$ randomly
3. for m = 1: M do

   Sample data point n_m randomly for each net

   (Optional) Generate adversarial example using $x'_{nm} = x_{nm} + \epsilon \text{sign}(\bigtriangledown_{xnm} l(\theta_m, x_{nm}, y_{nm}))$

   Minimize $l(\theta_m, x_{nm}, y_{nm})) + l(\theta_m, x'_{nm}, y_{nm}))$ w.r.t. θ_m

In our experiments, we used $M = 5$ due to computation time limitation, and it can yield uncertainty quality improvement with reasonable computational cost [54].

2.3.3 Ensemble MC Dropout Approach

After exploring two UQ techniques, we found that the MC dropout technique applies to one BNN and the deep ensembles technique on M normal NNs. They all show outstanding performance in our classification experiments, where the results are in chapter 4. We are curious to know if combining these two methods would reach another state-of-art performance. Hence, we look into the ensemble MC dropout approach.

The ensemble MC dropout approach [80] is a combination of the MC dropout method and the deep ensembles method. Instead of using normal NNs in deep ensembles, we replace them with Bayesian NNs and apply the MC dropout technique. Hence, with each model's probability results in the ensembles, we will receive probabilistic predictions of Bayesian NNs with M models.

In this section, we introduce three UQ methods, including the MC dropout method, deep ensembles method and ensemble MC dropout method. The next step is to apply

computation metrics to receive the uncertainty results in the convenient numerical form of these three UQ methods for further comparison. Hence in the next section, we will look into uncertainty computation methods.

2.4 Uncertainty Computation Measures

This section introduces the computation methods based on the UQ approaches from the last section. We apply two main categories of computation methods in the experiments: predictive variance and predictive entropy [67].

2.4.1 Prediction Variance

In probability theory and statistics, variance usually measures how a set of values is distributed. The computation method is the expectation value of a random variable X is the squared deviation from its mean value, μ. Mathematically, we can calculate the variance as:

$$Var(X) = E[(X - \mu)^2]$$

In general, a model passes the output of probabilities between $[0.0, 1.0]$ through the *Softmax* function in classification tasks. Then we compare prediction results with the ground truth by the evaluation measures (e.g., cross-entropy). The **prediction variance** computation method is based on these probability results. Regarding which UQ approach is chosen, we can either apply predictive variance or MC sample variance.

- **Predictive variance** can be learned directly from the output generated by the training process, which can be used for the deep ensembles method. We can collect the prediction results of M models and compute the variance with the predicted probabilities. The variance results are predictive variances.

- **MC sample variance** is the computation technique of uncertainty inferred from the variance of MC samples of the prediction results. It could be either used in a single Bayesian NN without the MC dropout method or ensemble MC dropout method. We collect the MC sample size output of each image and then choose the predictive values which decide the final class from each output to compute the variance. It is the MC sample variance for a BNN with the MC

dropout method. Hence, one extra step for ensemble MC dropout is to compute variance based on M models.

Variance is an effective way for computing uncertainty values. However, as declared in two variance methods, we focus on the predictive values of the chosen class, which decides the classification results but ignores the prediction values of other classes. Variance methods are better choices in the segmentation task. In Bayesian segmentation modelling, each pixel assigns to one probability value. In such a situation, we can obtain detailed information with variance methods.

We introduce another computation method — predictive entropy in the following section. It can cover more information in classification tasks, which applies to our experiments.

2.4.2 Predictive Entropy

Entropy is a measurement method of the randomness in an information system, especially in Machine Learning (ML) problems. Measuring entropy is a convenient way to collect chaos. It appears everywhere in the ML area, including decision tree construction, training NNs. For example, in classification tasks, *cross-entropy* is a common loss function to measure the difference between probability distributions for a set of events or a given variable. The basic mathematical formula of computing entropy is:

$$Entropy = -\sum_{i=1}^{c} p_i \log p_i$$

where c is a constant, i is the index of c, and p_i is the probability distribution. Compared with regular rigid matrices we apply in normal NNs, entropy would be a better choice on Bayesian NNs due to its stochastic process. Specifically, we approximate the entropy for input across each MC sample. Afterwards, we collect all the entropy of its MC samples to obtain the final uncertainty value for each image. The computation of the ensemble MC dropout method is similar; we only need to collect the uncertainty values from M models and compute its average uncertainty value. We can compute the entropy of the prediction outputs with M models for the deep ensembles method and compute its average value as the final uncertainty value.

We can either collect uncertainty for each category or find the most uncertain samples based on entropy values for further analysis. In general, a higher entropy

value refers to more information, which means higher uncertainty. Our experiments will present these assumptions thoroughly in chapter 4.

Chapter 3

Methodology and Experiment Setup

In chapter 3, we describe our methodology and experimental preparation. We present our experiment system by a flowchart in section 3.1. The datasets exploited description in this thesis are in section 3.2. Then in sections 3.3 and 3.4, we explain Transfer Learning and Inception Networks applications, which are the foundations for further uncertainty estimation in our experiments.

3.1 Experiment Workflow

In this section, we describe our experimental workflow in a flowchart. Figure 4 displays the whole workflow of our experiments. In brief, it can be summarized as two main stages to obtain uncertainty from a modelling classifier on a dataset:

1. **Classification Modeling Task**, which includes image pre-processing and training models to obtain a model with predictive probability outputs. We introduce the modelling and the related background in sections 3.3 and 3.4.

2. **Uncertainty Quantification (UQ) and Estimation**, which consists of:

   (a) Apply UQ measures at test time, including MC dropout, deep ensembles and ensemble MC dropout approaches in our experiments.

   (b) Apply UQ estimation and computation methods, e.g. predictive variance or entropy to numerical uncertainty results.

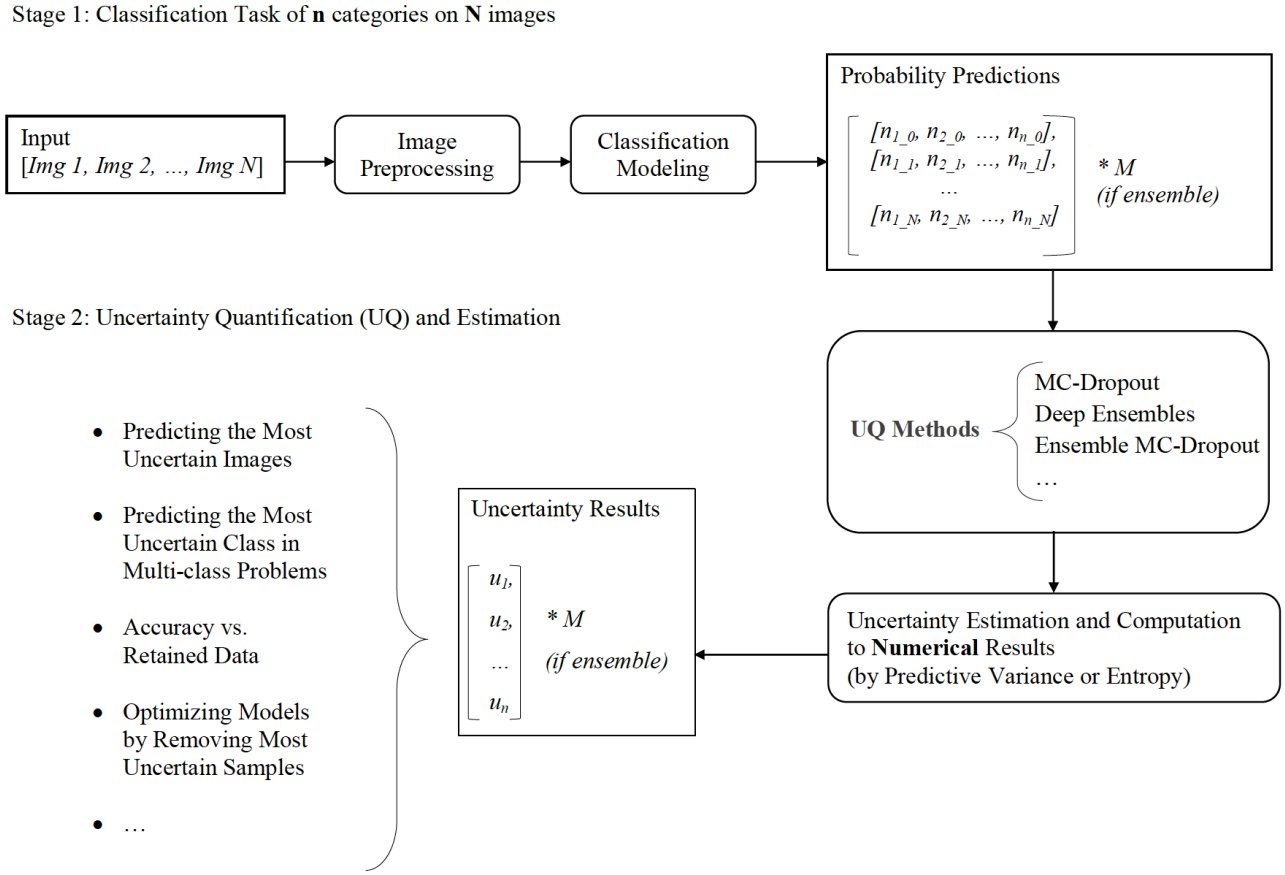


Figure 4: Experimental workflow

We also concentrate on how to evaluate and optimize our modelling work after we gain uncertain results. There are various applications we can accomplish, such as:

- Predicting the most uncertain images as shown in section 4.3.1
- Plotting accuracy vs. retained data based on uncertainty results to evaluate a database and its modelling as stated in section 4.3.2
- Optimizing models by removing the most uncertain samples as indicated in section 4.4.2
- Predicting the most uncertain class in a multi-class problem as illustrated in section 4.4.3

3.2 Databases

There are three databases we explored in our experiments. Firstly, we estimate uncertainty on the ALL-IDB2 [53] database to classify healthy cells and lymphoblasts cells. Secondly, to make our work and results more accurate, we apply our uncertainty methods to a newly released and larger dataset — SARS-CoV2 CT-scan database [81]. We classify CT-Scans to assist with early diagnosis of Covid-19. Lastly, we quantify uncertainty on the BreaKHis database [82], which comprises eight-category breast tumour images.

In the following sections, we would briefly introduce these three databases and their combinations. The experimental results would be shown and analyzed in chapter 4.

3.2.1 ALL-IDB Database

Acute Lymphocytic Leukemia (ALL) also refers to Acute Lymphoblastic Leukemia as a type of hematic disease resulting in fatally in a short period. Early diagnosis of this cancer plays a critical role for patients, especially for young children and adults over 50 [53]. Therefore, we are interested in the classification between standard and lymphocyte imaging on ALL datasets.

Acute Lymphoblastic Leukemia Image Database for Image Processing (ALL-IDB) database [53] is a public database of microscopic blood sample images published

by the Department of Information Technology, Università Degli Studi di Milano in 2011. The ALL-IDB database is composed of two subsets: ALL-IDB1 is mostly for segmentation test capability and ALL-IDB2 for classification test systems. We focus on ALL-IDB2 with the binary classification task between healthy cells and probable-lymphoblasts. The ALL-IDB2 collects cropped areas of interest with normal and blast cells. Figure 5 presents a few example images contained in the ALL-IDB2 dataset.

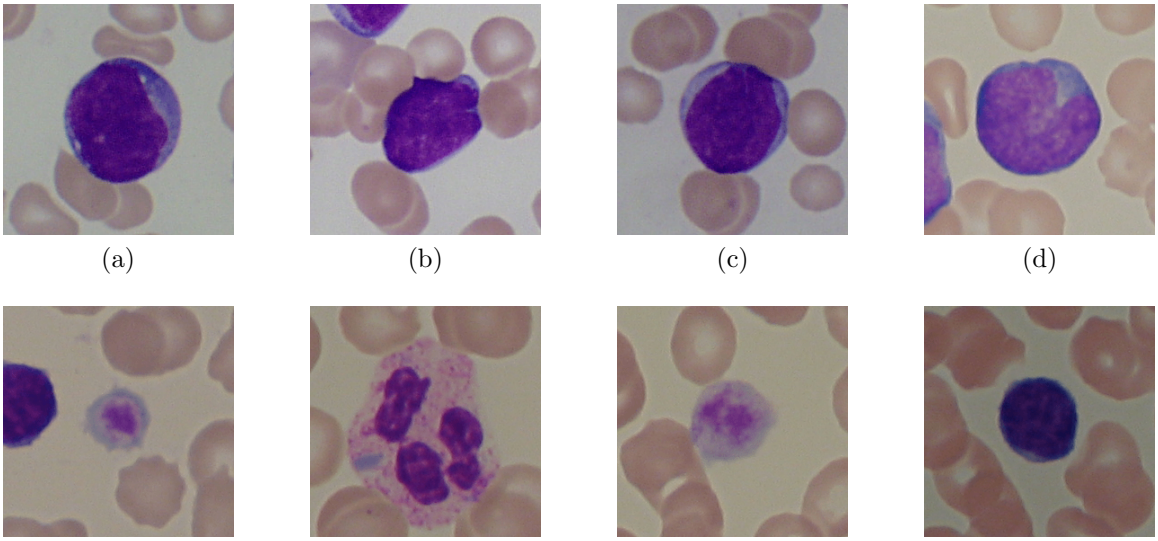


Figure 5: Examples of the images contained in ALL-IDB2: (a), (b), (c), (d) are healthy cells from non-ALL patients, (e), (f), (g), (h) are probable lymphoblasts from ALL patients.

There are 260 images collected in the ALL-IDB2 dataset, composed of 130 normal white blood cell images and 130 cell images with lymphoblasts.

3.2.2 SARS-CoV2 CT-scan Database

In 2020, the coronavirus disease 2019 (COVID-19) has spread and affected across 188 countries and territories. The World Health Organization (WHO) declared the COVID-19 outbreak a Public Health Emergency of International Concern (PHEIC) [101][8] on January 30, 2020 [102]. As of November 5, 2020, 48.2 million cases were reported worldwide, with more than 1.22 million deaths [42].

COVID-19 is an infectious disease caused by severe acute respiratory syndrome coronavirus 2 (SARS-CoV-2) [62]. Recent research has discovered imaging patterns

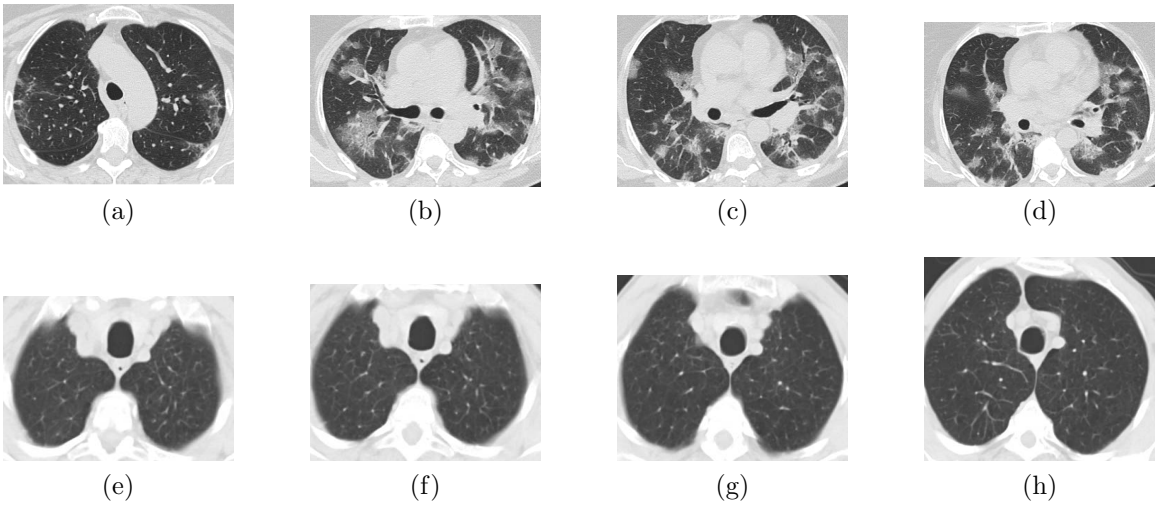


Figure 6: (a), (b), (c), (d) are images from infected by SARS-CoV-2 category; (e), (f), (g), (h) are images from non-infected by SARS-CoV-2 category.

in the Computed Tomography (CT) images of patients with coronavirus disease. Accordingly, CT-scan imaging represents an essential role in COVID-19 diagnosis.

SARS-CoV-2 CT-scan database [81] was published in May, 2020. This database has been collected CT-scan images from real patients from hospitals in Sao Paulo, Brazil. Due to ethical concerns, detailed information on patients has been omitted.

SARS-CoV-2 CT-scan dataset contains 2,482 CT-scan images with two categories: 1252 images for patients infected by SARS-CoV-2 and 1230 CT scans for non-infected by SARS-CoV-2 patients but who presented other pulmonary diseases. Each category was composed of 60 patients with balanced gender distribution. Table 1 represents a detailed data description of the SARS-CoV2 CT-scan dataset.

Patient Gender	Infected	Non-infected	Total
Male	32	28	60
Female	28	32	60
Number of Images	1252	1230	2482

Table 1: SARS-CoV-2 CT-scan database description

Figure 6 illustrates some examples of infected and non-infected by SARS-CoV-2 from the database.

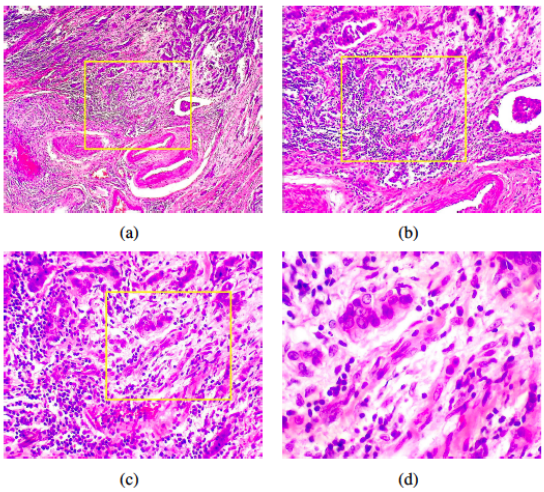


Figure 7: One slide of breast malignant tumor tissue from the same patient with four different magnifications:(a) 40X, (b)100X, (c)200X, (d) 400X. Highlighted rectangle was selected manually by pathologist to be used as the next higher magnification factor [82].

3.2.3 BreaKHis Database

The Breast Cancer Histopathological Image Classification (BreaKHis) database was published by the Department of Informatics, Federal University of Parana and LITIS Lab, the University of Rouen in 2016 [82]. The BreaKHis is composed of 9,109 microscopic biopsy images of 2,480 benign and 5,429 malignant breast tumours. Images were collected from 82 patients from January 2014 to December 2014 referred to P&D lab, Brazil, with a clinical indication of BC participation. There are four different magnifying factors (40X, 100X, 200X, and 400X). Figure 7 shows one slide of breast malignant tumour tissue from the same patient at four different magnifications.

BreaKHis database contains two main categories: Benign(B) and Malignant(M). Each category of breast tumours contains four distinct histological subtypes. Four benign tumours subtype adenosis (BA), fibroadenoma (BF), phyllodes tumour (BPT), and tubular adenoma (BTA), and four malignant tumours are carcinoma (MDC), lobular carcinoma (MLC), mucinous carcinoma (MMC) and papillary carcinoma (MPC). In total, there are eight types of images in the BreaKHis database. All the images are 700X460 pixels, 3-channel RGB (8-bit depth in each channel) with PNG format. Table 2 summarizes the description of BreaKHis.

In current work, binary classification task between Benign and Malignant has

Magnification	Benign	Malignant	Total
40X	625	1,370	1,995
100X	644	1,437	2,081
200X	623	1,390	2,013
400X	588	1,232	1,820
Total	2,480	5,429	7,909
Number of Patients	24	58	82

Table 2: BreaKHis database description

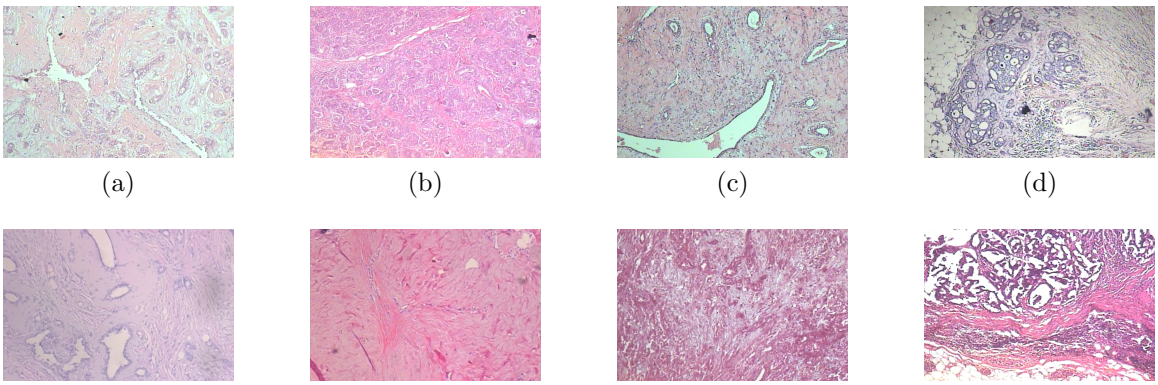


Figure 8: Eight subtypes example images from BreakHis database at 40X magnification factor: (a) BA, (b) BF, (c) BPT, (d) BTA, (e) MDC, (f) MLC, (g) MMC, (h) MPC.

archived excellent performance [83] [97]. Therefore, we focus on classification and uncertainty quantification tasks between eight sub-type images. Figure 8 displays the example images from eight classes at 40X magnification factor.

3.3 Transfer Learning

Currently, Transfer Learning (TL) has increasingly become a popular approach in Deep Learning (DL) field. Lorien Pratt first mentioned in a paper that describes the Discriminability-Based Transfer (DBT) algorithm in 1993. In 2009, the Transfer Learning approach was formally proposed and described in a research publication by L. Torrey and J. Shavlikin [77]. In recent years, a variety of TL methods have been

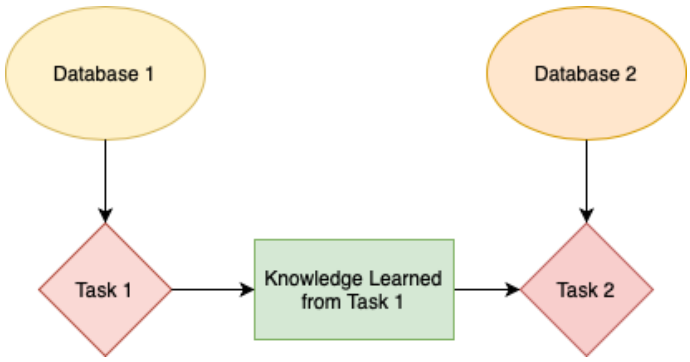


Figure 9: Transfer Learning architecture

applied in various tasks. The widespread of supervised learning in computer vision highlights the importance of TL.

Transfer learning is a research problem in the machine learning field which concentrates on storing knowledge while solving one problem and employing the solution to a different but associated problem [103]. In DL problems, we gained knowledge and extracted features from the existing trained dataset and correlated weights. Then we apply them to another new but similar dataset. For example, we can apply the experience gained from recognizing cats to learn to recognize tigers. Figure 9 represents the elemental overflow of TL.

Transfer Learning allows us not training models from scratch but reuse previous experience as a starting point for a new task to increase productivity. Throughout sharing learned parameters to our new model, we can improve accuracy and optimize computation efficiency simultaneously.

We can also describe the definition of TL in a mathematical way regarding the domain and the task. We assume a domain D. The domain D is composed of a feature space S and a corresponding marginal probability distribution $P(X)$, where $X = \{x_1,...x_n\} \in$ feature space S.

Given the condition of $D = \{S, P(X)\}$, there is another task T consists of a label space Y and an object predictive function f($\cdot$). In other words, $T = \{Y, f(\cdot)\}$. T is learned from the training pairs composed of $\{x_i, y_i\}$, where $x_i \in X$ and $y_i \in Y$. The predictive function $f(\cdot)$ can be used to predict target correlated label $f(x)$ [58], which we refer as Transfer Learning algorithm.

In practice, one of the most popular approaches in recent works is to apply Transfer Learning by loading a pre-trained model. A pre-trained model is a saved model

previously trained on a large dataset, typically in classification tasks, such as ImageNet [21]. The new model can benefit from not starting from scratch but already learned some features from previous work. That would be a good start point to receive better results. Considering we would deal with classification datasets, we decide to start with TL by loading pre-trained weights. Our model would start from loading pre-trained models and modify based on them. We receive better outcomes compared with training a network from scratch.

3.4 Inception Network Models

Inception Networks Model is a landmark in CNNs development, especially in classification tasks. Before Inception models, most NNs usually just stacked more layers and bigger datasets to obtain better performance. Along with the networks became heavy, the computation cost also increased. Another disadvantage of intense neural networks is its high probability of overfitting.

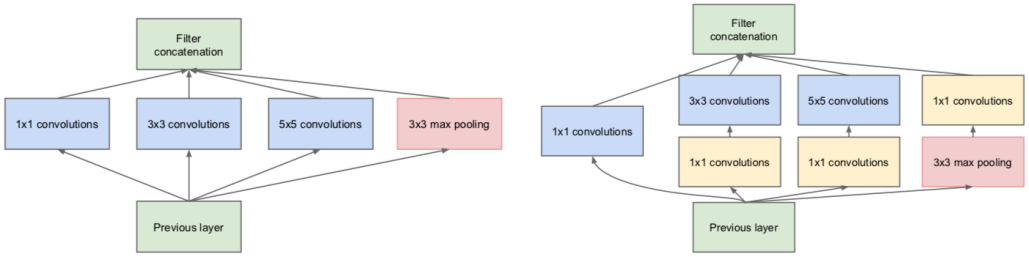
Some famous members from the Inception networks family frequently used in classification are Inception-V1 (GoogLeNet), Inception-V3, and Inception-ResNet. In our experiments, we find Inception-V3 and Inception-ResNet have shown good performance on both the binary and multi-class classification tasks. We consider these two Inception models and another residual architecture model – ResNet and TL approach as appropriate start points for further quantifying uncertainty tasks.

Hence, we will look at the Inception-V1 model to learn how inception module architecture works in section 3.4.1. Then we will present three primary models used in our experiments: Inception-V3 in section 3.4.2, ResNet in section 3.4.3, and Inception-ResNet in section 3.4.4.

3.4.1 Inception-V1

The first version Inception network, known as GoogLeNet (Inception-V1), was proposed by Szegeday, Liu et al. in 2014 [86]. The primary idea of the Inception-V1 model is to make networks "wider" than "deeper." The implemented modifications on modules in NNs are in Figure 10.

Instead of adding layers to make the networks deeper with higher computation cost, the authors filter with multiple sizes on the same level in the *naïve* inception



(a) Inception module, naïve version (b) Inception module with dimension reductions

Figure 10: Inception module in GoogLeNet [86]

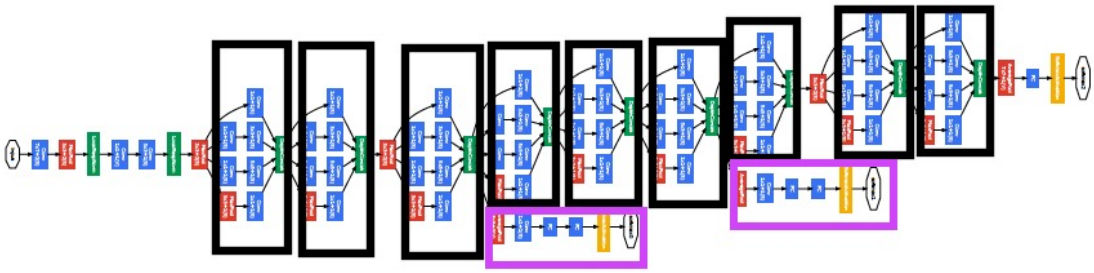


Figure 11: Inception-V1 architecture [86].

module. The different sizes of filters are 1×1, 3×3, and 5×5 convolutions. Meanwhile, max-pooling applies. Then we concatenate the outputs and send them to the next layer.

The second main idea of their proposed architecture is to reduce dimensions. Accordingly, the number of input channels is limited by adding an extra 1×1 convolution before the expensive 3×3 and 5×5 convolutions, as shown in Figure 10 (b). It is the basic module which builds up the Inception-V1 model.

The entire structure of Inception-V1 is shown in Figure 11 (input at left) [86]. It is 27 layers deep (counting pooling layers) with nine basic inception modules (black boxes). A global average pooling layer (red block) and a softmax layer (yellow block) are at the end of the whole structure.

3.4.2 Inception-V3

As an upgrade to the Inception-V1 model, Szegeday, Vanhoucke et al. proposed Inception-V2 and Inception-V3 [88] in 2015.

The two primary updates from Inception-V1 to Inception-V2 are:

- Replacing 5×5 convolution by two 3×3 convolutions in each module. Convolutions with larger size filters are at a higher cost. Especially under the condition of the same number of filters, a 5×5 convolution is 2.78 times computationally expensive than a 3×3 convolution.

- Replacing $n \times n$ by one $n \times 1$ and one $1 \times n$ block. With this two-layer factorization, the computational cost can decrease 33% for the same number of filters.

Inception-V3, as an upgrade version from Inception-V2, contains fewer parameters (7 million) than AlexNet [50] (60 million) and VGGNet [79] ($\approx$ 200 million). It has won the 1st Runner Up in ILSVRC (*Large Scale Visual Recognition Challenge*) in 2015 [75]. One indispensable mechanism in Inception-V3 is to use auxiliary-classifiers as regularizers. In Figure 11, pink boxes parts are named as Auxiliary Classifiers from Inception-V1 [86]. The original purpose of the auxiliary-classifier part is to obtain better convergence and promote more stable learning. In the Inception-V3 model, the authors argued the auxiliary-classifier part as a regularizer. The fact supported is that usually in NNs, the primary classifier performs better if the side branch is batch-normalization [40] or with a dropout layer.

Another modification is that they applied efficient grid-size reduction instead of regular downsizing by max pooling. As an example, in Figure 12, 320 feature maps are received by max pooling, and the other 320 feature maps are convolutions with stride 2. In total, 640 features concatenate. The model becomes less expensive but still efficient by this efficient grid-size reduction method.

To achieve the lowest error rates on ImageNet [21], the authors also factorized 7×7 convolutions, applied RMSProp Optimizer, and added label smoothing regularizing component to loss formula to prevent overfitting in Inception-V3 model. Overall, the entire architecture of Inception-V3 is shown in Figure 13.

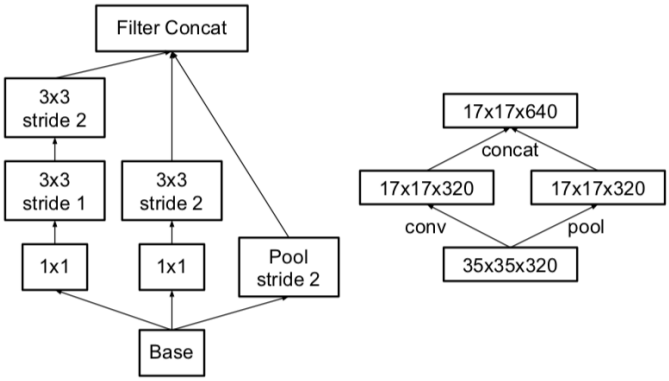


Figure 12: Grid-size reduction in Inception-V3 module [88] (right) and its detailed structure (left)

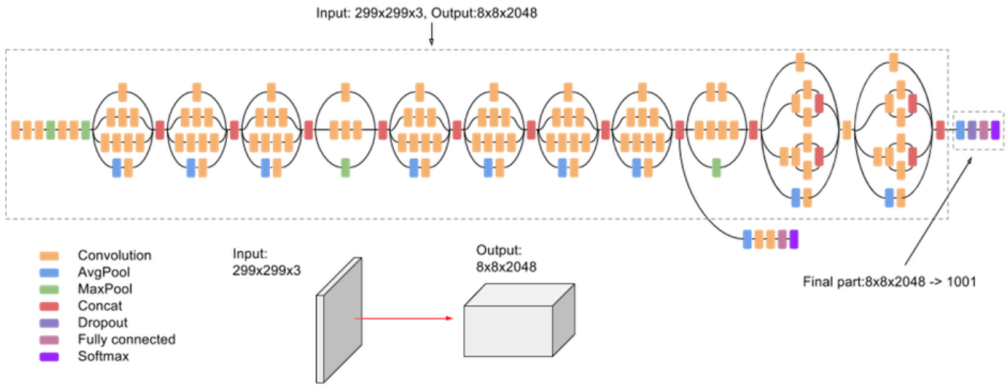


Figure 13: Inception-V3 architecture [88]

.

3.4.3 ResNet

As a prerequisite to digging into Inception-ResNet networks, we would introduce Residual Network (ResNet) in this section. In particular, we present the residual block — the most important structural component in ResNet in section 3.4.3.1. Then in the next subsection, we look into the architecture of ResNet152V2 [32], which we would apply to our experiments.

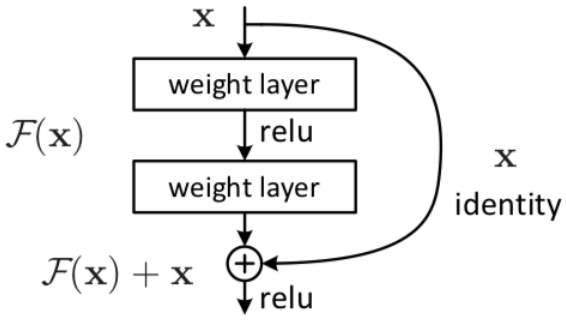


Figure 14: Residual Learning: a building block [31]

3.4.3.1 Residual Block

ResNet was proposed by He, Zhang et al [31] in 2015. The main purpose of ResNet was the same as Inception networks: optimization on NN's architecture instead of adding more layers and data. The most important component of this paper is *Residual Learning*, which applies to Inception-ResNet models. Figure 14 presents an example of a building block with residual learning ideas from the original paper.

We can find the difference in outputs between regular blocks in basic NNs and ResNet. Residual learning collects both the output and previous inputs as the input for the next step. This modified connection is named as *Skip Connection*. Generally speaking, we can consider two layers as one block. Except for the path following two convolutional layers to reach the output, this block's input also directly connects to the output.

In other words, we can assume x as input, w as weights, F as activation function, $F(x, \{w\})$ as the residual mapping to be learned, and final output y. The mathematical presentation of a block with skip connection is:

$$y = x + F(x, \{w\})$$

Considering the case the dimensions of x and F are not equal, we can use a linear projection w_s in last equation to match the dimensions:

$$y = w_s x + F(x, \{w\})$$

which is the mathematics presentation of a residual block. ResNet consists of NNs and residual blocks. In specific, ResNet used a 34-layer plain network inspired by

VGG-19 [79] and modified by adding skip connections. It has shown state-of-art performance on popular public datasets, such as ImageNet [21] and CIFAR [49].

3.4.3.2 ResNet152V2

ResNet152 was proposed in the same paper as Residual Networks [31]. As the name demonstrated, it is a 152-layer deep Residual Networks. It was the most in-depth network on ImageNet in 2015, while it has lower computation cost and better performance on ImageNet compared with VGG networks [79]. The same authors proposed ResNet152V2 in the next year after proposing ResNet. The main difference between the original and V2 is that V2 applies batch normalization before each weight layer.

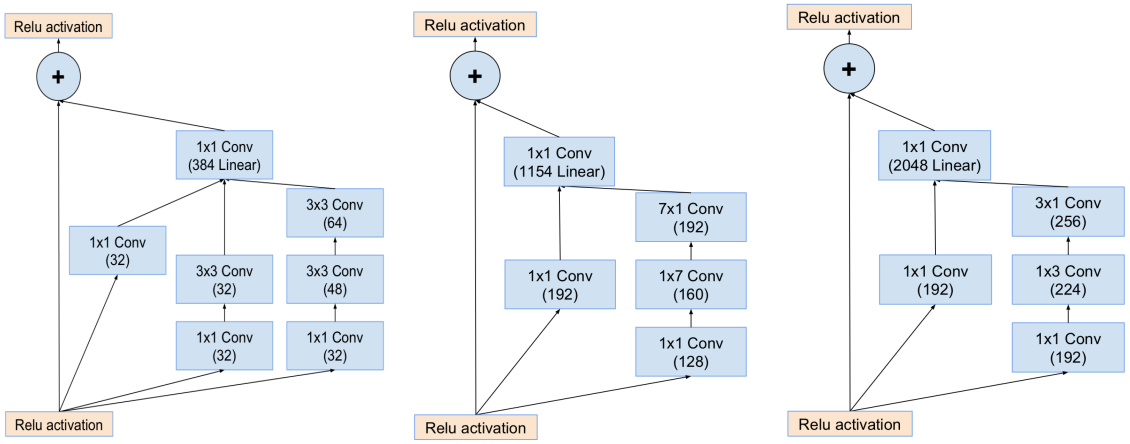
3.4.4 Inception-ResNet-V2

Inception-ResNet-V1 and Inception-ResNet-V2 were proposed in the same paper [89] Szegeday, Ioffe et al. in 2016. Inception-V3 and ResNet inspired Inception-Resnet-V1 and V2. Both of them are variations of the Inception-V3 model. The only difference between Inception-ResNet-V1 and V2 is the hyper-parameter settings. The Inception-ResNet-V2 model is with a relatively lower computation cost but higher accuracy. Consequently, Inception-ResNet-V2 became a widespread network for classification tasks in practical.

Inception-ResNet-V2 is a CNN with 164 layers deep, which trains the ImageNet database. The hypothesis of the model is a combination of inception blocks architecture and residual connections. Three Inception-ResNet modules A, B, and C, as shown in Figure 15, are the fundamental blocks used in Inception-ResNet-V2. They are similar but simplified from Inception-v3 modules. We can discover convolutional layers combined with residual connections. These procedures not only reduce computation time but also prevent degradation problems resulting from deep structures.

Another immediate modification in Inception-ResNet-V2 is the pooling operation. They are replaced by reduction blocks, which still contain max pooling but also combine with convolutional filers. Meanwhile, the authors applied to scale to residuals with a value between 0.1 to 0.3 before adding to previous layer activation to avoid the model “died” and increase stability.

Overall, the compressed view of Inception-ResNet-V2 architecture is shown in Figure 16 [3].



(a) (b) (c)

Figure 15: (a) Inception-ResNet-A Module, (b) Inception-ResNet-B Module, (c) Inception-ResNet-C Module used in Inception-ResNet-v2 Network [89].

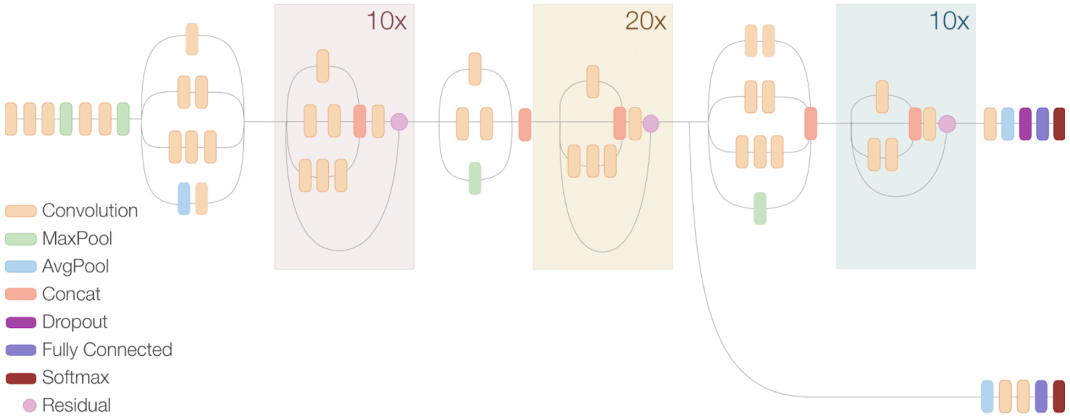


Figure 16: Inception-ResNet-V2 architecture compressed view [3].

Chapter 4

Experiments and Results

Chapter 4 presents the results of three experiments and the practical use of Uncertainty Quantification (UQ). In specific:

- Section 4.1 describes the coding environment, techniques and data pre-processing preparation for the following experiments.
- Section 4.2 presents UQ of binary classification on the ALL-IDB2 database. We focus on introducing how to capture uncertainty by three approaches in classification tasks in this section.
- Section 4.3 shows UQ on another binary classification task on a more extensive and later dataset — SARS-CoV2 dataset. This section focuses on introducing how we use uncertainty results to find the most uncertain images. We also implement the automatic diagnosis system based on the uncertainty results to show one direction of optimizing and improving current classification modelling work.
- Section 4.4 proposes UQ on a multi-class classification task — BreaKHis database. The BreaKHis is a relevant large dataset and composed of images at different magnifications in eight categories. The classification task on histopathological images is usually a time-consuming and challenging task due to its complex nature. We apply UQ measures on BreaKHis and implement UQ applications (e.g. finding the most uncertain category, removing uncertain data to optimize modelling work) as a summary of our work to show the practical utility of Uncertainty Quantification.

4.1 Data Argumentation and Preparation

Our codes are implemented by programming language Python 3.6.9 [95]. The modelling part is built up under Tensorflow 2.3.0 [1] and Keras 2.4.0 [16] frameworks. Uncertainty Quantification is mainly achieved by Numpy 1.18.5 [30] and Scipy 1.4.1 [96] libraries. Plots are generated under Matplotlib 3.3.2 [39] and Seaborn [98] libraries. Before feeding data to models, we apply some common image pre-processing procedures in all the experiments, such as random:

- **Rescaling**. To treat all images simultaneously, we rescale the pixels values by rescaling factor 1./255. In this case, all the pixels which located in $[0, 255]$ (255 as the maximum value of a pixel) to $[0, 1]$.
- **Rotation** refers to random rotations with the setup degree range between $[0, 360]$.
- **Height and Width Shift** implies shifting the input to the left or right(horizontal) and up or down (vertical). In pixels in an input image, a shift means to move all the pixels into one direction.
- **Horizontal and Vertical flips** indicate randomly flipping inputs horizontally or vertically. A flip is to reverse columns or rows of pixels regarding a horizontal or vertical flip, respectively [61]. It is different from the height/width shift.
- **Shearing Intensity** refers to shear angle (unit in degrees) in counter-clockwise direction.
- **Brightness** picks a brightness shift value from the setup range. It can either randomly darken images (value < 1.0) or brighten images (value > 1.0). When the shift value equals 1.0, the image keeps the same brightness.

Excluding these general pre-processing techniques, we also apply other measures depending on the specific characterization. For example:

- **Data argumentation**. The BreaKHis database is highly unbalanced with 3,451 images in the MDC category but 444 images in the BA category. In this case, we apply *Augmentor* [10] image augmentation library to BA category. The *Augmentor* framework allows augmenting images automatically (following

Category	Train	Validation	Test	Total
Normal	91	26	13	130
Probable-lymphoblast	91	26	13	130
Total	182	52	26	260

Table 3: ALL-IDB2 database train/validation/test composition description

artificial data generation) to expand databases as inputs in machine learning algorithms, especially deep learning and NNs.

- **Resizing**. As we mentioned in Chapter 3, we apply pre-trained Transfer Learning modelling techniques and relevant weights in the experiments to promise a reasonable start point. The loaded weights are pre-trained on ImageNet [21]. ImageNet holds more than 14 million images organizing into 1,000 categories. The default size is 299×299 for Inception-V3 and Inception-ResNet V2 and 224×224. Hence we can adjust our input size to achieve better performance.

4.2 Uncertainty Quantification on ALL-IDB2

This section presents our experiment on measuring the uncertainty of the ALL-IDB2 database via multiple UQ approaches.

As we described in 3.2.1, the All-IDB2 database contains two categories of images with a ratio of 1: 1, which is a binary classification task between healthy cells and white blood cells. All the images in ALL-IDB2 are with a size of 256×256. We split the whole database into train, validation and test subsets as the ratio of 7: 2: 1. Table 3 displays the composition of these sets.

4.2.1 Capturing Uncertainty via Three UQ Approaches

This section explains how we obtain uncertainty results by MC dropout, deep ensembles, and ensemble MC dropout approaches in the binary classification task on ALL-IDB2.

We apply three pre-trained models in this experiment, including ResNet, Inception-V3 and Inception-ResNet. To compare the results between different approaches, we

keep the variables and hyper-parameters as same as possible. Accordingly, we then add the same number *Dense* and regular *Dropout* layers with *regularizers* with same parameters to prevent overfitting. Table 4 shows the hyper-parameter configuration.

Hyper-parameter	Value
Batch size	8
Optimizer	Adam [47]
Learning rate	0.0001
Loss Function	Categorical Cross Entropy
Epoch	50
Dropout (both training and test)	0.5

Table 4: Hyper-parameter configuration in ALL-IDB2 experiment

Before the last *Softmax* layer to generate probabilities of each image, we add one activated dropout layer if we use MC dropout and ensemble MC dropout methods, or one regular dropout layer with deep ensembles method.

After the training process, we obtain models and weights. The next step is to generate the uncertainty value of each input based on the predicted output. In specific for each UQ approach:

- **MC dropout**, as we introduced in section 2.3.1, is a method based on Bayesian NNs and activated dropout function at inference time. We receive each image's predicted probability results from the last layer of the model with softmax activation. As an example on *Image_0*, we receive an array [0.3, 0.7]. This array indicates *Image_0* has 0.3 probability of being a healthy-class and 0.7 probability to be a probable-lymphoblast class. Accordingly, our model decides *Image_0* is a probable-lymphoblast since it has higher probability intended be in this class ($0.7 > 0.3$). This distribution is how a normal classification model classifies inputs. By the end, each image will have a prediction as $[x, 1-x]$ predicted probability distribution in binary classification tasks.

  To explore how we can obtain uncertainty, we need to start with MC sampling. An MC sample means one-time dropout activation at inference time. Thus, the MC sample size decides how many times we activate the dropout function at test time. We use the MC sample size = 30 in this experiment, which means we can obtain $30\times$ $[x, 1-x]$ arrays for each image. We choose *entropy* as the computation metrics here. Then we compute the entropy of each image based

on its probability in each MC sample. Then we calculate the average value of 30 MC samples of each image. The final result is the uncertainty of the MC dropout method. Mathematically, it is:

$$Uncertainty_{mcd} = -(\sum_{i=1}^{M}\sum_{j=1}^{c} p_j \log p_j)/M_i$$

where M is MC sample size, c is the number of classes, i, j are the indexes of M, c and p_i of the probability distribution of each image.

- **Deep ensembles** method operates in another way, unlike the other two. With the deep ensembles method, our model is not a BNN with randomness characterization. Therefore, we collect the results of each image trained on five models and compute the average mean. Then we calculate the entropy of these mean values as the uncertainty value on each image in deep ensembles methodology.

- **Ensemble MC dropout** is a combination of MC dropout and deep ensembles. Instead of one BNNs in the MC dropout method, it contains $D = 5$ Bayesian models. Therefore, following the similar way to compute MC dropout but with the average value of these five model, our formula can be illustrated as:

$$Uncertainty_{ens-mcd} = -\sum_{k=1}^{D}((\sum_{i=1}^{M}\sum_{j=1}^{c} p_j \log p_j)/M_i)/D_k$$

where D presents how many models in ensemble MC dropout, k, i, j are the indexes of D, M, c and other variables same as MC dropout uncertainty formula above.

Then we can receive the uncertainty value of each image following the computation ways above. This experiment calculates the mean of uncertainty values from all the images obtained from multiple models and UQ measures. Table 5 presents the uncertainty mean values of samples in the validation/test sets.

We describe how to compute uncertainty with three UQ approaches separately in this section. However, we do not usually compare these values (e.g. mean, median, and variance) directly. In typical cases, we usually combine DL modelling into the comparison between uncertainty evaluation cases. Fortunately, the evaluation methods in DL modelling (e.g. accuracy or AUC[12]) can offer another stable and straightforward way to compare UQ measures more conveniently. In section 4.3, we will explain it in detail.

Dataset	Validation			Test		
Model	ResNet	IncepV3	IncepRes	ResNet	IncepV3	IncepRes
MC dropout	0.1607	0.1217	0.0919	0.1345	0.1192	0.0572
Deep ensembles	0.1204	0.1146	0.0695	0.0723	0.0782	0.0326
Ensemble MC dropout	0.2235	0.1213	0.1247	0.1108	0.1220	0.0911

Table 5: Uncertainty mean value of samples in the validation/test sets by three UQ approaches on ALL-IDB2 database

4.2.2 Evaluation and Association between Uncertainty and Accurateness

From the last section, we learn the uncertainty values of the whole validation/test sets. Further, we wonder whether the accurateness of a model is related to its uncertainty. To justify, we demonstrate the comparison of the uncertainty results between wrong-predicted and right-predicted samples. Table 6 presents the comparison of multiple models by three UQ approaches on validation/test sets. It also concludes the size of wrong/right-predicted samples in each experiment.

From the comparison between wrong-predicted and right-predicted samples, one key finding is that at the most time, the uncertainty of wrong-predicted samples is higher than right-predicted samples. In other words, the samples with lower accuracy are more uncertain than samples with higher accurateness. ALL-IDB is a relevant small database (260 images in total) compared with the later public databases. Therefore, our next experiment is to apply the Uncertainty Quantification approaches to a new public, more massive database — SARS-Cov2 database. We will prove the association between uncertainty, efficiency and accuracy and other new findings related by another method.

4.3 Uncertainty Quantification on SARS-CoV2

In this section, we will explore the uncertainty distribution in the SARS-CoV2 database. Meanwhile, we will learn the association of uncertainty between accuracy and related model optimization measures via UQ methods.

Approach		Validation set			Test set		
MC dropout		ResNet	IncepV3	IncepRes	ResNet	IncepV3	IncepRes
Wrong-pred	Unc	0.4873	0.3697	0.2234	0.3972	0.3566	0.3035
	Num	2	8	6	3	4	2
Right-pred	Unc	0.1919	0.0745	0.1142	0.0969	0.0762	0.0366
	Num	50	44	46	23	22	24
Deep ensembles		ResNet	IncepV3	IncepRes	ResNet	IncepV3	IncepRes
Wrong-pred	Unc	0.3735	N/A	N/A	0.0528	0.5498	0.0003
	Num	3	0	0	2	2	1
Right-pred	Unc	0.1049	0.1146	0.0695	0.0739	0.0389	0.0339
	Num	49	52	52	24	24	25
Ensemble MCD		ResNet	IncepV3	IncepRes	ResNet	IncepV3	IncepRes
Wrong-pred	Unc	0.6422	0.3304	0.6624	0.3144	0.3282	0.6006
	Num	1	4	1	2	2	2
Right-pred	Unc	0.2153	0.1038	0.1142	0.0938	0.1049	0.0486
	Num	51	48	51	24	24	24

Table 6: Uncertainty mean value (*Unc*) and sample size (*Num*) of wrong-predicted (*Wrong-pred*) and right-predicted (*Right-pred*) samples on validation/test sets by MC dropout, deep ensemble and ensemble MC dropout UQ methods on ResNet152V2 (*ResNet*) , Inception-V3 (*IncepV3*) and Inception-ResNet-V2 (*IncepRes*) models.

4.3.1 Finding the Most Uncertain Images in SARS-CoV2

Same as the last experiment, we split the train/validation/test sets with a ratio of 7: 2: 1. The specific data distribution is as shown in Table 7. The hyper-parameter setting is mostly the same as the last experiment. Some modifications apply at the image pre-processing stage, depending on the categorization of SARS-CoV2 images. For example, as the size of images in the database is not the same, e.g. for there exist 277×210, 321×299, 378×281 and other sizes in the same folder. Therefore, we resize all the images to 299×299.

At the classification task stage, we still use pre-trained models (ResNet152V2, Inception-V3 and Inception-ResNet-V2) and added *Dense* and *Dropout* layers in this experiment. We still follow the rule to keep as many as hyper-parameters and variables the same. Lately, at the uncertainty measurement stage, to promise the efficiency of results and computation cost, we still apply $M = 5$ for deep ensembles and ensemble MC dropout methods. The size of the MC samples is 100 in this experiment.

Category	Train	Validation	Test	Total
Infected by SARS-CoV2	877	249	126	1252
Non-infected by SARS-CoV2	864	244	122	1230
Total	1741	493	248	2482

Table 7: SARS-CoV2 train/validation/test sets composition description

From the uncertainty comparison between wrong and right-predicted samples in section 4.2.2, we have preliminary knowledge that higher uncertainty will result in worse accurateness most time. Then we develop how the accuracy will be affected by removing the most uncertain images from the database. If the accuracy/ROC improves when we move a few images with the highest uncertainties from the whole database, we can say that uncertainty and accuracy are positively related. As an example to justify this assumption, we use the experimental setup of ResNet as a pre-trained model and MC dropout as the UQ method on the test set here.

After the workflow of applying ResNet to training data to receive a Bayesian model, feeding test data to the model with $M = 5$ and 100 times MC sampling to obtain the uncertainties, and evaluating outputs by the equation from section 4.2.1, we obtain the uncertainty results and a Bayesian model. The accuracy of this model

on the test set is 0.9635.

The next step is to move the most uncertain samples out of the test set. We use entropy as the computation metrics to capture uncertainty in this experiment. Therefore, the most uncertain samples are the ones with the highest entropy value.

We pick the ten most uncertain samples out of 248 images in the test set. The mean uncertainty value of the test set is 0.0441, with the value of standard deviation 0.1304. Table 8 displays the index and uncertainty value of the ten most uncertain samples.

Index	124	**189**	**183**	220	122	**125**	**179**	**123**	24	**222**
Unc	0.425	0.521	0.589	0.622	0.638	0.648	0.656	0.656	0.678	0.687
Pred	0	**0**	**0**	1	0	**1**	**0**	**1**	0	1
True	0	**1**	**1**	1	0	**0**	**1**	**0**	0	1

Table 8: The indexes (row 1) and uncertainty (row 2) values of ten most uncertain samples in the SARS-CoV2 database by ResNet model and MC dropout method. The samples in **bold** are wrong-predicted found by the model. The prediction outcomes (row 3) and right labels (row 4) are shown as a comparison.

After removing these ten most uncertain samples, the **accuracy** has improved from **0.9635** to **0.9749**. In contrast, the accuracy of these removing samples is only 0.5. To evaluate the results more precisely, we can also use the confusion matrices [92], as shown in Table 9. We categorize the counts of numbers in Table 9 as:

- True Positive (TP) is when predicted class and actual class are both positive, which refers to predicted and actual outcomes are both infected in this task.
- True Negative (TN) is when predicted class and actual class are both negative, which refers to predicted and actual outcomes are both non-infected in this task.
- False Positive (FP) is when the predicted class is positive, and the actual class is negative, which refers to the predicted outcome is infected, and the actual outcome is non-infected.
- False Negative (FN) is when the predicted class is negative, and the actual class is positive, which refers to predicted outcome is the non-infected and actual outcome is infected in this task.

Original Test Set	Predicted Infected	Predicted Non-infected
Actual Infected	123	3
Actual Non-infected	6	116
New Test Set	Predicted Infected	Predicted Non-infected
Actual Infected	120	1
Actual Non-infected	3	114

Table 9: Confusion matrices of the original test set (top) and new test set with ten most uncertain images removed (bottom)

where positive usually refers to diseased and negative usually refers to healthy. Hence in our experiment, we consider infected samples are positive and non-infected samples are negative.

We can obtain the values of specificity and sensitivity from confusion matrix. Specificity and sensitivity [105] are statistical evaluation measures widely used in medical area. The computation equation are:

- **Sensitivity** is the proportion of positives (infected) which are detected properly. Mathematically, it can be described as:

$$Sensitivity = \frac{TP}{TP + FN}$$

- **Specificity** is the proportion of negatives (non-infected) which are detected properly. Mathematically, it can be described as:

$$Specificity = \frac{TN}{TN + FP}$$

After the computation by the equations, we can find the **sensitivity** of class infected has increased from **0.95** to **0.97**, and the **specificity** has increased from **0.98** to **0.99**. These comparison results mean this model's ability to analyze both infected and non-infected has improved by removing the ten most uncertain images.

To summarize this experiment, we discover five wrong-predicted images (from 9 wrong-predicted images in total) by removing the ten most uncertain images (from 248 images in total). The accuracy, sensitivity, and specificity are all improved. We can find that using UQ approaches can make our modelling more efficient and accurate. It can lead the model to extract more specific features by telling the model which samples are more certain than others. In the next section, we will describe how modelling can be optimized systemically and practically and display our experimental results of the SARS-Cov2 database.

4.3.2 Uncertainty Distribution Quantification on Data Retention

This section explains how to combine uncertainty computation methods into DL modelling to better evaluate UQ problems in this experiment.

From the last section, we find that the accuracy of the most uncertain data in the test set is much lower than the rest data (which we can refer them as certain data). In most research work, we often apply models to evaluate all the images in the database. However, in practice, we usually have other choices, e.g. ask experts if we are uncertain. Figure 17 illustrates an example of an automatic diagnosis system in real-world applications. A probabilistic model accepts the test data X^{test}. Then it returns the uncertainty results (e.g. σ_{pred}, H_{pred}). There is an uncertainty threshold of T. The predictions of uncertainty below T will be sent to the model and classified to return the output y^{test}_{pred}. Otherwise, it will be transferred to a medical expert.

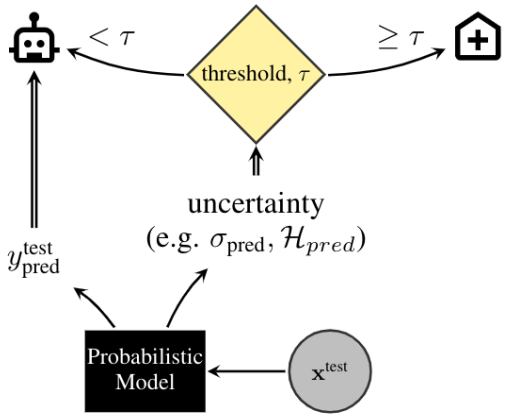


Figure 17: Automatic diagnosis system [23]

A benchmark of Diabetic Retinopathy Tasks has been proposed in 2019, which guides a systemic way to present Bayesian Deep Learning (BDL) in medical diagnostics [23]. They detected diabetic retinopathy and selected the most uncertain samples to an expert for further analysis. Inspired by their work, we compare accuracy with uncertainty results by setting up different thresholds T to better describe how UQ can optimize real-world medical applications.

Instead of referring the most uncertain ten images in last experiment, we remove uncertain samples by data proportion in this experiment. For examples, we obtain uncertain values $u = \{u_1, u_2, ..., u_{10}\}$ from a database S with ten samples

$S = \{S_1, S_2, ..., S_{10}\}$. Then we sort them from largest to lowest as:

$$u_8 > u_4 > u_6 > u_9 > u_2 > u_{10} > u_1 > u_7 > u_3 > u_5$$

which means S_8 has the highest uncertainty, in other words, is the most uncertain sample. $S_4, S_6, S_9, S_2, S_{10}, S_1, S_7, S_3$ follows, and S_5 is the most certain one of ten samples.

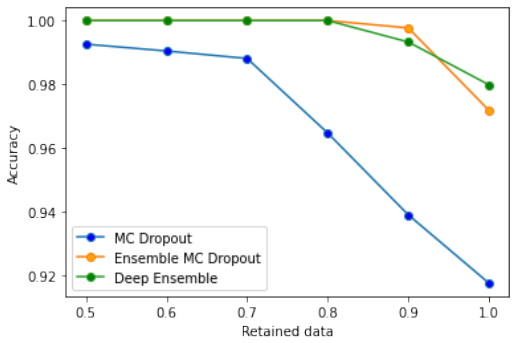
Next, we need to set up a threshold T. Let us assume T as 80%. We want to send the most certain 80% data to the model for analyzing and refer the uncertain rest data to medical experts. It means we need to remove $1 - 80\% = 20\%$ uncertain data from modelling work. In the case of database S, we will remove $20\% \times 10 = 2$ samples from all samples. Hence S_8 and S_4 will be removed since they have the highest uncertainties. At this point, we have 80% data retained in the database. We consider these 80% data as uncertain and safe as *retained data* for further Deep Learning modelling work.

To apply this system to the SARS-CoV2 database, we compare three UQ methodologies with three models for both validation and test sets separately. Figure 18 displays how accuracy (y-axis) behaves based on how much the retained data (x-axis) holds.

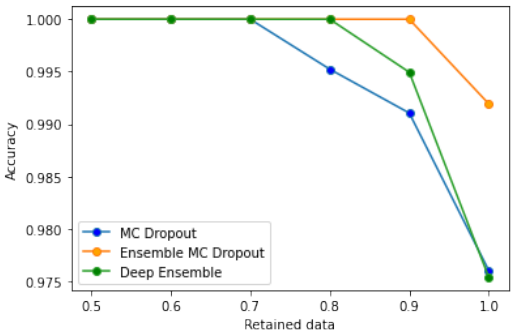
We can discover some findings from the comparison line graphs. For example:

- Ensemble MC dropout usually achieves better accuracy with the same retained proportion than deep ensembles and MC dropout on the SARS-CoV2 database.
- Inception-ResNet has a higher accuracy start point than the other two models for both validation and test sets. We can say Inception-ResNet could be a better classifier among these three models on the SARS-CoV2 database.
- The start points of accuracy on validation sets are higher than test sets.

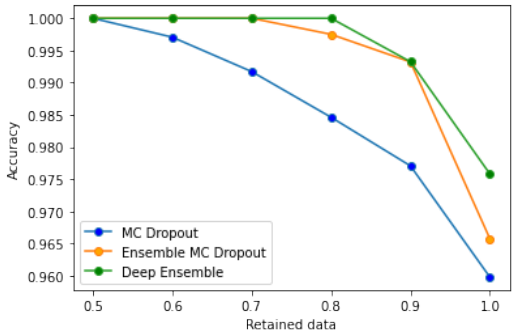
There are still some observations we can learn. The most significant discovery in these experiments is that accuracy keeps improving, along with more uncertain samples removal. We can notice that after removing $40 \sim 50\%$ data, most methods can reach 100% accuracy. Generally, in DL modelling work, we usually use methods, for example, data argumentation and adding layers to make a model deeper to improve accuracy. For example, this experiment offers a new method by uncertainty measurement to better extract primary features and make DL models more efficient. It is



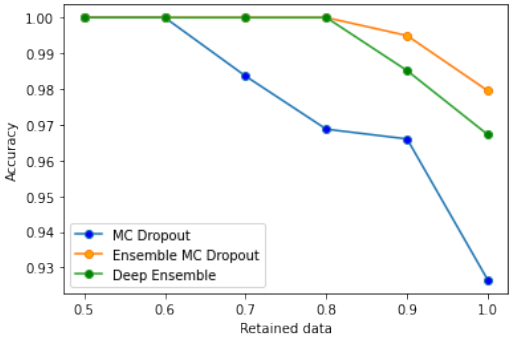
(a) ResNet, Validation



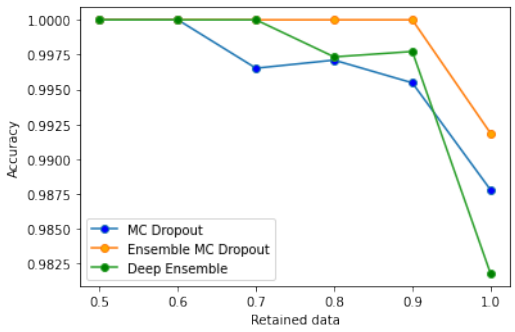
(b) ResNet, Test



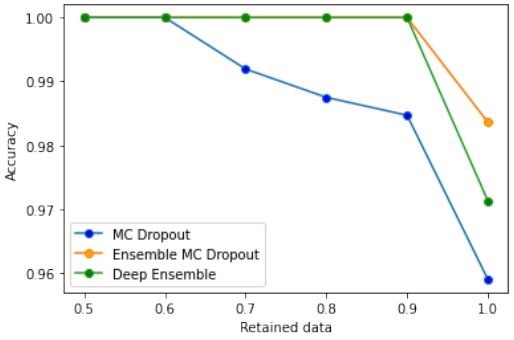
(c) Inception-V3, Validation



(d) Inception-V3, Test



(e) Inception-ResNet, Validation



(f) Inception-ResNet, Test

Figure 18: Accuracy vs retained data distribution of three models via three UQ approaches on validation/test sets. The caption of each plot is named as {*model, dataset*}.

also convenient and useful for further work, e.g. generalization, comparing different models' stability.

In reality, it may not be possible to send around half of the uncertain data to medical experts due to the effectiveness. However, it is still significantly considerable to send the most uncertain samples with a reasonable threshold T to promise the safety and accurateness of a DL modelling work.

4.4 Uncertainty Quantification on BreaKHis

Breast cancer is one of the most common causes of cancers for women [14] [70]. Automatic classification on histopathological images plays an important character in computer-aided breast cancer prognosis, and diagnosis [4]. The multi-classification task on breast cancer images mainly refers to identify subordinate categories (e.g. fibroadenoma, adenosis). BreaKHis database is a large-scale dataset consisting of eight-category and four-magnification breast cancer images. In this section, we present our experiments on the BreaKHis database, summarize the experimental methods we have explored, and propose another UQ application to analyze uncertainty inside each category. Overall, our experiment separates into three phases:

1. classification on eight categories at each magnification factor
2. exploration and comparison of uncertainty at different magnifications
3. investigation on predicting the most uncertain category histopathological images on test sets

Accordingly, we will identify these three phases, display related results and explain discoveries in the following three sections.

4.4.1 Classification of Eight Categories on BreaKHis

Table 10 describes the detailed image distribution under four magnifications (40X, 100X, 200X, and 400X) of eight categories: benign adenosis (BA), benign fibroadenoma (BF), benign phyllodes tumour (BPT), benign tubular adenoma (BTA), malignant carcinoma (MDC), malignant lobular carcinoma (MLC), malignant mucinous carcinoma (MMC) and malignant papillary carcinoma (MPC).

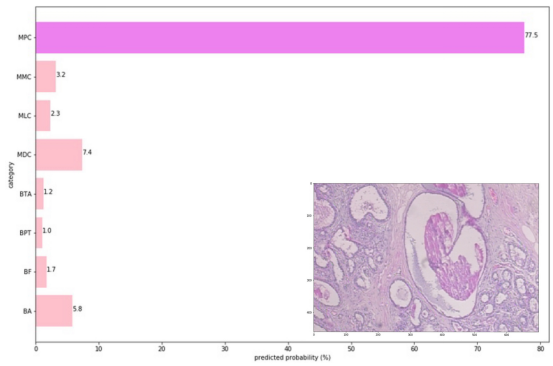
Category	40X	100X	200X	400X	**Total**
BA	114	113	111	106	444
BF	253	260	265	237	1,015
BPT	109	121	108	115	453
BTA	149	150	140	130	569
MDC	864	903	896	787	3,450
MLC	156	170	163	137	626
MMC	205	222	195	169	791
MPC	145	142	135	138	560
Total	1,995	2,081	2,013	1,819	7,908

Table 10: BreaKHis detailed description table with eight sub-category and four magnifications

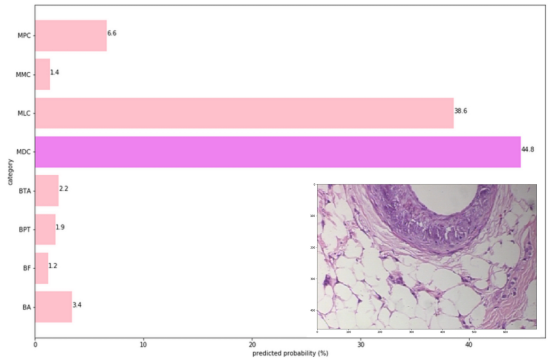
We can notice the imbalance of data distribution from the above table, e.g. *MDC* category contains more images than other categories. Imbalanced data may result in a disproportionate ratio of ignoring some classes' features, then not representing results equally. Accordingly, the first step is data augmentation. We split the original database with the ratio $7:2:1$ for training/validation/test sets. We apply data augmentation on the training and validation sets and keep the test set its original size.

We first explore different modelling with a small amount of data from BreaKHis and review some baselines on BreaKHis multi-classification researches [4] [76] to decide the valid pre-trained models for this experiment. It shows that Inception-V3 and Inception-ResNet-V2 models to BreaKHis returns good classification results as well as acceptable computation cost. Hence, we choose these two pre-trained models as the start point of the classification task. Similar to the last two experiments, some adjustments apply to pre-trained models based on histopathology images' characterization and the large-scale database after augmentation, which includes increasing epoch size, increased *Dense* layers. The probability prediction results of images would be captured, as shown in Figure 19.

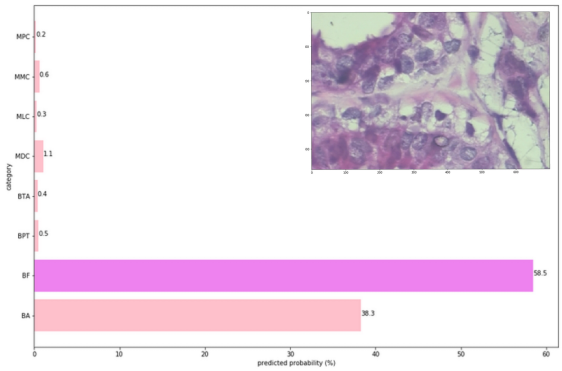
Next, the configuration of UQ estimation methods keeps the same as the last two experiments. One activated *dropout* layer adds before the last *softmax* layer at both the training and test time for MC dropout and ensemble MC dropout. $M = 5$ has been used for both deep ensembles and ensemble MC dropout. The numerical uncertainty value on each image computes by predictive entropy.



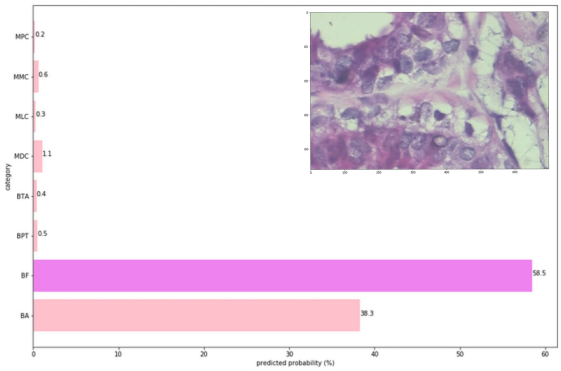
(a) 40X, True Label: MPC, Predicted Label: MPC



(b) 100X, True Label: MDC, Predicted Label: MDC



(c) 200X, True Label: BF, Predicted Label: BF



(d) 400X, True Label: BA, Predicted Label: BF

Figure 19: Example of images at four magnifications with predicted results and true labels presented

Table 11 displays the accuracy of validation and test sets using three UQ approaches with two base models at four magnifications (40X, 100X, 200X and 400X) separately.

Magnification		40X		100X		200X		400X	
Dataset		vali	test	vali	test	vali	test	vali	test
IncepV3	MCD	0.869	0.843	0.916	0.858	0.874	0.786	0.831	0.769
	Deep Ens	0.906	0.899	0.923	0.849	**0.913**	**0.885**	0.865	0.785
	Ens MCD	0.894	0.883	0.934	0.861	0.912	0.819	0.864	0.781
IncepRes	MCD	0.880	0.847	0.936	0.841	0.894	0.795	0.836	0.794
	Deep Ens	0.938	0.882	**0.939**	**0.882**	0.905	0.830	0.884	0.794
	Ens MCD	**0.897**	**0.907**	0.937	0.864	0.912	0.861	**0.896**	**0.825**

Table 11: Accuracy on validation/test sets by MC dropout (*MCD*), Deep ensembles (*Deep Ens*) and Ensemble MC dropout (*Ensemble MCD*) UQ approaches with Inception-V3 (*IncepV3*) and Inception-ResNet-V2 (*IncepRes*) models at four magnifications (*40X, 100X, 200X and 400X*)

4.4.2 Uncertainty Exploration at Four Magnifications

This section shows the accuracy performance and retained data at four magnifications on validation and test sets separately to compare the uncertainty outcomes by line graphs. Each graph contains six plots with the combination of three UQ measures and two models on one set.

Figure 20 presents performance of accuracy vs. retained data of three UQ approaches on **validation** sets. We can find along with removing more uncertain samples by decreasing threshold T; the accuracy keeps increasing. All three UQ approaches perform well as predicted with both Inception-V3 and Inception-ResNet-V2 models on four-magnification sets.

One interesting finding from the modelling aspect is that at the start point, we notice that Inception-ResNet-V2 performs better than Inception-V3 on validation sets at all four magnifications. However, Inception-V3 can achieve better performance during the process of applying UQ measures occasionally. Specifically, when we conclude the best performed combination of {*model, UQ measure*}, we find at 200X {*Inception-V3, Ensemble MC Dropout*} model (orange line in Figure 20 (c)) is the

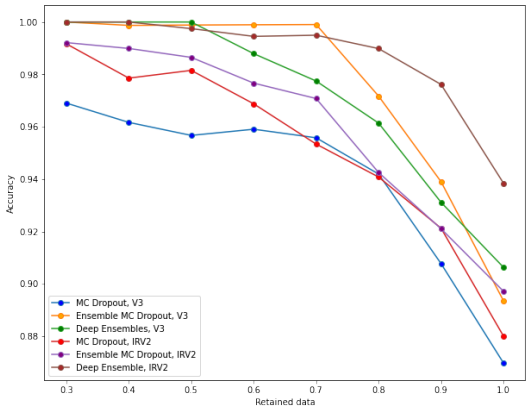
best functioning combination. It also performs as one of the best combinations at 40X. The other two best performed combination at 40X are {*Inception-ResNet-V2, Deep Ensembles*} and {*Inception-V3, Deep Ensembles*}. At 100X and 400X, another combination {*Inception-ResNet-V3, Ensemble MC Dropout*} achieves better performance than others (purple lines in Figure 20 (b) and (d)). Next, we concentrate on how UQ approaches perform unknown datasets — test sets.

Figure 21 displays performance of accuracy vs. retained data of three UQ approaches on **test** sets. The results and trending plots are not as stable as validation sets, especially with the deep ensembles approach. Some characterizations of three UQ methods we can find from the outcomes are:

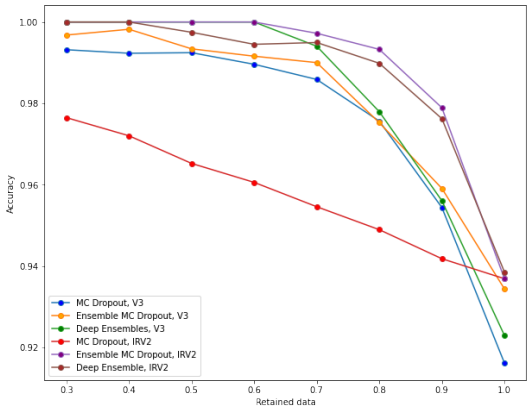
- **Deep ensembles** can achieve great performance and accuracy on validation sets. However, when we apply it to the unknown data, e.g. test set, the outcomes are not as stable as MC dropout and ensemble MC dropout. For example, as we can find in Table 11, {*Inception-V3, Deep Ensembles*} achieves the best performance at both validation and test at 200X at the start point. However, we then look into how it performs under the same condition then find the model not improving when removing uncertain data (green line in Figure 21 (c)). It is not a good sign for generalization. On the contrary, if we choose {*Inception-ResNet-V2, Ensemble MC dropout* } at 200X test instead, the start point of its performance is not as good as deep ensembles. Nevertheless, we can discover its performance has improved relative stably (purple line in Figure 21 (c)), and can even reach to 100% when we set T to 50%.

- As we can learn from Table 11, the accurateness by **MC dropout** is not as good as the other two methods. However, the MC dropout method performs more stable on test set than deep ensembles. It can detect uncertainty and improve modelling as expected. Another advantage of MC dropout is that it can save more computation cost since it only needs one model and MC sampling implementation. Both deep ensembles and ensemble MC dropout are built upon multiple models. Hence, MC dropout would be the preferred method if computation cost is limited.

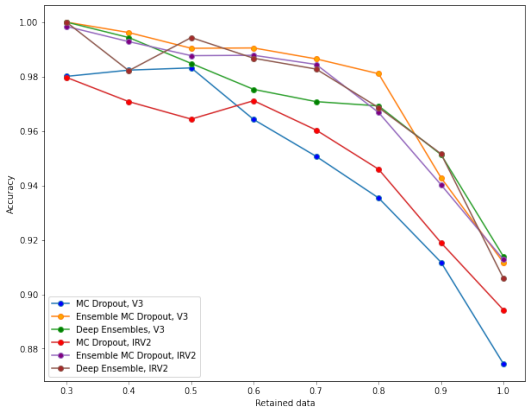
- **Ensemble MC dropout** performs relevantly stable compared with the other two UQ measures. We can also find that both inception models' accuracy is



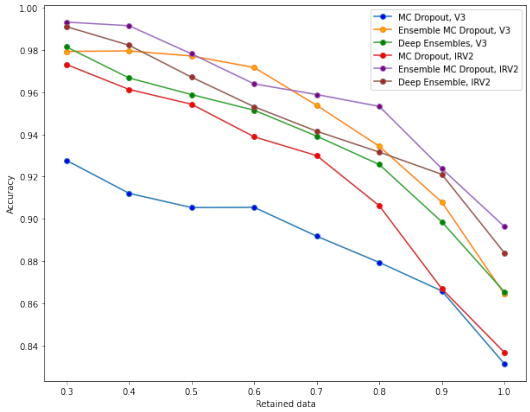
(a) 40X, Validation



(b) 100X, Validation

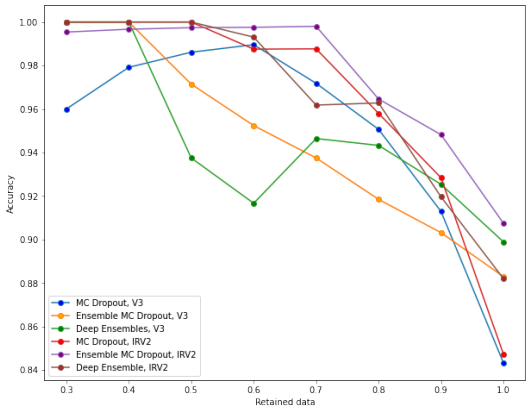


(c) 200X, Validation

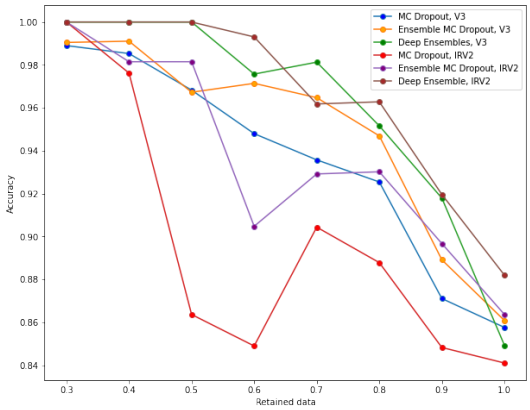


(d) 400X, Validation

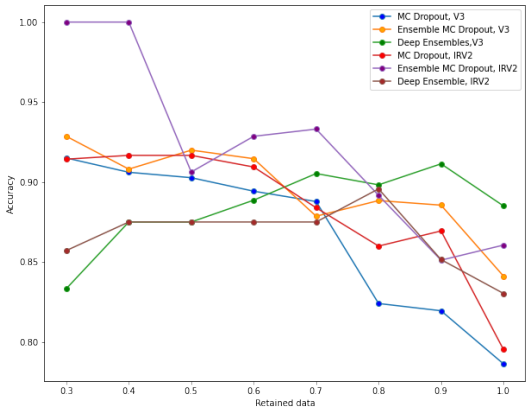
Figure 20: Accuracy vs retained data distribution on **validation** sets at four magnifications. The captions are named in the form of {*Magnification, Dataset*}. Each graph contains six plots presenting the combination of three UQ approaches and two models by different colors. Each plot is named by the way of {*UQ Approach, Model*}, which can be found at the corner of each graph.



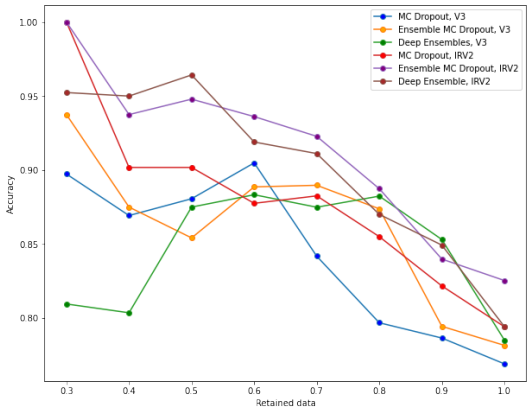
(a) 40X, Test



(b) 100X, Test



(c) 200X, Test



(d) 400X, Test

Figure 21: Accuracy vs retained data distribution on **test** sets at four magnifications. The captions are named in the form of {*Magnification, Dataset*}. Each graph contains six plots presenting the combination of three UQ approaches and two models by different colors. Each plot is named by the way of {*UQ Approach, Model*}, which can be found at the corner of each graph.

higher than the other two UQ measures on test sets. Therefore, for accurateness improvement and potential generalization in the future, we find ensemble MC dropout would be a suitable choice for detecting uncertainty in the eight-class classification task on BreaKHis under the condition of reasonable computational cost.

We set the threshold T as 80 % to keep a balance between modelling and sending uncertain images to medical professionals under the automatic diagnosis system. Then we compare our results while 80% data retained with currently published approaches of multi-classification tasks of BreaKHis, as shown in Table 12 and 13.

References	Methods	Accuracy at four magnifications			
		40X	100X	200X	400X
F. Spanhol et al.(2016) [83]	CNN + patches	85.6	83.5	83.1	80.8
Z. Han et al. (2017) [29]	AlexNet + aug	70.1	75.8	73.6	84.6
	CSDCNN + aug	92.8	93.9	**93.7**	**92.9**
D. Bardou et al. (2018) [6]	CNN + aug	83.97	84.48	80.83	81.03
	Ensemble CNN model	88.23	84.64	83.31	83.98
H. Erfankhah et al. (2019) [22]	LBP	88.3	88.3	87.1	83.4
Y. Jiang et al. (2019) [41]	BHCNet-6 + ERF	94.43	94.45	92.27	91.15
S Boumaraf et al. (2020) [11]	BW fine-tuned ResNet-18	94.49	93.27	91.29	89.56
Present work (80% retained)	IncepV3 + MCD	95.06	92.54	82.40	79.68
	IncepV3 + DE	91.66	95.16	89.82	88.24
	IncepV3 + Ens MCD	91.84	94.70	90.75	87.38
	IncepRes + MCD	95.80	88.78	86.00	85.50
	IncepRes + DE	96.28	**96.28**	89.57	87.01
	IncepRes + Ens MCD	**96.45**	93.02	89.16	88.73

Table 12: Comparison of the accuracy performance of the eight-class classification on BreaKHis with the previous work. For the first six references, 100% of testing data was used.

4.4.3 Finding the Most Uncertain Category

This section proposes another UQ application, which allows us to find the most uncertain category in a multi-classification task. This application can be accomplished based on the comparison results of accuracy and retained data from the last section. We assume the threshold T as 70% and propose one equation for computing *removing*

References	Magnification	AUC	Precision	Recall	F1-measure
D. Bardou et al. (2018) [6]	40X	\	84.27	83.79	83.74
	100X	\	84.29	84.48	84.31
	200X	\	81.85	80.83	80.48
	400X	\	80.84	81.03	80.63
Y. Jiang et al. (2019) [41]	40X	99.76	95.25	95.55	95.39
	100X	99.78	94.51	94.64	94.42
	200X	99.51	90.71	92.24	91.42
	400X	99.30	90.74	91.09	90.75
S Boumaraf et al. (2020) [11]	40X	\	93.81	94.78	94.15
	100X	\	92.94	91.59	92.23
	200X	\	91.18	88.28	89.47
	400X	\	87.97	87.97	87.77
Present work (80% retained)	40X	99.81	97.85	96.46	96.99
	100X	99.62	96.28	96.28	96.28
	200X	95.64	94.83	94.44	93.99
	400X	97.44	91.25	88.74	89.76

Table 13: The evaluation metrics computed from best result of our work at each magnification factor and compare with the previous work. For the first three references, 100% of testing data was used.

percentage, which assists us in finding the most uncertain category:

$$removing\ percentage = \frac{\#\ of\ removed\ images}{\#\ of\ total\ images}$$

Removing percentage refers to present how much proportion the removing samples occupy in the total number of instances of one category. For example, in Figure 21 (a), the most uncertain class we predicted is MLC, with the value of its removing percentage of 0.625. It means 62.5 % of MLC samples would be selected into the most uncertain 30 % samples from all eight-category images in the test set. A higher removing percentage means more images from this category are selected while removing the most uncertain samples. In other words, this category is more uncertain than others. The next step is to rank the *removing percentage* values from maximum to minimum to present the most uncertain category to the most certain one.

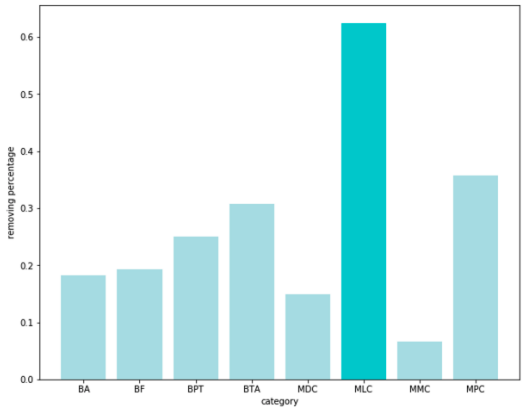
To evaluate our prediction on the most uncertain category, we compute the uncertainty values directly within the test sets. We also rank them from the maximum (most uncertain) to the minimum (most certain) and compare these results by ranking the values of removing percentage. If they point to the same most category, this

means *removing percentage* can be used to predict the most uncertain type as we expected.

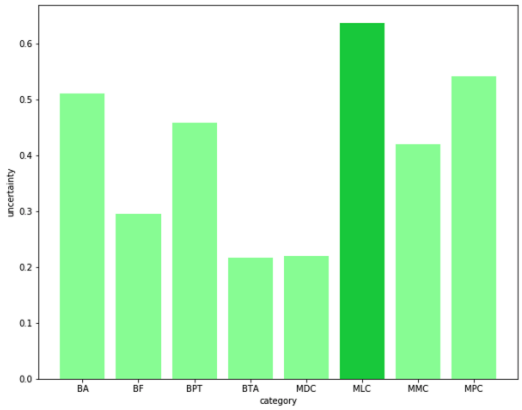
Figure 22 and 23 present our experiments on test sets with {*Inception-ResNet-V2, Ensemble MC Dropout*} at four magnifications. The left plots (blue) are our predicted removing percentage, and the most uncertain categories are highlighted. The right plots (green) are the uncertainty values for true labels.

We can find that by computing *removing percentage* values of each category, we can find the most uncertain types as expected.

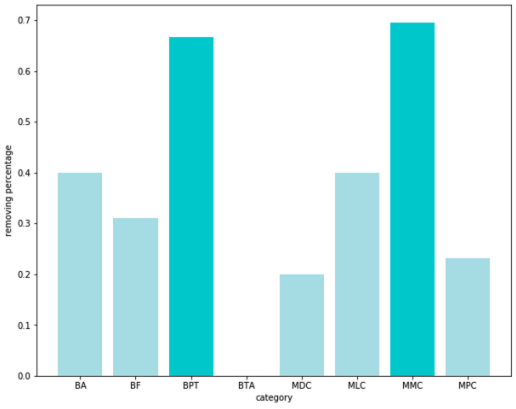
During the experiments, one constraint is that although the most uncertain category features are apparent and can be predicted precisely with a reasonable threshold, the differences between certain varieties are not as noticeable as the uncertain ones. Hence, one of our future work is to optimize the computation and evaluation of the most uncertain category in a multi-classification problem.



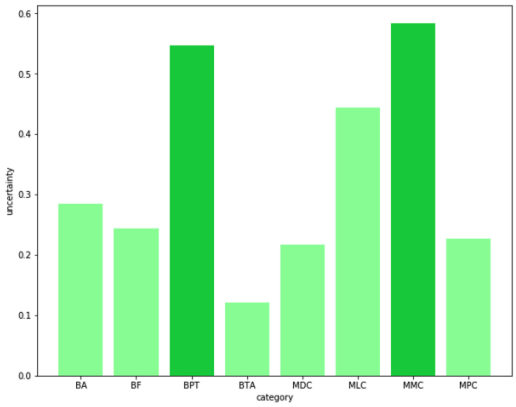
(a) 40X, Prediction



(b) 40X, True

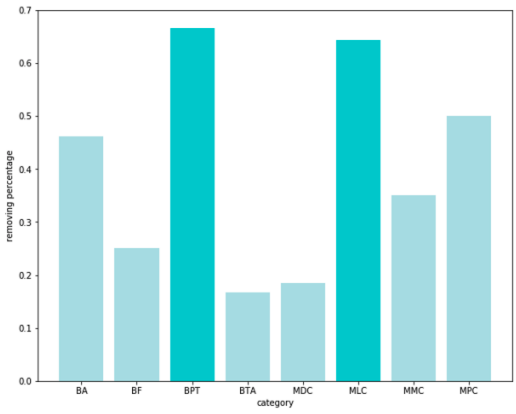


(c) 100X, Prediction

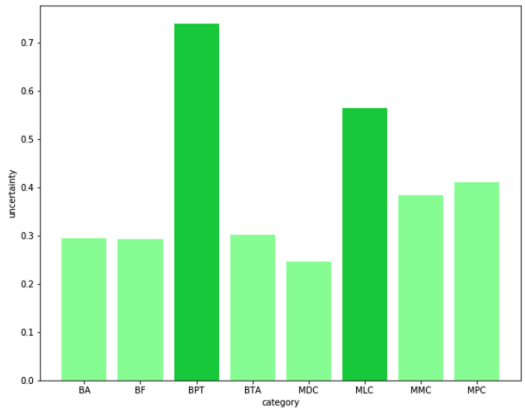


(d) 100X, True

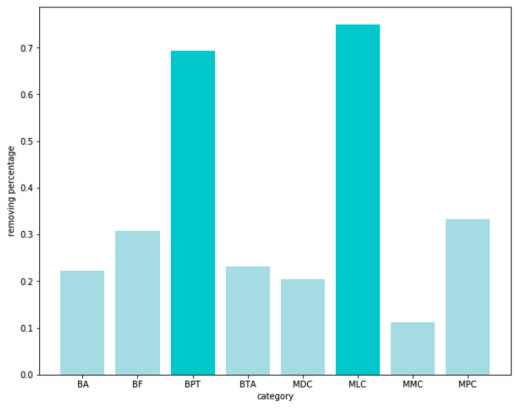
Figure 22: Removing percentage based on uncertainty predictions (left) vs. uncertainty of true labels (right) at magnification 40X and 100X



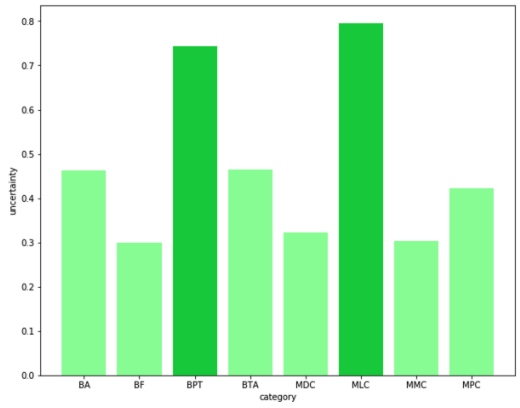
(a) 200X, Prediction



(b) 200X, True



(c) 400X, Prediction



(d) 400X, True

Figure 23: Removing percentage based on uncertainty predictions (left) vs. uncertainty of true labels (right) at magnification 200X and 400X

Chapter 5

Conclusion and Future Work

In this thesis, we learn what uncertainty is and where it comes from and explore the uncertainty quantification (UQ) approaches in binary and multi-classification problems. Meanwhile, we also contribute to exploring how accuracy is related to uncertainty and building up some UQ applications, such as finding the most uncertain samples or the most uncertain category in a multi-class project. Some potential future work directions of our research are:

- There are some great contributions already on segmentation tasks uncertainty quantification. In our work, we went through how to quantify uncertainty in classification tasks. For future work, we consider **evaluating uncertainty on object detection tasks** as a potential topic.

- Another future objective of our research is to **compute two categories of uncertainty separately in multi-classification tasks**. Generally speaking, the uncertainty we calculate is the total uncertainty of the modelling process and database. One of our future work will be computation on aleatoric uncertainty and epistemic uncertainty separately. Some research on predicting two categories of uncertainty on segmentation tasks is published [48], which would be a good start point for us to look into multi-classification studies.

- We also consider the **generalization approach of UQ applications**. To achieve the generalization goal, we will have many components to consider, such as proper thresholds T, cost-effective modelling and computation estimations depending on the databases.

- Various uncertainty approaches and evaluation metrics have been proposed recently, such as Deterministic Uncertainty Quantification (DUQ) method [94] and Mean-Field Variational Inference evaluation metrics [72] [99]. Hence, we are **browsing more approaches to optimize the existing UQ system** we use in this thesis.

Bibliography

[1] Martín Abadi, Ashish Agarwal, and Paul Barham et al. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Last accessed on September 12, 2020: `http://tensorflow.org/`.

[2] Adrian Albert, Jasleen Kaur, and Marta C. González. Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale. *arXiv preprint arXiv: 1704.02965*, 2017.

[3] Alex Alex Alemi. Improving inception and image classification in tensorflow, August 2016. Last accessed on October 21, 2020: `https://ai.googleblog.com/2016/08/improving-inception-and-image.html`.

[4] Md. Zahangir Alom, Chris Yakopcic, Tarek M. Taha, and Vijayan K. Asari. Breast cancer classification from histopathological images with inception recurrent residual convolutional neural network. *arXiv preprint arXiv: 1811.04241*, 2018.

[5] Dario Amodei, Chris Olah, Jacob Steinhardt, Paul F. Christiano, John Schulman, and Dan Mané. Concrete problems in AI safety. *arXiv preprint arXiv: 1606.06565*, 2016.

[6] Dalal Bardou, Kun Zhang, and Sayed Mohammad Ahmad. Classification of breast cancer based on histology images using convolutional neural networks. *IEEE Access*, 6:24680–24693, 2018. doi: 10.1109/ACCESS.2018.2831280.

[7] Edmon Begoli, Tanmoy Bhattacharya, and Dimitri F. Kusnezov. The need for uncertainty quantification in machine-assisted medical decision making. *Nature Machine Intelligence*, 1(1):20–23, 1 2019. doi: 10.1038/s42256-018-0004-1.

[8] Miriam Berger, Shibani Mahtani, Siobhán O'Grady, and Marisa Iati. Hundreds of evacuees to be held on bases in california; hong kong and taiwan restrict travel from mainland china, February 2020. Last accessed on October 19, 2020: `https://www.washingtonpost.com/world/asia_pacific/coronavirus-china-live-updates/2020/02/05/114ced8a-479c-11ea-bc78-8a18f7afcee7_story.html)`.

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

[10] Marcus D Bloice, Peter M Roth, and Andreas Holzinger. Biomedical image augmentation using Augmentor. *Bioinformatics*, 35(21):4522–4524, 2019. ISSN 1367-4803. doi: 10.1093/bioinformatics/btz259.

[11] Said Boumaraf, Xiabi Liu, Zhongshu Zheng, Xiaohong Ma, and Chokri Ferkous. A new transfer learning based approach to magnification dependent and independent classification of breast cancer in histopathological images. *Biomedical Signal Processing and Control*, 63:102192, 2021. ISSN 1746-8094. doi: 10.1016/j.bspc.2020.102192.

[12] Andrew P. Bradley. The use of the area under the roc curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, July 1997. ISSN 0031-3203. doi: 10.1016/S0031-3203(96)00142-2.

[13] Axel Brando, José A. Rodríguez-Serrano, Mauricio Ciprian, Roberto Maestre, and Jordi Vitrià. Uncertainty modelling in deep networks: Forecasting short and noisy series. *arXiv preprint arXiv: 1807.09011*, 2018.

[14] Freddie Bray, Peter McCarron, and D Maxwell Parkin. The changing global patterns of female breast cancer incidence and mortality. *Breast Cancer Research*, 6(6):pp. 229–39, 2004. doi: 10.1186/bcr932.

[15] Guillaume Chaslot, Sander Bakkes, Istvan Szita, and Pieter Spronck. Monte-carlo tree search: A new framework for game ai. In *Proceedings of the Fourth AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, AIIDE'08, page 216–217. AAAI Press, 2008.

[16] François Chollet et al. Keras, 2015. Last accessed on November 11, 2020: `https://keras.io`.

[17] Dan Claudiu Ciresan, Ueli Meier, Luca Maria Gambardella, and Jurgen Schmidhuber. Convolutional neural network committees for handwritten character classification. In *2011 International Conference on Document Analysis and Recognition*, pages 1135–1139, 2011. doi: 10.1109/ICDAR.2011.229.

[18] Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of the 25th International Conference on Machine Learning*, ICML'08, page 160–167, New York, NY, USA, 2008. Association for Computing Machinery. ISBN 9781605582054. doi: 10.1145/1390156.1390177.

[19] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[20] Yann Le Cun, Bernard Boser, John Denker, Donnie Henderson, R. E. Howard, Wayne Hubbard, and Lawrence Jackel. Handwritten digit recognition with a back-propagation network. In *Advances in Neural Information Processing Systems*, pages 396–404. Morgan Kaufmann, 1990.

[21] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Fei-Fei Li. ImageNet: A Large-Scale Hierarchical Image Database. In *CVPR09*, 2009.

[22] Hamed Erfankhah, Mehran Yazdi, Morteza Babaie, and Hamid R. Tizhoosh. Heterogeneity-aware local binary patterns for retrieval of histopathology images. *IEEE Access*, 7:18354–18367, 2019. doi: 10.1109/ACCESS.2019.2897281.

[23] Angelos Filos, Sebastian Farquhar, Aidan N Gomez, Tim GJ Rudner, Zachary Kenton, Lewis Smith, Milad Alizadeh, Arnoud de Kroon, and Yarin Gal. A systematic comparison of bayesian deep learning robustness in diabetic retinopathy tasks. *arXiv preprint arXiv:1912.10481*, 2019.

[24] Simon James Fong, Gloria Li, Nilanjan Dey, Rubén González Crespo, and Enrique Herrera-Viedma. Composite monte carlo decision making under high uncertainty of novel coronavirus epidemic using hybridized deep learning and

fuzzy rule induction. *Applied Soft Computing*, 93:106282, 2020. ISSN 1568-4946. doi: 10.1016/j.asoc.2020.106282.

[25] Yarin Gal and Zoubin Ghahramani. 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, page 1050–1059. JMLR.org, 2016.

[26] Tilmann Gneiting and Adrian E Raftery. Strictly proper scoring rules, prediction, and estimation. *Journal of the American Statistical Association*, 102(477): 359–378, 2007. doi: 10.1198/016214506000001437.

[27] Ian Goodfellow, Jonathon Shlens, and Christian Szegedy. Explaining and harnessing adversarial examples. *arXiv preprint arXiv: 1412.6572*, 2015.

[28] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. Last accessed on October 30, 2020: `http://www.deeplearningbook.org`.

[29] Zhongyi Han, Benzheng Wei, Yuanjie Zheng, Yilong Yin, Kejian Li, and Shuo Li. Breast cancer multi-classification from histopathological images with structured deep learning model. *Scientific Reports*, 7, 06 2017. doi: 10.1038/s41598-017-04075-z.

[30] Charles R. Harris, K. Jarrod Millman, and et al. Array programming with NumPy. *Nature*, 585(7825):357–362, September 2020. doi: 10.1038/s41586-020-2649-2.

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *arXiv preprint arXiv: 1512.03385*, 2015.

[32] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. *arXiv preprint arXiv: 1603.05027*, 2016.

[33] Rochester Hills. *Statistics via Monte Carlo Simulation with Fortran*. MI:JMASM, 2003. ISBN 978-0-9740236-0-1.

[34] Geoffrey Hinton, Li Deng, and Dong Yu et al. Deep neural networks for acoustic modeling in speech recognition: The shared views of four research

groups. *IEEE Signal Processing Magazine*, 29(6):82–97, 2012. doi: 10.1109/MSP.2012.2205597.

[35] Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.*, 6(2):107–116, April 1998. ISSN 0218-4885. doi: 10.1142/S0218488598000094.

[36] J J Hopfield. Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Sciences*, 79 (8):2554–2558, 1982. ISSN 0027-8424. doi: 10.1073/pnas.79.8.2554.

[37] Rein Houthooft, Xi Chen, Yan Duan, John Schulman, Filip De Turck, and Pieter Abbeel. Curiosity-driven exploration in deep reinforcement learning via bayesian neural networks. *arXiv preprint arXiv: 1605.09674*, 2016.

[38] John P. Huelsenbeck, Fredrik Ronquist, Rasmus Nielsen, and Jonathan P. Bollback. Bayesian inference of phylogeny and its impact on evolutionary biology. *Science*, 294(5550):2310–2314, 2001. ISSN 0036-8075. doi: 10.1126/science.1065889.

[39] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing in Science and Engineering*, 9(3):90–95, 2007. doi: 10.1109/MCSE.2007.55.

[40] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv: 1502.03167*, 2015.

[41] Yun Jiang, Li Chen, Hai Zhang, and Xiao Xiao. Breast cancer histopathological image classification using convolutional neural networks with small se-resnet module. *PLOS ONE*, 14(3):1–21, 03 2019. doi: 10.1371/journal.pone.0214587.

[42] Johns Hopkins University (JHU). Covid-19 dashboard by the center for systems science and engineering (csse) at johns hopkins university (jhu), 2020. Last accessed on November 6, 2020: `https://gisanddata.maps.arcgis.com/apps/opsdashboard/index.html`.

[43] James M. Joyce. *Kullback-Leibler Divergence*, pages 720–722. Springer Berlin Heidelberg, Berlin, Heidelberg, 2011. ISBN 978-3-642-04898-2. doi: 10.1007/978-3-642-04898-2_327.

[44] James T. Kajiya. The rendering equation. *SIGGRAPH Computer Graphics*, 20 (4):143–150, August 1986. ISSN 0097-8930. doi: 10.1145/15886.15902.

[45] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5574–5584. Curran Associates, Inc., 2017.

[46] Marc C. Kennedy and Anthony O'Hagan. Bayesian calibration of computer models. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 63(3):425–464. doi: 10.1111/1467-9868.00294.

[47] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[48] Armen Der Kiureghian and Ove Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2), August 2008.

[49] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 2009. Last accessed on October 25, 2020: `http://www.cs.toronto.edu/~kriz/cifar.html`.

[50] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. *Commun. ACM*, 60(6):84–90, May 2017. ISSN 0001-0782. doi: 10.1145/3065386.

[51] Dirk P. Kroese, T. Brereton, T. Taimre, and Z. Botev. Why the monte carlo method is so important today. *Wiley Interdisciplinary Reviews: Computational Statistics*, 6:386–392, 2014.

[52] Volodymyr Kuleshov, Nathan Fenner, and Stefano Ermon. Accurate uncertainties for deep learning using calibrated regression. *arXiv preprint arXiv: 1807.00263*, 2018.

[53] R. D. Labati, V. Piuri, and F. Scotti. All-idb: The acute lymphoblastic leukemia image database for image processing. In *2011 18th IEEE International Conference on Image Processing*, pages 2045–2048, 2011.

[54] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles, 2017.

[55] Yann LeCun and Yoshua Bengio. *Convolutional Networks for Images, Speech, and Time Series*, page 255–258. MIT Press, Cambridge, MA, USA, 1998. ISBN 0262511029.

[56] Yann LeCun, Patrick Haffner, Léon Bottou, and Yoshua Bengio. Object recognition with gradient-based learning. In *Shape, Contour and Grouping in Computer Vision*, page 319, Berlin, Heidelberg, 1999. Springer-Verlag. ISBN 3540667229.

[57] Jaakko Lehtinen, Jacob Munkberg, Jon Hasselgren, Samuli Laine, Tero Karras, Miika Aittala, and Timo Aila. Noise2noise: Learning image restoration without clean data. *arXiv preprint arXiv: 1803.04189*, 2018.

[58] Yuan-Pin Lin and Tzyy-Ping Jung. Improving eeg-based emotion classification using conditional transfer learning. *Frontiers in Human Neuroscience*, 11:334, 2017. ISSN 1662-5161. doi: 10.3389/fnhum.2017.00334.

[59] Geert Litjens, Thijs Kooi, Babak Ehteshami Bejnordi, Arnaud Arindra Adiyoso Setio, Francesco Ciompi, Mohsen Ghafoorian, Jeroen A.W.M. van der Laak, Bram van Ginneken, and Clara I. Sánchez. A survey on deep learning in medical image analysis. *Medical Image Analysis*, 42:60 – 88, 2017. ISSN 1361-8415. doi: 10.1016/j.media.2017.07.005.

[60] Xiaojiao Mao, Chunhua Shen, and Yu-Bin Yang. Image restoration using very deep convolutional encoder-decoder networks with symmetric skip connections. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29, pages 2802–2810. Curran Associates, Inc., 2016.

[61] Machine Learning Mastery. How to configure image data augmentation when training deep learning neural networks, April 2019. Last accessed on October 21,

2020: https://mc.ai/how-to-configure-image-data-augmentation-when-training-deep-learning-neural-networks/.

[62] Mayo Clinic Staff. Coronavirus disease 2019 (covid-19), September 2020. Last accessed on October 25, 2020: https://www.mayoclinic.org/diseases-conditions/coronavirus/symptoms-causes/syc-20479963.

[63] Mariusz Milik and Jeffrey Skolnick. Insertion of peptide chains into lipid membranes: an off-lattice monte carlo dynamics model. *Proteins*, 15 1:10–25, 1993.

[64] Seonwoo Min, Byunghan Lee, and Sungroh Yoon. Deep learning in bioinformatics. *Briefings in Bioinformatics*, 18(5):851–869, 07 2016. ISSN 1467-5463. doi: 10.1093/bib/bbw068.

[65] Marvin Minsky and Seymour Papert. *Perceptrons: An Introduction to Computational Geometry*. MIT Press, Cambridge, MA, USA, 1969.

[66] Hothaifa Al-Qassab Mohammed Al-Qizwini, Iman Barjasteh and Hayder Radha. Deep learning algorithm for autonomous driving using googlenet. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 89–96, 2017. doi: 10.1109/IVS.2017.7995703.

[67] Tanya Nair, Doina Precup, Douglas L. Arnold, and Tal Arbel. Exploring uncertainty measures in deep networks for multiple sclerosis lesion detection and segmentation. *arXiv preprint arXiv: 1808.01200*, 2018.

[68] Andrew Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *Proceedings of the Twenty-First International Conference on Machine Learning*, ICML'04, page 78, New York, NY, USA, 2004. Association for Computing Machinery. ISBN 1581138385. doi: 10.1145/1015330.1015435.

[69] Petri Nokelainen, Timo Nevalainen, and Kreeta Niemi. *Mind or Machine? Opportunities and Limits of Automation*, pages 13–24. Springer International Publishing, Cham, 2018. ISBN 978-3-319-63257-5. doi: 10.1007/978-3-319-63257-5_2.

[70] Global Burden of Disease Cancer Collaboration. Global, Regional, and National Cancer Incidence, Mortality, Years of Life Lost, Years Lived With Disability,

and Disability-Adjusted Life-Years for 29 Cancer Groups, 1990 to 2017: A Systematic Analysis for the Global Burden of Disease Study. *JAMA Oncology*, 5(12):1749–1768, 12 2019. ISSN 2374-2437. doi: 10.1001/jamaoncol.2019.2996.

[71] Pedro Ojeda, Martin E Garcia, Aurora Londoño, and Nan-Yow Chen. Monte carlo simulations of proteins in cages: influence of confinement on the stability of intermediate states. *Biophysical journal*, 96(3), February 2009. doi: 10.1529/biophysj.107.125369.

[72] Carsten Peterson and James R. Anderson. A mean field theory learning algorithm for neural networks. *Complex Systems*, 1(5), 2006.

[73] Carl Edward Rasmussen. *Gaussian Processes in Machine Learning*, pages 63–71. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004. ISBN 978-3-540-28650-9. doi: 10.1007/978-3-540-28650-9_4.

[74] Christian P. Robert and George Casella. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2005. ISBN 0387212396.

[75] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, Alexander C. Berg, and Li Fei-Fei. ImageNet Large Scale Visual Recognition Challenge. *International Journal of Computer Vision (IJCV)*, 115(3):211–252, 2015. doi: 10.1007/s11263-015-0816-y.

[76] Ankur Satpute. Breakhist-dataset-image-classification. *GitHub repository*, 2020. Last accessed on September 15, 2020: `https://github.com/Anki0909/BreakHist-Dataset-Image-Classification`.

[77] Jude Shavlik and Lisa Torrey. *Handbook of Research on Machine Learning Applications*. IGI Global, 2009. Last accessed on October 12, 2020: `ftp://ftp.cs.wisc.edu/machine-learning/shavlik-group/torrey.handbook09.pdf`.

[78] Dinggang Shen, Guorong Wu, and Heung-Il Suk. Deep learning in medical image analysis. *Annual Review of Biomedical Engineering*, 19(1):221–248, 2017. doi: 10.1146/annurev-bioeng-071516-044442. PMID: 28301734.

[79] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[80] Lewis Smith and Yarin Gal. Understanding measures of uncertainty for adversarial example detection. *arXiv preprint arXiv: 1803.08533*, 2018.

[81] Eduardo Soares, Plamen Angelov, Sarah Biaso, Michele Higa Froes, and Daniel Kanda Abe. Sars-cov-2 ct-scan dataset: A large dataset of real patients ct scans for sars-cov-2 identification. *medRxiv*, 2020. doi: 10.1101/2020.04.24.20078584.

[82] Fabio A. Spanhol, lUIZ S Oliveira, Caroline Petitjean, and Laurent Heutte. A dataset for breast cancer histopathological image classification. *IEEE Transactions on Biomedical Engineering (TBME), 63(7):1455-1462*, 2016.

[83] Fabio Alexandre Spanhol, Luiz S. Oliveira, Caroline Petitjean, and Laurent Heutte. Breast cancer histopathological image classification using convolutional neural networks. In *2016 International Joint Conference on Neural Networks (IJCNN)*, pages 2560–2567, 2016. doi: 10.1109/IJCNN.2016.7727519.

[84] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, 2014.

[85] Stanford University. Convolutional neural networks (cnns / convnets), 2020. Last accessed on October 21, 2020: `https://cs231n.github.io/convolutional-networks/`.

[86] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott E. Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. *arXiv preprint arXiv: /1409.4842*, 2014.

[87] Christian Szegedy, Wojciech Zaremba, Ilya Sutskever, Joan Bruna, Dumitru Erhan, Ian Goodfellow, and Rob Fergus. Intriguing properties of neural networks. *arXiv preprint arXiv: 1312.6199*, 2014.

[88] Christian Szegedy, Vincent Vanhoucke, Sergey Ioffe, Jonathon Shlens, and Zbigniew Wojna. Rethinking the inception architecture for computer vision. *arXiv preprint arXiv: 1512.00567*, 2015.

[89] Christian Szegedy, Sergey Ioffe, and Vincent Vanhoucke. Inception-v4, inception-resnet and the impact of residual connections on learning. *arXiv preprint arXiv: 1602.07261*, 2016.

[90] Binhua Tang, Zixiang Pan, Kang Yin, and Asif Khateeb. Recent advances of deep learning in bioinformatics and computational biology. *Frontiers in Genetics*, 10:214, 2019. ISSN 1664-8021. doi: 10.3389/fgene.2019.00214.

[91] The Franklin Institute Award. Kunihiko fukushima, Apr 2020. Last accessed on November 29, 2020: `https://www.fi.edu/laureates/kunihiko-fukushima`.

[92] Kai Ming Ting. *Confusion Matrix*, pages 260–260. Springer US, Boston, MA, 2017. ISBN 978-1-4899-7687-1. doi: 10.1007/978-1-4899-7687-1_50.

[93] Ayşegül Uçar, Yakup Demir, and Cüneyt Güzeliş. Object recognition and detection with deep learning for autonomous driving applications. *SIMULATION*, 93(9):759–769, 2017. doi: 10.1177/0037549717709932.

[94] Joost van Amersfoort, Lewis Smith, Yee Whye Teh, and Yarin Gal. Uncertainty estimation using a single deep deterministic neural network. *arXiv preprint arXiv: 2003.02037*, 2020.

[95] Guido Van Rossum and Fred L. Drake. *Python 3 Reference Manual*. CreateSpace, Scotts Valley, CA, 2009. ISBN 1441412697.

[96] Pauli Virtanen, Ralf Gommers, and et al. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[97] Pin Wang, Xianling Hu, Yongming Li, Qianqian Liu, and Xinjian Zhu. Automatic cell nuclei segmentation and classification of breast cancer histopathology images. *Signal Processing*, 122:1 – 13, 2016. ISSN 0165-1684. doi: 10.1016/j.sigpro.2015.11.011.

[98] Michael Waskom and the seaborn development team. Mwaskom/seaborn, September 2020. Last accessed on November 1, 2020: https://seaborn.pydata.org/index.html.

[99] Yeming Wen, Paul Vicol, Jimmy Ba, Dustin Tran, and Roger Grosse. Flipout: Efficient pseudo-independent weight perturbations on mini-batches. *arXiv preprint arXiv: 1803.04386*, 2018.

[100] Wikipedia. Max-pooling / pooling, 2017. Last accessed on October 20, 2020: https://computersciencewiki.org/index.php/Max-pooling_/_Pooling.

[101] World Health Organization (WHO). Statement on the second meeting of the international health regulations (2005) emergency committee regarding the outbreak of novel coronavirus (2019-ncov), January 2020. Last accessed on October 6, 2020: https://www.who.int/news/item/30-01-2020-statement-on-the-second-meeting-of-the-international-health-regulations-(2005)-emergency-committee-regarding-the-outbreak-of-novel-coronavirus-(2019-ncov).

[102] World Health Organization (WHO). Who director-general's opening remarks at the media briefing on covid-19 - 11 march 2020, March 2020. Last accessed on October 6, 2020: https://www.who.int/dg/speeches/detail/who-director-general-s-opening-remarks-at-the-media-briefing-on-covid-19---11-march-2020.

[103] Liu Yang, Steve Hanneke, and Jaime Carbonell. A theory of transfer learning with applications to active learning. *Machine Learning*, 90(2):161–189, February 2013. ISSN 0885-6125.

[104] Yoshua Bengio Yann Lecun, Leon Bottou and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11): 2278–2324, 1998.

[105] Jacob Yerushalmy. Statistical problems in assessing methods of medical diagnosis, with special reference to x-ray techniques. *Public Health Reports (1896-1970)*, 62(40):1432–1449, 1947. ISSN 00946214.

[106] Tom Young, Devamanyu Hazarika, Soujanya Poria, and Erik Cambria. Recent trends in deep learning based natural language processing. *IEEE Computational Intelligence Magazine*, 13(3):55–75, 2018. doi: 10.1109/MCI.2018.2840738.