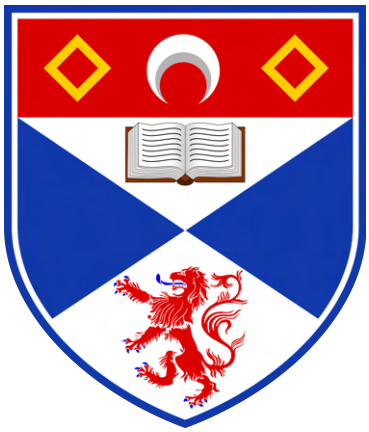
CS5199 Individual Masters Project:

Breast Cancer Detection In Low Resolution Images

Nathan Poole
Matriculation Number: 170004680



University of St Andrews
School of Computer Science

Supervisors: Christina Fell, Dr David Harris-Birtill

January 14, 2022

Abstract

Early metastasis detection is a crucial prognostic factor for breast cancer patients. The current clinical process is a laborious and manual workload on pathologists. These factors have driven recent research in applying deep learning techniques towards breast cancer detection. The resulting state-of-the-art systems can perform comparably to pathologists for tumour detection and localisation within pathology slides. However, these systems are not clinically viable as they require significant time or compute power to process high-resolution images.

This project's objective was to investigate breast cancer detection via deep learning techniques using low-resolution whole-slide images. The intent was to reveal whether machine learning models could be developed that provide high confidence results using fractional resources by using low- versus high-resolution images.

A deep learning pipeline has been developed which reduces high-resolution whole-slide images to a sufficiently small size so they can be fed as input into a pre-trained CNN for binary classification (i.e., cancer or normal). Several improvements have been implemented to boost general performance, namely supplementing the training data, adding data augmentations, and applying tissue detection.

Ultimately, the developed low-resolution models are effectively skill-less for very low-resolution inputs (i.e., 299 x 299 to 2048 x 2048 pixels). An observed significant decrease in model inference time is a superficial benefit given the loss in classification capability. This is a disappointing but expected result arising from an obvious limitation with the methodology; very low-resolutions, effectively zooming out on a slide, preserves only information about macro-cellular structures which is often insufficient alone to detect cancer.

In conclusion, the developed deep learning pipeline is unsuitable as a clinically viable cancer detection system when using the input resolutions investigated. Thus, this deep learning pipeline is insufficient as a prior filter to quickly classify easy pathology cases, which would justify use of the slow, high-performance *Camelyon* submissions for difficult but rare cases. There may exist some optimal input resolution that balances prediction performance and resource requirements which was not empirically investigated in this project. However, a clinically viable approach would more reasonably encapsulate both the rapidness of low-resolution slide processing with the confidence of high-resolution slide processing. The effectiveness of such multi-resolution approaches has been demonstrated in the literature but has yet to be applied to metastasis detection in pathology slides.

Keywords

Deep Learning; Convolutional Neural Networks; Medical Image Analysis; Breast Cancer Detection; Whole-Slide Classification; Tumour Localisation; Whole-Slide Images; Low Resolution; Camelyon.

Declaration

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgement. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is approximately 18,500 words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bonafide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

Signed:  Date: January 14, 2022

Ethics

The data sets used for training and evaluating the proposed machine learning models contain special category data: data concerning health in the form of histopathology whole-slide images.

These data sets, *Camelyon-16* and *Camelyon-17*, are open access (i.e., publicly available) and intended for use in research. The health data in the data sets was collected from 5 medical centres in the Netherlands, where the sources of the data were fully anonymised. The collection of the data was approved by the local ethics committee of the Radboud University Medical Center (RUMC), and the need for informed consent was waived due to the completely anonymous nature of the data sets.

Use of the *Camelyon-16* and *Camelyon-17* data sets is licensed under the '`CC0 Universal (CC0 1.0) Public Domain Dedication`'. The data sets are used only for research and not for commercial purposes. A full ethics application was submitted to the University Teaching and Research Ethics Committee (UTREC) at the University of St Andrews, whom authorised the project (see section 8.1 for the approval letter).

Contents

Chapter 1

Introduction

1.1 Problem Description

Machine learning systems exist for the automatic detection of breast cancer in histopathology whole-slide images with high confidence. Such systems can potentially automate large portions of conventional diagnostic procedures used to identify breast cancer, improving support for diagnoses via digital second opinion or reducing cognitive load by shifting work away from medical personnel.

However, these current systems are complex as they often fully utilise high-resolution whole-slide images with dimensions that are hundreds of thousands of pixels in width and height. Such images represent pathology slides at considerably high magnification. Due to the high resolution of the images, these systems are typically resource intensive, requiring either significant time or compute power, which hinders their clinical viability.

This project investigates automated breast cancer detection via deep learning techniques using lower resolution images (i.e., digital histopathology slides at a lower magnification). The investigation intends to reveal whether machine learning models can be developed that provide high confidence results with some fractional amount of resources by using low- versus high-resolution whole-slide images.

1.2 Project Objectives

The following are the primary objectives of the project:

- ☑ Perform a literature review analysing state-of-the-art machine learning techniques for the automated classification and localisation of cancer using low- and high-resolution whole-slide images. This literature review will consider current models in the context of metastases detection with whole-slide images of lymph node sections, as given by the *Camelyon-16* and *Camelyon-17* data sets, as well as models developed in the wider context of automated cancer detection.
- ☑ Develop and optimise (e.g., via hyper-parameter tuning) a machine learning model for the classification of breast cancer in low-resolution histopathology slides using techniques (e.g. ResNet) informed by the literature review.
- ☑ Perform a results comparison between the techniques analysed in the literature review and the developed model for low-resolution whole-slide images. This comparison should assess differences between techniques that use higher versus lower resolution images in terms of detection confidence and resource requirements, and relate this to the observed performance of actual pathologists.

The following secondary objectives extend the project:

- ☑ Improve the techniques used to develop the low-resolution model(s). Perhaps the developed techniques can be adapted further to yield better performance. Alternatively, implement a different technique (also informed by the literature review) for comparison to the already developed low-resolution image model(s). Perhaps the choice of technique had an effect on the ability to classify at low image resolutions.

☑ Investigate a range of image resolutions to determine possible image resolution 'boundaries' for performance. Perhaps some minimum whole-slide image resolution is required for acceptable result confidence, whilst beyond some maximum resolution does not significantly improve the confidence of results and, therefore, needlessly uses excessive resources.

The following tertiary objectives further extend the project:

☐ Develop/extend and optimise a machine learning model for tumour localisation using low-resolution images.

☐ Perform a results comparison between the models developed for whole-slide classification and tumour localisation using low-resolution images. The comparison should investigate which of the developed classification and localisation models provides higher confidence results for low-resolution images and relate this to existing models from the literature that use high-resolution images.

1.3 Extent Of Project Success

Overall, 5 of the 7 project objectives have been completed, with future work being detailed for the remaining objectives.

Primary Objectives: A comprehensive context survey has been performed (see chapter 2). A deep learning pipeline for cancer classification in low-resolution pathology slides has been developed and optimised (see chapter 3). A results analysis and comparison of this pipeline is detailed (see chapter 4 and chapter 5).

Secondary Objectives: The deep learning pipeline for cancer classification has undergone several iterations of technique improvement (see chapter 3). A range of input resolutions has been investigated (see chapter 4 and chapter 5).

Tertiary Objectives: Whilst the tertiary objectives have not been completed, future work on the implementation and motivations concerning these objectives have been explained (see section 3.3).

1.4 Project Report Structure

- **Introduction** - Describes the problem investigated by this project, followed by specific project objectives and the extent of success in completing them.
- **Context Survey** - Surveys the context, background literature, and related work concerning automated breast cancer detection in histopathology whole-slide images. The historiography of breast cancer detection is established, leading to the current state-of-the-art. Techniques at the state-of-the-art are surveyed in depth via the *Camelyon-16* and *Camelyon-17* challenges. Low-resolution techniques from the wider medical image analysis literature are also explored.
- **Method** - Justifies the high-level design considerations of the project, specifically regarding the deep learning pipelines.
- **Results** - Presents and analyses the results for this project's developed low-resolution models.
- **Discussion** - Discusses the results of the developed low-resolution models in terms of prediction confidence and resource requirements, whilst comparing to other state-of-the-art techniques and actual pathologists.
- **Conclusions** - Critically evaluates the project by discussing the accomplished objectives, project limitations, and future work.

Chapter 2

Context Survey

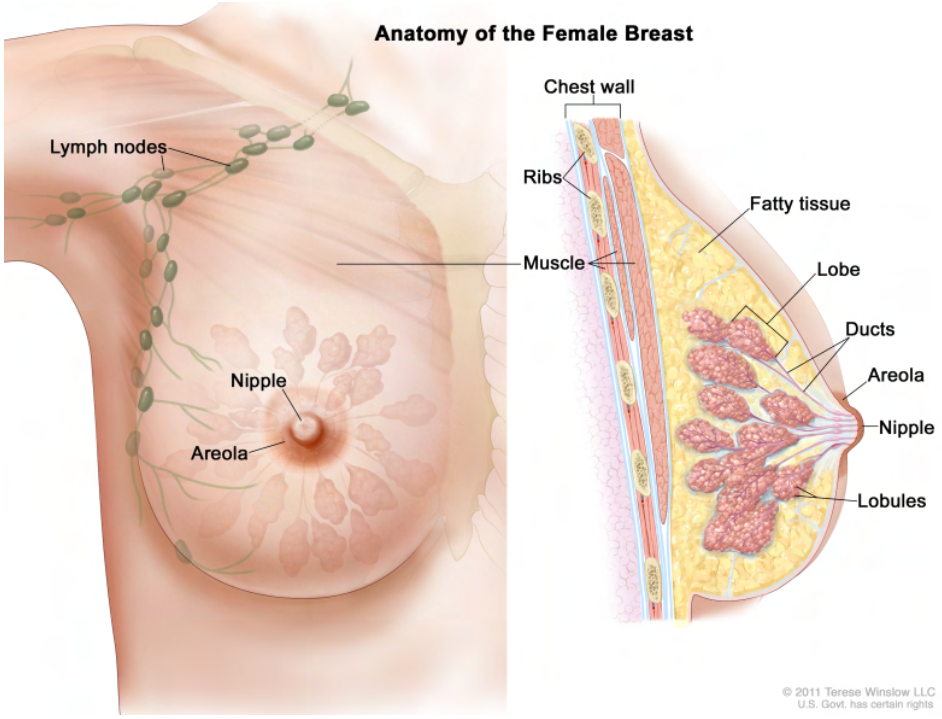
In this chapter, the medical context regarding breast cancer diagnosis in pathology slides is established. The historiography of breast cancer detection is surveyed, from manual conventional diagnosis to automated state-of-the-art deep learning techniques. Current state-of-the-art techniques are surveyed in depth by assessing submissions to the *Camelyon-16* and *Camelyon-17* challenges ("Camelyon-16", 2015; "Camelyon-17", 2016). Other techniques using low-resolution images from the broader field of medical image analysis are also considered. Thus, this chapter places this project within the medical image analysis context and provides guidance for design decisions pertaining to this project's developed low-resolution image models.

2.1 Conventional Histopathological Breast Cancer Diagnosis

The severity of a patient's cancer diagnosis is dependant on the stage of the cancer. Cancer staging is determined by factors such as the size and location of the tumour, and more importantly whether the cancer has metastasised (i.e., the cancer has spread). Prognosis for breast cancer patients in England is generally optimistic, with an average 5-year overall survival rate of 85%. However, this significantly deteriorates to 26.2% when the breast cancer metastasises ("Cancer Survival In England - Adults Diagnosed", 2019). The early diagnosis of breast cancer can improve prognosis and chance of survival, crucially so when the presence of metastases are established. Even when metastases are not present, the accurate classification of such can prevent patients undergoing unnecessary, possibly detrimental treatments.

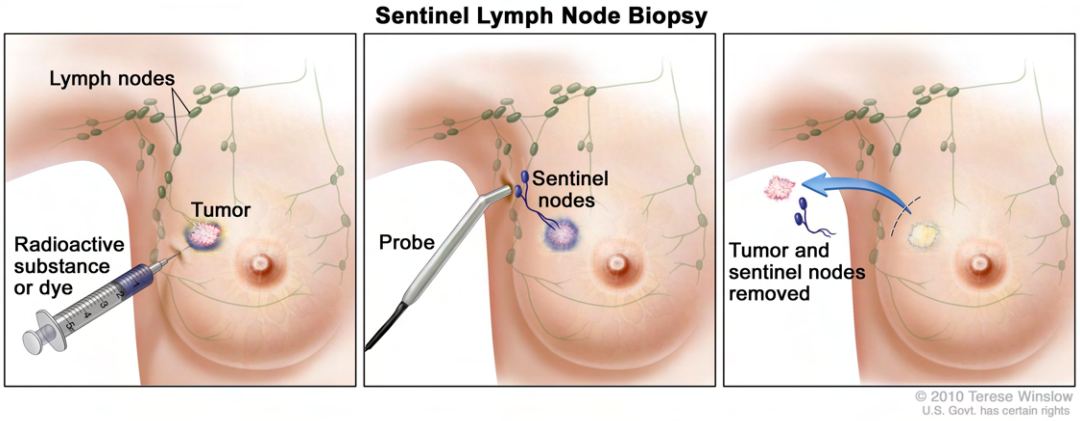
Breast cancer is likely to form in the ducts and lobes, with nearby axillary lymph nodes (small glands that filter lymph fluid through the lymphatic system) in the underarm being the first place breast cancer is likely to spread (see Figure 2.1) (Cascetta, 2021). Thus, metastatic involvement of lymph nodes is one of the most important prognostic variables in breast cancer. It is also an important predictive factor for the presence of distant metastases as the lymphatic system distributes across the human body.

Figure 2.1: Anatomy of the female breast including regions where cancer is likely to form and spread ("Cancer Information Summaries (PDQ)", 2002).



L. J. Martin (2020) outlines the tissue extraction process needed for pathological analysis. Patients undergo sentinel node biopsy, a method that ensures only lymph nodes most likely to have cancer are removed (see Figure 2.2). This minimises the invasiveness of lymph node extraction. A tracer (e.g., radioactive substance or dye) is injected into the area around a tumour to identify the lymph nodes the tumour drains into, which are referred to as sentinel lymph nodes. One to three sentinel nodes are usually removed to be tested for cancer. Then, fixation is carried out on the extracted sentinel nodes; the nodes are processed to make glass slides that hold tissue sections just a few micrometres thick (3 to 5 slides per lymph node) (L. J. Martin, 2020). Typically, stains are added to the glass slides to make the tissue cells more visible (the cells would otherwise be transparent) (Litjens et al., 2018). Stains usually employ immuno-histochemistry to highlight the presence of proteins normally found in breast cancer cells and not in lymph nodes, which aims to improve pathologist sensitivity. However, immuno-histochemistry testing of sentinel lymph nodes increases cost and slide preparation time. Even with immuno-histochemistry stained slides, the identification of small cancer metastases can be tedious and inaccurate. The hematoxylin and eosin (H&E) stain is most widely used across many different tissue types, inducing sharp blue and pink contrasts across various (sub)cellular structures (Litjens et al., 2018).

Figure 2.2: Illustration of the sentinel lymph node biopsy procedure ("Sentinel Lymph Node Biopsy", 2019).



Conventionally, the diagnostic procedure involves the glass slides being inspected by a pathologist (a doctor who diagnoses and studies disease involving cells and tissue samples) under a microscope to determine whether metastases are present (Litjens et al., 2018). The task consists of the localisation and identification of small lesions in the full slide space. There are three distinct categories of breast cancer lesion a pathologist seeks to identify (see Table 2.1): isolated tumour cells (ITCs), micro-metastases, and macro-metastases (Litjens et al., 2018).

Table 2.1: Types of metastases and their corresponding size information (Litjens et al., 2018).

Metastases Type	**Size Information**
Individual Tumour Cells (ITCs)	Single tumour cells or a cluster of tumour cells not larger than 0.2mm or less than 200 cells.
Micro-metastases	Larger than 0.2mm and/or containing more than 200 cells, but not larger than 2mm.
Macro-metastases	Larger than 2 mm.

In conventional breast cancer diagnosis, diagnostic accuracy is dependent on a pathologist's experience, which is built up over many years of observations of different patient tissue samples and confirmed diagnoses (Yue et al., 2018). However, high diagnostic accuracy cannot be guaranteed (Yue et al., 2018). The most senior of pathologists can miss metastases, especially in the case of micro-metastases and isolated tumour cells, which are difficult to detect given the large area of tissue that has to be examined per slide (van Diest et al., 2010). Even if pathologists were perfectly accurate, the task is time consuming and labour intensive, requiring large amounts of reading time through extensive microscope assessment (Litjens et al., 2017). For example, in the *Camelyon-16* challenge, a pathologist without time constraints required roughly 30 hours to exhaustively analyse 129 slides and missed 27.6% of metastases (Bejnordi et al., 2017). Furthermore, the imposition of time constraints to simulate clinical requirements resulted in a panel of 11 pathologists missing a mean of 37.2% of metastases (median 120 minutes to analyse the same 129 slides) (Bejnordi et al., 2017). In the majority of cases, pathologists are investigating benign tissue (i.e., absence of metastases); 60–70% of sentinel lymph nodes do not contain any metastases (Kim et al., 2006). Thus, a disadvantageous proportion of a pathologist's workload is spent simply confirming negative cases. When metastases are identified, the tissue samples are prone to misinterpretation and may require repeat assessments involving multiple pathologists ("Camelyon-16", 2015). A retrospective study showed that review of slide diagnoses by pathology experts changed the nodal status in 24% of patients (Vestjens et al., 2012).

2.2 Automated Histopathological Breast Cancer Detection

2.2.1 Towards The State-Of-The-Art

Automated breast cancer detection using histopathology slides is a research area within the more general field of digital pathology - the study of disease in tissue specimens using virtual microscopy.

Pathology slides have been digitised since before whole-slide imaging scanners became commercially available in the 1990s. Pathologists could create partial-slide images containing regions of interest (Pantanowitz, 2010). Computer-aided diagnosis emerged as a result, where digital tools are utilised to aid the diagnostic procedure. However, cancer detection was still a completely manual task carried out by pathologists. In fact, computer-aided diagnosis was rarely adopted into the regular workflows of pathologists due to issues with the digital nature of the slide images, such as colour variations in monitors unnecessarily complicating assessments (Griffin & Treanor, 2017).

The emergence of digital pathology facilitated new medical procedures, such as telepathology in the 1980s. In telepathology, digital slide images are communicated between labs for remote pathology diagnosis (Griffin & Treanor, 2017). Several advantages arise from this practice: low resource labs can employ the expertise of top (international) academic centres, second opinions can be more easily requested via remote consults, and physical shipping/storage of fragile glass slides is mitigated (Madabhushi &

Lee, 2016). Telepathology is especially beneficial where remote diagnosis is essential, such as with intra-operative frozen section procedures (Griffin & Treanor, 2017). Intra-operative frozen section procedures are used for rapidly preparing histo-pathology slides for inspection during surgery. In breast cancer diagnosis, this assists in determining whether further lymph nodes need to be extracted - a crucial decision in breast conserving surgery (i.e., lumpectomy or partial mastectomy) (Jorns et al., 2012). Diagnosis in such procedures is manually conducted by a remote pathologist; thus, digital pathology should be distinguished from automated pathology.

As computational power and medical data became more available for research, machine learning techniques were applied for automated cancer detection (Litjens et al., 2017). These techniques typically involved human engineers collaborating with pathologists to determine hand-crafted features about the problem domain (e.g., mitoses, uniformity of cell size, uniformity of cell shape) (Litjens et al., 2018). The features are manually decided, but their association to the predictions is learnt automatically. This marked a shift in breast cancer diagnosis from manual human-based diagnostic systems to automated systems that are trained by computers using human-driven hand-crafted features (Litjens et al., 2017; Madabhushi & Lee, 2016). Such systems could mitigate subjectivity and variability in pathology diagnosis (Griffin & Treanor, 2017). The use of hand-crafted features tends to provide prediction transparency and, hence, are intuitive to a pathologist or clinician. This is vital for diagnosis as decision accountability is required (Madabhushi & Lee, 2016). However, deriving such hand-crafted features is challenging since this involves a fundamental understanding of the nature of the disease and its manifestation within tissue (Madabhushi & Lee, 2016). This issue drives research in deep learning techniques, which involve automatic feature discovery (investigated in subsection 2.2.5).

2.2.2 Machine Learning Tasks

Machine learning techniques mainly perform classification or regression (Russell & Norvig, 2020). Fundamentally, classification involves predicting some label, whilst regression involves predicting some quantity. Classification is more pertinent to automated breast cancer diagnosis; metastasis detection requires qualifying, not quantifying.

Classification can be further separated into two forms that concern this project: whole-slide classification and tumour localisation ("Camelyon-16", 2015).

Whole-slide classification involves the slide-level prediction of metastasis, either with binary labels (i.e., metastasis or normal), or with multi-class labels (i.e., no cancer, isolated tumour cells, micro-metastases, or macro-metastases) (Litjens et al., 2018).

Tumour localisation refers to identifying and segmenting metastases within a slide (Litjens et al., 2018). This typically involves pixel-wise or patch-based classification, where the status of every pixel (or groups of pixels) in an image is predicted. Training for this task typically experiences a skewed class imbalance as most pixels/patches usually correspond to a non-object class (i.e., no cancer). This is an issue as these non-object samples are easy to discriminate, preventing the learning process from prioritising the minority of challenging training samples (i.e., isolated tumour cells and micro-metastasis) (Litjens et al., 2017).

There are other machine learning tasks that are useful in the context of pathology image processing, such as content-based image retrieval (Akakin & Gurcan, 2012) and image generation/enhancement (Abraham et al., 2021). These tasks will not be explored by this project as they do not directly relate to automated breast cancer detection.

2.2.3 Machine Learning Algorithm Types

Machine learning algorithms mostly correspond to two main categories describing the learning process: supervised and unsupervised learning (Litjens et al., 2017; Russell & Norvig, 2020).

In supervised learning, a model is trained for the problem using a data set of input features with corresponding expected outputs. The expected outputs can take several forms depending on the type of task being performed, such as binary and multi-class classification (Litjens et al., 2017; Russell & Norvig, 2020). Typically, supervised training involves finding a set of (optimal) model parameters that best predict the data by minimising some loss function that relates the model outputs to the expected outputs (Yue et al., 2018).

In unsupervised learning, a model is trained using a data set of input features where no such corresponding outputs are given (i.e., the data set is unlabelled). Unsupervised learning seeks to find patterns in the data, as seen in component analysis and clustering (Litjens et al., 2017; Russell & Norvig, 2020).

Whilst unsupervised learning may allow for the existing wealth of unlabelled medical data to be leveraged, this project exclusively explores supervised learning. The aim of this project is to improve the clinical viability of existing state-of-the-art methods for breast cancer detection. Supervised approaches have been greatly explored in this field, and the existence of the *Camelyon* challenges permit meaningful comparisons to these approaches, which is not the case with unsupervised learning research.

Additionally, two other nuanced machine learning algorithm types are semi-supervised learning (Zhu & Goldberg, 2009) and reinforcement learning (Sutton & Barto, 2018). Whilst these are growing research areas, most machine learning algorithms for automated breast cancer diagnosis are supervised (Litjens et al., 2017).

2.2.4 Whole-Slide Images And The *Camelyon* Challenge

Advances in technology have resulted in the creation of rapid slide scanners capable of digitising glass slides (containing tissue specimens) into whole-slide images (WSIs) at high resolutions (e.g. 240nm per pixel) (Litjens et al., 2018). Typically, WSIs at the highest resolution are hundreds of thousands of pixels in width and height (size is in the order of gigapixels), with each pixel requiring 3 bytes for the RGB channels ("Camelyon-16", 2015; Litjens et al., 2018). To effectively manage the storage requirements of such images, WSIs are generally stored in a multi-resolution pyramid structure (see Figure 2.3) ("Camelyon-16", 2015). A lower magnification view of a glass slide (using a microscope) corresponds to a lower-resolution image, created by down-sampling the high-resolution WSI (the base of the pyramid structure) (see Table 2.3). Additionally, each image in the pyramid is stored as a series of tiles, which facilitates the rapid retrieval of sub-regions in the image.

Figure 2.3: Typical pyramid structure of whole-slide images, where lower resolutions correspond to lower magnification views of the slide ("Camelyon-16", 2015, "Data - Visualizing the images and annotations").

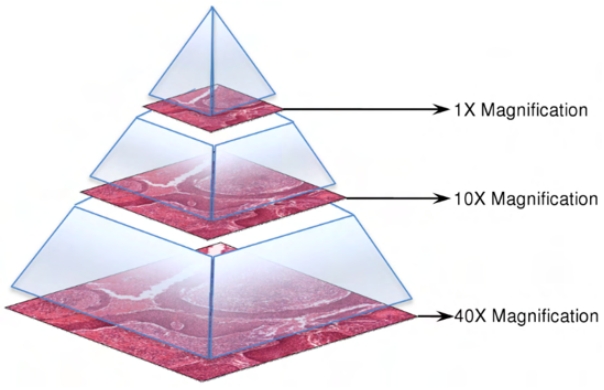


Table 2.3: Table relating WSI resolution levels to their corresponding area per pixel (Bejnordi et al., 2017). Level 8 represents the lowest resolution (i.e., thumbnail view), and level 0 is the highest resolution; a lower resolution means each pixel represents a larger area of tissue.

Resolution Level	**Area Per Pixel (μm^2)**
8	62.2
7	31.1
6	15.6
5	7.8
4	3.9
3	1.94
2	0.97
1	0.49
0	0.24

The quantity of data present with WSIs makes them suited for analysis using machine learning algorithms (Litjens et al., 2018). An increased availability of WSI data sets, as well as recent improvements in GPUs and GPU acceleration libraries, has made automated digital pathology a popular research area for deep learning techniques (Litjens et al., 2017). Specifically, this research is focused on three domains: (1) nuclei detection, segmentation, and classification, (2) large organ segmentation, and (3) disease detection and classification at the lesion- or slide-level. This project lies in the third domain, by applying machine learning techniques to low-resolution WSIs for tumour localisation and whole-slide classification. However, this deep learning research is bottle-necked by the need for the WSIs to be labelled, which requires laborious annotating by pathology experts.

The development of these deep learning techniques has been facilitated by recent online challenges regarding medical image analysis. The most pertinent of these challenges are *Camelyon* ("Camelyon-16", 2015; "Camelyon-17", 2016) and *Tupac* (Veta et al., 2019), which relate to processing breast cancer tissue samples. *Camelyon* (*CAncer MEtastases in LYmph nOdes challeNge*) was the first challenge providing data sets consisting of annotated WSIs (Litjens et al., 2017). These online challenges have enabled meaningful comparison between developed techniques by providing a platform for equivalent evaluation. In this project, the *Camelyon* data sets are used. The corresponding challenge results provide a baseline of both machine learning model performance and actual pathologist performance by which a developed low-resolution model can be compared against.

2.2.5 Convolutional Neural Networks And Deep Learning

The state-of-the-art has advanced automated breast cancer detection in histo-pathology further; current systems allow computers to learn not just the associations between features and outputs as before, but also the features themselves. This is the fundamental basis of deep learning techniques, which pervades current research (Litjens et al., 2017).

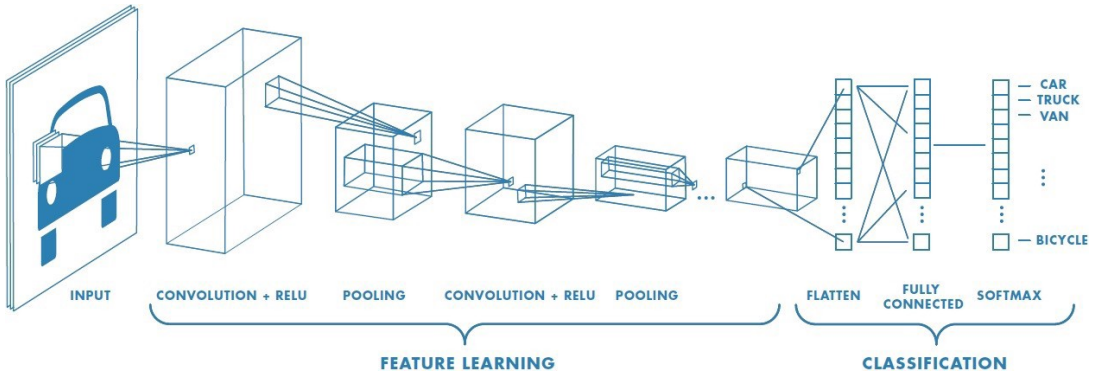
The most popular deep learning architectures are convolutional neural networks (CNNs) and recurrent neural networks (RNNs), though CNNs are currently the most widely used in medical image analysis (Litjens et al., 2017).

CNNs have been researched since the late 1970s (Fukushima, 1980) and were applied to medical image analysis as early as 1995 (Lo et al., 1995). The first real-world application of CNNs occurred for hand-written digit recognition with 'LeNet' (LeCun, 1998). However, the use of CNNs did not gather momentum until the development of various techniques for efficiently training deep neural networks.

An overview of CNNs is given by Russell and Norvig (2020). CNNs consist of many layers, which extend artificial neural networks to include convolutional and pooling layers (see Figure 2.4). Each layer can perform feature extraction or classification. As each layer feeds output into the next layer, features can hierarchically become more complex (Litjens et al., 2018). For example, the first layers can identify edges

and subsequent layers can combine these into more meaningful objects, eventually leading to complex structures like lesions. A key component of CNNs is that they are partially connected; neurons in one layer may connect to a subset of neurons in the next layer (Russell & Norvig, 2020). These neurons are connected such that they model a convolution operation, which allows recognition of a single feature (a convolutional filter) across the entire image, making CNNs highly efficient for image processing (Géron, 2019; Litjens et al., 2018). Conventional neural networks cannot provide such efficiency; a fully-connected neural network with only 10 neurons in the first layer would require ten million connections in the first layer alone for a 1,000 x 1,000 pixel input image. The number of connections required for a WSI, which has a size order in gigapixels, is far more excessive.

Figure 2.4: Example structure of a typical convolutional neural network (Saha, 2018).



However, deep learning approaches have been criticised for their dependency on large amounts of training data and lack of intuition and transparency with regards to their generated image features (Madabhushi & Lee, 2016). There have been efforts to find ways to relate transformed feature spaces to hand-crafted attributes (Ginsburg et al., 2015), which would improve the transparency of deep learning techniques and mitigate their 'black-box' nature that hinders clinical adoption. H. Wang et al. (2014) applied a convergent approach (for the detection of mitosis within breast cancer pathology images) to combat these issues. The approach combines domain inspired features with deep learning features and was shown to yield superior detection accuracy compared to deep learning or hand-crafted feature-based strategies alone (H. Wang et al., 2014).

2.3 Comparison Of *Camelyon-16* Techniques

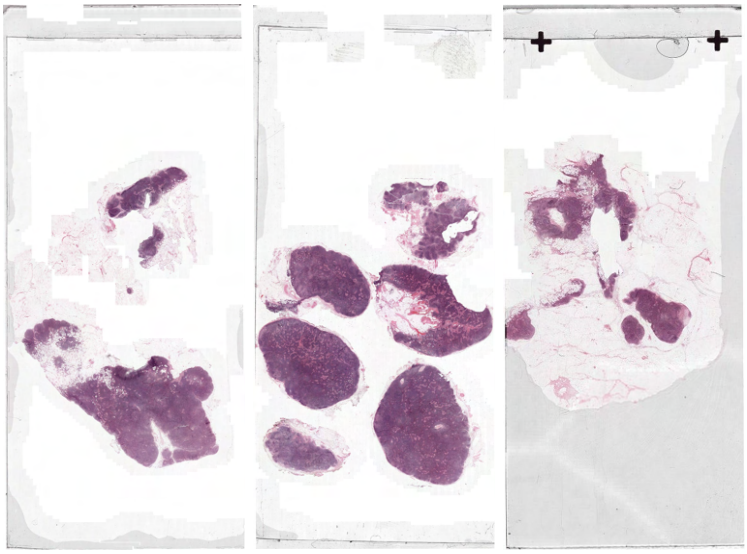
The following sections will investigate machine learning techniques, such as state-of-the-art deep learning methods, as they have been applied to the field of automated breast cancer detection in pathology slides within the *Camelyon-16* challenge ("Camelyon-16", 2015).

To facilitate discussion of the submissions to the *Camelyon-16* challenge, a table has been provided (see section 8.2) relating team pseudonyms (as they will be used in this report) with their full challenge team names and citations. The following sections derive considerable knowledge of the submission systems from the diagnostic assessment of the challenge (and supplementary material) carried out by Bejnordi et al. (2017), as well as the individual submission papers, where available.

2.3.1 Pre-Processing

The *Camelyon-16* challenge submissions carried out several pre-processing steps on the input WSIs (examples in Figure 2.5) to aid training of the machine learning models. Namely, pre-processing corresponded to the detection of tissue regions within the input slides, and the standardisation of H&E staining applied to the tissue.

Figure 2.5: The following are example WSIs ('*normal_011.png*', '*normal_076.png*', and '*tumor_001.png*'), which serve to provide context for the pre-processing steps discussed in the following sections.



2.3.1.1 Tissue Detection

A key pre-processing step carried out by all submissions was the detection of the H&E-stained tissue within a WSI and separation of such tissue from the (white) background. The background contains no valuable training information, so its removal improves efficiency. Most teams used a threshold-based segmentation method to detect foreground (i.e., tissue) regions, which creates tissue region masks to apply over the original WSIs.

Some teams ('*Minsk*', '*NLPLogix*', '*CULab*', etc.) used colour thresholding with some empirically or intuitively chosen threshold value to detect tissue. For example, the '*ExB*' team checked if the mean difference between the different RGB channels was lower than some threshold. As the background of the WSIs are represented by whiter colours, the background has similar values for each of the RGB colour channels (i.e., low differences between them). Thus, the foreground is segmented easily, though this assumes staining is robust across the tissue regions in all the WSIs. The '*HTW-Berlin*' team simplified this further by checking only the difference between the red and green colour channels, presumably since the blue colour channel is prominent across every WSI (in the pink/blue contrasts generated by H&E staining and the white background).

High performing teams, such as the top performing '*HMS-MIT*' team (for both tasks), used the Otsu (1979) thresholding algorithm to automatically determine the threshold that gives optimal segmentation. Otsu thresholding involves iterating through all possible threshold values and measuring the spread for the pixel levels on each side of the threshold (i.e., the pixels falling in the foreground or background). The aim is to determine the threshold value where the sum of foreground and background spreads is minimal (i.e., the weighted variance between the foreground and background pixels is the least) (Otsu, 1979). In the cases where Otsu thresholding was used, some submissions (e.g., '*METU*') also employed median filtering to remove unwanted distortions generated by the algorithm.

Additionally, many teams performed connected component analysis to eliminate small noisy areas and better isolate the tissue regions. The '*Minsk*' and '*Tampere*' teams also applied morphological operations to the resulting threshold images to remove spurious tissue regions. However, this step appears non-crucial; neither of these teams achieved particularly good performance having implemented morphological operations (other teams did not and achieved better performance). Of course, such comparisons are not so simple as the submission rankings indicate the performance of the entire submission systems, which differ at every stage.

Other than threshold-based segmentation, some teams applied machine learning techniques to the images for tissue detection; namely, '*Warwick-QU*' used a fully convolutional neural network,

'*SmartImaging*' used a support vector machine (Cortes & Vapnik, 1995), and '*LIB*' used k-means clustering.

2.3.1.2 Stain Standardisation

An invaluable pre-processing technique proved to be stain standardisation. The colour appearance of the WSIs is modified to resemble some reference sample, which aims to reduce appearance variability between digital slides.

The importance of color standardisation for CNN based tissue classification in H&E stained images was demonstrated by Sethi et al. (2016). Within the *Camelyon-16* challenge, two different slide scanners were used to digitise H&E stained slides from two different institutions (Bejnordi et al., 2017). The digitised slides from both institutions contain visible colour variations. The '*HMS-MIT*' team improved their whole-slide classification performance by adding a stain standardisation technique, the whole-slide image color standardiser (Bejnordi, Litjens, Timofeeva, et al., 2015), to assist with handling stain variations; their AUC score improved from 0.923 to 0.994 (Bejnordi et al., 2017). Other methods for stain standardisation were used with varying success: '*Warwick-QU*' and '*LIB*' used Reinhard staining normalisation (Reinhard et al., 2001), '*Tampere*' employed histogram matching, and other teams opted for custom techniques (e.g., '*SmartImaging*').

Alternatively, separate CNNs could be trained per institution to combat the stain variations, as opposed to training a single CNN on stain-standardised slides. This approach was attempted by the '*Minsk*' team. However, they generally achieved poor performance compared to teams that used a single classifier, regardless of whether stain standardisation was used or not. For example, '*HMS-MIT*' performed significantly better for both tasks even though both teams used 'GoogLeNet' architectures (Szegedy et al., 2015) and hard negative mining. Of course, stain standardisation is only partially responsible for the difference in performance between these teams (the methods of these teams differ in other aspects).

2.3.1.3 Pre-Processing Resolution Level

A key feature of relevance to this project is the resolution level used by submissions when pre-processing the WSIs. Submissions typically used multiple resolutions throughout their machine learning pipeline: WSI resolution for pre-processing, sample resolution for training, and probability map resolution for post-processing and classification.

Most available resolution levels (see Table 2.3) were used by multiple submissions when pre-processing the input WSIs, from the highest resolution at level 0 (e.g., '*NLPLogix*' and '*OsakaUniv*') to level 7 (e.g., '*Minsk*' and '*METU*'). The highest performing teams across both tasks ('*HMS-MIT*', '*HMS-MGH*' and '*CULab*') used a reasonably low resolution (level 5). Despite this, lower resolution pre-processing is typically mapped to a high resolution (e.g., with patch generation) for training and predictions with these teams. Thus, the models of these teams cannot be regarded as low resolution. Also, care needs to be taken when considering which levels are low and high resolution as the difference in area per pixel is non-linear between levels. Level 5 is considered reasonably low-resolution but is still $\sim$4 times larger than the next lowest resolution (level 6), and $\sim$16 times larger than the lowest resolution used by any submission (level 7).

2.3.2 Machine Learning Framework

For all submissions, the pre-processed WSIs were used to train machine learning models according to the machine learning framework designed by each submission. The output of these machine learning frameworks is identical across submissions: a probability map is produced for each input WSI, indicating the likelihood of metastases across the slide space. Key differences in machine learning framework between the submissions are explored in the following sections.

2.3.2.1 Network Architecture

Overall, submissions based on deep learning algorithms performed significantly better than other methods; the 19 top-performing algorithms for both whole-slide classification and tumour localisation all

used deep convolutional neural networks as the underlying methodology (Bejnordi et al., 2017). Some deep learning algorithms were able to achieve better performance than actual pathologists participating in a simulation exercise. The highest performing team for the whole-slide classification task, '*HMS-MIT*', achieved an AUC score of 0.994 compared to a mean AUC score of 0.810 for the time-constrained pathologists. The '*HMS-MIT*' performance is comparable to the pathologist without time constraints, who achieved an AUC score of 0.966 (Bejnordi et al., 2017).

Other than deep learning, some hand-crafted feature techniques were used. For these techniques, different statistical and structural texture features were extracted, such as colour scale-invariant feature transform (SIFT) descriptors (Lowe, 2004), local binary patterns (Ojala et al., 2002), histogram of gradients (Dalal & Triggs, 2005), and grey-level co-occurrence features (Haralick et al., 1973). These were combined with widely used supervised classifiers, such as support vector networks (Cortes & Vapnik, 1995) and random forest classifiers (Breiman, 2001). However, algorithms based on manually engineered features performed less well (Bejnordi et al., 2017); the best submission of this class ('*Tampere* I') used random forests and ranked 20^{th} for whole-slide classification.

The common component for submissions achieving the highest performance was the use of CNNs (Bejnordi et al., 2017). The majority of submitted algorithms (25 out of 32) were based on deep CNNs. The top 10 CNN submissions exclusively used three types of architecture: GoogLeNet (Szegedy et al., 2015), ResNet (He et al., 2016), and VGG-Net (Simonyan & Zisserman, 2014). The suitability of these network architectures for whole-slide classification and tumour localisation is established by their general success in the '*ImageNet Large Scale Visual Recognition Challenge*'. GoogLeNet won the object detection and classification tasks in 2014, whilst VGG-Net won the localisation task (Russakovsky et al., 2015). ResNet won the object detection, localisation, and classification tasks in 2015 (He et al., 2016; Russakovsky et al., 2015).

Despite the use of deep CNN architectures, the ranking among teams using these techniques varied significantly, ranging from 1^{st} to 30^{th} (out of 32 submissions) for whole-slide classification. Instead, auxiliary strategies to improve system generalisation seemed more important (Bejnordi et al., 2017). This analysis that network architecture is of lesser importance is raised by Litjens et al. (2017) as a factor of deep learning techniques across medical image analysis in general: "one striking conclusion we can draw is that the exact architecture is not the most important determinant in getting a good solution...in challenges like the '*Kaggle Diabetic Retinopathy Challenge*', many researchers use the exact same architectures, the same type of networks, but have widely varying results".

Other than the three main architectures used by the top performing teams, submissions largely used: U-Net or SegNet (neural network architectures designed specifically for segmentation in biomedical images) (Kendall et al., 2015; Ronneberger et al., 2015), the older AlexNet (neural network architecture that was the winner of the ImageNet Large Scale Visual Recognition Challenge 2012 for the object detection, localisation and classification tasks) (Krizhevsky et al., 2012; Russakovsky et al., 2015), or a custom CNN architecture (or alternate classifier) which did not perform comparatively well.

2.3.2.2 Transfer Learning

Several submissions used transfer learning by fine-tuning pre-trained networks. In transfer learning, network weights are initialised when training for one task using the already established weights of a model pre-trained on a different task. This technique is commonly used when limited training data is available and allows for the abilities of a network trained on a larger data set to be leveraged. Thus, knowledge gathered by high-performing CNNs in general domains can be transferred to related domains, such as medical image analysis (Mormont et al., 2018).

The '*HMS-MGH*' and '*CULab*' teams made use of transfer learning and were amongst the 3 teams that gave the highest performance submissions for both tumour localisation and whole-slide classification. The CNNs used in systems '*HMS-MGH* III', '*HMS-MGH* II', and '*CULab* III' were ResNet-101, GoogLeNet-22 and VGG-Net-16, respectively. Each was initialised by weights from pre-trained networks and fine-tuned with the *Camelyon-16* challenge data. ResNet-101 was pre-trained on the '*MS-COCO*' data set (Lin et al., 2014) and the other two models were pre-trained on the large scale 1000-class '*ImageNet*' data set (Russakovsky et al., 2015).

For the *Camelyon-16* challenge, transfer learning is arguably beneficial as the associated data set contains just 270 labelled WSIs (Litjens et al., 2018). With the use of patch sampling and data augmentation, the number of training samples can be extended to the order of hundreds of thousands; however, this is significantly smaller than the millions of training samples provided by the '*MS-COCO*' and '*ImageNet*' data sets. Despite this, Liu et al. (2017) suggest that network pre-training does not significantly improve model performance; there may be too large a domain difference between pathology images and natural scenes, as given by '*ImageNet*' and '*MS-COCO*', leading to limited knowledge transferability. They concede that this conclusion may arise from their use of extensive patch sampling and data augmentation, resulting in accurate models without pre-training. Regardless, network pre-training is observed to also improve model convergence speed.

2.3.2.3 Patch Sampling

Given the enormity in resolution of the WSIs, most *Camelyon-16* submissions opted for patch-based training (i.e., groups of pixels are extracted from the WSIs and used as training samples). Key factors of patch-based training include patch size, resolution at which patches are sampled, sampling strategy, and data augmentation.

Patch size was generally either the default input size of the network architecture used (e.g., 224 x 224 pixels for standard GoogLeNet), or the maximum patch size available given GPU memory limitations.

Patch resolution refers to the WSI resolution level (see Table 2.3) that the patches are sampled from to generate training samples. Most teams, especially teams with high performance for both tasks (e.g., '*HMS-MIT*', '*HMS-MGH*' and'*CULab*'), used high resolution patches taken at levels 0 or 1. Despite this, teams using high resolutions performed variably well overall; some teams using lower resolution levels, like 5 or 6 (e.g., '*Camp-Tum*' and '*Tampere*'), were able to perform better than teams using a higher resolution. This suggests that using higher resolution patches does not innately provide better classification unless the model is designed to leverage the chosen resolution. It is also noted that no teams opted to use the lowest resolution levels available, 7 and 8.

Sampling strategy refers to the process for choosing samples from the WSIs and typically aims to address class imbalance, which is pervasive in patch-based training. In the *Camelyon-16* data set, 60% of the WSIs belong to the normal (i.e., negative) class, so there is a minor class imbalance (Litjens et al., 2018). However, sampling patches from within the WSIs results in a severe majority negative class as metastatic regions account for a small area of tissue within a WSI. Most submissions, including '*HMS-MIT*' (the top performing team for both tasks), used an equal number of positive and negative cases obtained uniformly and randomly from the WSIs. This reflects a desire to focus training on learning harder and infrequent positive samples to prevent missed metastases. Alternatively, some teams used more negative samples than positive samples. For example, '*HMS-MGH*' had 25% of samples as positive, whilst '*CULab*' had an even lower positive proportion at 5%. This reflects a desire to better identify negative cases and prevent false positives. Another contentious point between submissions was in how negative patches were sampled. Some teams took negative patches solely from negative slides, whilst other teams allowed for negative patches to be taken from non-metastatic regions in positive slides as well. Interestingly, '*Warwick-QU*' used spectral clustering to find visually distinct positive and negative patches, which aims to reduce the training redundancy that arises from using very similar training examples. Also, Liu et al. (2017) notes that some submissions pre-sample sets of patches from each slide (e.g., '*HMS-MIT*'), which limits the breadth of patches seen during training and biases slides containing more patches (both normal and metastatic).

Additionally, several submissions sought to improve their models through the use of hard negative mining to prevent false positives. Hard negative mining involves discovering negative samples that are non-trivial to distinguish from positive samples: for instance, taking the false positives of an initial classification round and feeding them into further training ('*HMS-MIT*', '*Camp-Tum*', etc.).

Most submissions sought to further extend the number of training samples through the use of data augmentation, which involves applying outcome-invariant transformations to the existing training samples to create new, plausible training samples. Extending the training data in this way is a form of regularisation, which aims to reduce over-fitting. Many invariances were exploited by the submissions,

with most teams utilising rotations, vertical flipping, and horizontal flipping of the tissue specimens (valid as tissue specimens do not have a canonical orientation). More rigorous data augmentation techniques were used by some teams, such as random Gaussian blurring ('*RadboudUMC*'), random cropping ('*HMS-MIT*', '*HMS-MGH*', etc.), and the addition of colour noise ('*HMS-MIT* II').

2.3.2.4 Ensemble CNNs

Some *Camelyon-16* submissions used an ensemble of CNNs in their machine learning pipeline, which involves combining the output from different models (e.g., by averaging the predictions of the different models) with the goal of improving overall performance.

Classifying small objects into classes, such as micro- and macro-metastasis, usually involves both local information on lesion appearance and global contextual information about lesion location for accurate classification (Litjens et al., 2017). Several authors have used multi-stream architectures (i.e., network ensembles) to handle these requirements in a multi-scale approach. For example, Kawahara and Hamarneh (2016) used a multi-stream CNN to classify skin lesions, where each stream used a different resolution of the input image. In the *Camelyon-16* challenge, similar techniques have been used. The '*UnivToronto*' team calculated predictions using a mean of results produced by different classifiers, each using a different resolution of the WSI. Furthermore, the high-performing '*HMS-MIT*' team used an ensemble of two CNNs for metastasis identification, one trained before and one trained after hard-negative mining. Liu et al. (2017) found that ensembles involving independently trained models yielded additional but smaller improvements, and gave diminishing returns after three models.

2.3.2.5 Hyper-Parameter Optimisation

Hyper-parameter optimisation (e.g., learning rate, regularisation strength, mini-batch size) is an important step to improve the performance of a model, though it is usually of secondary importance when compared to pre-processing strategies and training data quality (Litjens et al., 2017). There is no clearly defined process to perfectly optimise a model, so optimisation usually falls to engineer expertise (e.g., intuition-based random search) or empirical investigation (e.g., exhaustive grid search). Whilst Bejnordi et al. (2017) discusses hyper-parameter tuning as a contributor in the variable success of CNN models in the *Camelyon-16* challenge, the submissions fail to address specific strategies for performing the tuning. This is likely because hyper-parameter optimisation is an issue in the general domain of machine learning and is outwith the challenge context.

2.3.3 Post-Processing

Teams performed post-processing on the patch-based probability maps produced for each WSI (by their machine learning framework) to complete the *Camelyon-16* challenge objectives of per-slide tumour localisation and whole-slide classification. Differences in method between submissions are discussed in the following sections.

2.3.3.1 Tumour Localisation Task

The tumour localisation (i.e., metastasis identification) task requires identifying the coordinates of any metastatic regions within the WSIs and calculating a probability score for each identified region (Bejnordi et al., 2017).

All submissions used their main machine learning frameworks (random forest classifiers, CNNs, ensembles, etc.) to generate (from classified patches or tiles or pixels) probability maps for each WSI. Probability maps were generated by submissions at a range of resolution levels from as high as level 0 (e.g., '*ExB*') to as low as level 7 (e.g., '*Minsk*'). Amongst the submissions, tumour identification with the maps generally required three distinct steps: post-processing, lesion co-ordinate calculation, and probability score calculation.

Most teams applied thresholding to the probability map, which acts as a minimum confidence requirement for metastasis identification. As such, most teams opted for a high confidence threshold (no lower than 0.9). The '*Warwick-QU*' team differed in that they used multiple thresholds to generate masks

representing different levels of confidence. The '*NLPLogix*' team opted not to use simple thresholding, but a segmentation strategy using 'GrabCut' (Rother et al., 2004) and 'Watershed' (Meyer, 1994).

Other post-processing steps sought to improve the probability map depending on the intended co-ordinate and probability score calculation strategies. For example, '*RadboudUMC*' eroded the map, '*Camp-Tum*' applied Gaussian filtering, and many teams further filtered the map with morphological operations (e.g., '*Minsk*', '*QuincyWong*', and '*VisiLab*'). Most teams also applied connected component analysis to isolate metastatic tissue regions within the map (e.g., '*HMS-MIT*' and '*HMS-MGH*'), and some took this further by also removing small spurious regions according to some criteria. For example, '*HMS-MGH*' removed regions where the major-axis length was smaller than some measure, whilst '*Minsk*' and '*DeepCare*' simply eliminated regions smaller than some chosen amount of pixels. However, Liu et al. (2017) identifies that whilst a connected component approach can improve performance, this approach is sensitive to the connectivity threshold and can confound model evaluation (multiple nearby tumours can be grouped as one). Instead, Liu et al. (2017) employ a non-maxima suppression approach.

Methods for co-ordinate calculation varied according to which locales were considered representative of the identified lesions. Some teams, including the top performing '*HMS-MIT*' and '*HMS-MGH*' teams, simply assigned lesion co-ordinates as the gravitational centre of the identified lesions. Other teams (e.g., '*Minsk*') assigned lesion co-ordinates as the nearest point to the gravitational centre that sat on the morphological skeleton of the lesion. These teams argue that an identified region may have a complex shape, so its coordinates may not be defined simply as the centre of its bounding box or centre of gravity; in both cases the centre may lie outside of the detected region (e.g., C-shaped lesion).

Methods for lesion probability scoring differed according to the coordinate calculation strategy. All teams either took the probability score for each lesion as those given by the probability map at the lesion coordinate, or calculated a probability score over the metastatic region. Simple strategies, like calculating the mean probability over the lesion, or using the maximum probability within the lesion, were popular (used by '*HMS-MGH*', '*CULab*', etc.). The '*Warwick-QU*' team was more conservative in their probability estimates by using the minimum probability with each lesion, but weighted this using the lesion area. This scoring reflects increasing confidence with increasing lesion size, which was also used by '*QuincyWong*'; smaller lesions were given a probability score penalty, whilst larger lesions were given a bonus. Alternatively, some teams opted to go as far as using another classifier (e.g., random forest) to generate probability scores for each lesion. These classifiers used statistical summary features from the probability maps, such as identified lesion cluster size, area, orientation, major/minor-axis ratio, etc. ('*NLPLogix*' and '*Camp-Tum*').

2.3.3.2 Whole-Slide Classification Task

The whole-slide classification task involves assigning a probability score for each WSI, representing the confidence that the slide contains metastases (Bejnordi et al., 2017). Overall, solutions to this task fit into two categories: a simple approach is used based on the solution to the tumour localisation task, or a new classifier is trained.

In the former case, approaches are simple as they leverage processing already carried out to complete the localisation task. For example, many teams simply reported the probability score for each slide as the maximum probability of the identified lesions ('*CULab*', '*ExB*', '*Tampere*', etc.) or the probability of the largest lesion ('*Warwick-QU*'). More conservative scoring teams opted for calculating the mean probability of identified lesions. For example, '*Camp-Tum*' calculated the mean probability of the top three candidate lesions, whilst '*UnivSFlorida*' computed the probability score as the weighted arithmetic mean (with weights 3 and 1) of the two metastatic regions with the highest probability scores. Although these approaches worked well for many of the teams, including '*CULab* III' and '*ExB*' that were ranked 4^{th} and 6^{th} in the image classification task, these approaches may not take fully into account metastases characteristics (e.g., slides containing multiple high-score and/or larger identified metastases should have an increased confidence of containing metastases).

Many teams opted to train additional classifiers for whole-slide classification. Of these classifiers, some acted independently from the main machine learning framework (effectively meaning a different classifier was trained for each task), whilst others used features formed from the results of the localisation task.

Two of the top performing teams for whole-slide classification, '*HMS-MIT*' and '*HMS-MGH*', both used random forest classifiers (Breiman, 2001). They used local features calculated from the probability map generated by the localisation task, such as metastatic region area, eccentricity, major axis length, max/mean/min intensity, solidity, etc. However, '*HMS-MIT*' extended their set of features to include global features as well, such as (1) the ratio between the area of metastatic regions and the tissue area, and (2) the sum of all cancer metastases probabilities detected in the metastasis identification task, divided by the tissue area. As a result, '*HMS-MIT*' took the top ranking over '*HMS-MGH*', outperformed the mean performance of the pathologists with time constraints, and performed comparably to the pathologist without time constraints. Another popular choice of classifier was support vector machines (Cortes & Vapnik, 1995), which was used by '*VisiLab*', '*SmartImaging*', and '*DeepCare*'. These teams similarly used morphological and geometric features as seen with the random forest classifiers, but did not perform comparatively well. Interestingly, the '*RadboudUMC*' submission used the probability map from the localisation task but at a low resolution as input to a 15-layer VGG-like network. This submission was ranked 17^{th} out of 32 submissions.

2.3.4 *Camelyon-16* Techniques Summary

In summary, there are common components between submissions that underpin successful pipelines. When pre-processing, Otsu thresholding for tissue detection was most popular and was used by the top performing teams ('*HMS-MIT*', '*HMS-MGH*', etc.). Also, stain standardisation had a clear importance; this technique boosted the ranking of multiple teams when applied, including the top-performing '*HMS-MIT*' team. In terms of machine learning framework, deep CNNs were dominant, with few proven architectures being favoured amongst submissions: GoogLeNet, Res-Net, and VGG-Net. However, strategies for model robustness were identified as more important (Bejnordi et al., 2017), such as data augmentation, transferred learning, and hard negative mining. For tumour localisation, post-processing of the probability maps by applying thresholding and connected component analysis to identify lesions was pervasive. For whole-slide classification, secondary classifiers (with random forests appearing the best choice) using local and global features extracted from the probability maps gave the best performance.

2.4 Overview Of *Camelyon-17* Techniques

2.4.1 *Camelyon-17* Brief

Camelyon-17 ("Camelyon-17", 2016) is the successor to *Camelyon-16*, which aims to further the clinical viability of automated digital pathology techniques by focusing not just on single WSIs for slide-level classification, but on multiple WSIs per patient for patient-level pN-stage classification. Here, pN-staging (from the internationally accepted TNM system for classifying the extent of breast cancer spread in patients with a solid tumour) refers to whether cancer has spread to the regional lymph nodes.

The medical context of the *Camelyon-17* challenge extends that of *Camelyon-16*, so there is overlap between both challenges; both require the whole-slide classification of WSIs. The *Camelyon-17* challenge was re-opened in May 2017; only submissions accepted up to the last challenge review, conducted by Bandi et al. (2018), are considered. This review considers the submissions of 12 teams (see section 8.3).

2.4.2 Comparison Of *Camelyon-17* Techniques

All twelve teams followed the same pipeline steps: pre-processing, slide-level classification and post-processing, and patient-level classification. The latter step is not relevant to this project.

In pre-processing, all teams performed tissue detection in the WSIs using simple filtering and thresholding algorithms, with Otsu adaptive thresholding at a lower resolution level being most popular. Method differences were primarily found in which colour space the thresholds were applied (RGB, HSV, etc.), and the type of morphological operations that were used to refine the thresholded image. These techniques were similarly observed with *Camelyon-16*. However, *Camelyon-17* teams also had to handle differing slide background colours (black or white) when performing tissue detection, which was caused by the expansion of the challenge data sets to include slides from 5 medical centres (*Camelyon-16* involved 2 centres).

To perform slide-level metastases detection, all teams trained CNNs with image tiles extracted from normal and metastatic tissue regions (i.e., negative and positive samples) to obtain metastasis probability maps. Almost all participants used the *Camelyon-16* WSIs, as well as the 50 exhaustively annotated WSIs and all negative WSIs from the expanded *Camelyon-17* data set. Some teams exclusively used the *Camelyon-16* data set, with the best ranking 4th ('*MIL-GPAT*').

Variants of proven network architectures were used: ResNet, GoogLeNet, VGG-Net, U-Net, and one team used DenseNet. Five of the teams used model ensembles but only 2 teams ('*MIL-GPAT*' and '*ML-KA*') used fundamentally different networks in their ensembles. For example, '*MIL-GPAT*' combined two GoogLeNets with different input patch sizes and a ResNet-50 architecture. Eight teams used pre-trained networks (i.e., transfer learning), all pre-trained on the '*ImageNet*' challenge, except team '*HMS-MGH-CCDS*' who used a network that had been pre-trained on '*MS-COCO*'. These techniques have been continued from *Camelyon-16*.

Almost all teams performed extensive data augmentation; only team '*Chengshenghua*' (ranking 6th) did not use any data augmentation. Random mirroring and rotations were most popular, with other strategies including random cropping and applying affine transformations. To improve CNN robustness to colour variation caused by scanner differences and/or staining protocols (from 5 medical centres), most participants used colour augmentation by adding noise to the colour channels. Some teams also adjusted brightness, contrast and gamma. Colour augmentation approaches are somewhat novel when compared to the analysed *Camelyon-16* techniques. Just two teams tried to use stain normalisation algorithms to ensure a uniform color distribution across the images, as seen with *Camelyon-16*, even though stain variation is a more pervasive issue in *Camelyon-17*. Liu et al. (2017) demonstrated that stain normalisation is unnecessary when extensive colour augmentation of the training samples is used as the models learn color-insensitive features.

In slide-level post-processing, most participants thresholded the produced probability map and collected several features from identified lesions as input to a separate classifier to determine the WSI class (normal, micro-metastases, etc.). In *Camelyon-16*, many submissions suffered from high false positive rates (Bejnordi et al., 2017). The winner in *Camelyon-16* solved this by extracting features from the probability map and feeding these features to a random forest classifier. This approach was replicated by several *Camelyon-17* teams.

Interestingly, team '*VCA-TUe*' used test time augmentations to generate the probability map. Test time augmentation involves applying training augmentations to patches when testing to get multiple metastasis probabilities per patch. Often these are averaged to obtain a final patch probability, however, team '*VCA-TUe*' used the most certain likelihood (i.e. closest to 1.0).

In summary, most *Camelyon-17* submissions mimic the trialled and tested techniques implemented by the top performing teams of *Camelyon-16*. Novelty in approaches that separate submissions from those of *Camelyon-16* concern the use of additional training samples provided by the *Camelyon-17* challenge, colour augmentations to handle stain variability (as opposed to stain standardisation), and probability map generation via test time augmentations.

2.5 Low-Resolution Techniques In Medical Image Analysis

As discussed in section 2.3 and section 2.4 (regarding *Camelyon*), automated cancer detection approaches mainly used single-resolution models involving high-resolution images (Bejnordi et al., 2017). These models were able to make high confidence predictions for tumour localisation and whole-slide classification at the expense of model resource requirements. For the *Camelyon* challenges, such approaches focused on what models were possible, by seeking to maximise localisation and classification accuracy, as opposed to applicable in a real clinical setting. Inference time for these models is significant, making clinical adoption unlikely; a method reproduction for the top-performing *HMS-MIT* team found that their high-resolution method requires roughly 6 minutes and 35 seconds per high-resolution WSI on average.[1] However, recent research in medical image analysis has pivoted (especially since the

[1]The reproduced *HMS-MIT* method was developed by the *iCAIRD* team (of Christina Fell, Mahnaz Mohammadi and David Morrison) at the University of St Andrews.

culmination of the *Camelyon* challenges), placing greater emphasis on the clinical viability of developed techniques.

One such factor for improving clinical viability involves enhancing efficiency by lowering WSI input resolution, which is the focus of this project. Developing effective models that utilise low-resolution images can potentially yield high performance results with low resource requirements (compared to high-resolution models). A low-resolution model for the *Camelyon-16* challenge will likely reduce inference time compared to current patch-based approaches, which are used in almost all submissions (Bejnordi et al., 2017), by alleviating the need to process a high-resolution image into a multitude of patches for tumour region predictions. Liu et al. (2017) notes that each slide includes 10,000 to 400,000 (median 90,000) patches which have to be processed in a patch-based approach.

To this end, current medical image analysis research involving low-resolution image processing fits two main categories. The first is the development of single-resolution models using low-resolution images (the opposite of most *Camelyon* submissions). The second is the development of multi-scale schemes, where low- and high-resolution analysis is combined. Research is richer in the latter category than the former; few attempts have been made to investigate (at length) the ability of models using strictly low resolution images for medical image analysis tasks (this project is amongst the first). There is a somewhat obvious reason for the disparity in research between the two categories; single-resolution models using low-resolution images forgo the advantages of using high-resolution images (i.e., detailed features based on greater spatial resolution), whilst multi-resolution schemes compromise between the two, prioritising both model performance and resource requirements.

2.5.1 Multi-Scale Techniques

There are two forms of multi-scale techniques observed in the literature.

The first form is in the use of CNN ensembles, where WSIs are processed at multiple resolutions by different models and their results are aggregated. This approach seeks to improve model performance compared to single-resolution models by utilising the contextual features gained by lower resolutions (e.g., inter-tissue structures) in conjunction with detailed features gained by higher resolutions (e.g., cell morphology and intra-tissue structures). Efficiency improvement is not a motivation of this approach, since inference utilises multiple models involving higher resolution images.

For example, Tokunaga et al. (2019) presented a segmentation method that can adaptively use image features from different resolutions to segment multiple cancer sub-type regions in an input image. The method aggregates several segmentation CNNs (i.e., an ensemble), which are trained using different image magnifications (5x, 10x, and 20x magnification), by adaptively changing the weight of each expert depending on the input image (using a 'weighting CNN'). The method leverages features attained from both wide and narrow field-of-view images that may be useful for identifying the sub-types. The method was shown to give better performance for multiple segmentation tasks (binary segmentation of cancerous tissue and multi-class segmentation of tissue sub-types) when compared to state-of-the-art networks (UNet, SegNet, DeepLabV3+) trained at each of the magnification levels considered (Tokunaga et al., 2019).

The second form of multi-scale approach is in the use of a mechanism by which low resolution image processing is prioritised, but low confidence results permit further high-resolution processing. This emulates the methodology of a pathologist by first investigating tissue at a low magnification before zooming in on regions of interest in obscure and difficult cases. Conventional patch-based processing methods do not reflect this highly efficient manner in which pathologists navigate slides in clinical microscopy. This type of multi-scale approach seeks to achieve comparable performance to ensemble or single-resolution approaches involving high-resolution images, whilst reducing inference time on average (most cases can be resolved at lower resolutions).

For example, Bejnordi, Litjens, Hermsen, et al. (2015) presented a multi-scale approach for the automatic detection of regions of interest (e.g., background, stroma, epithelial nuclei) in breast tissue WSIs by classifying super-pixels at multiple resolutions. Super-pixels are perceptually meaningful atomic

regions that aggregate visually homogeneous pixels while respecting object boundaries. They are an alternate visual primitive to pixels/patches. As boundaries are respected during partitioning of the image into super-pixels, more accurate segmentation results can be attained by allocating super-pixels to the appropriate target class. The approach significantly reduces inference time by classifying low resolution super-pixels, which has the least computational burden. Higher resolution super-pixels are only generated and classified when classification at low resolution yields insufficient confidence. The approach gave a good overall performance (AUC score of 0.958) with an independent test set involving 10 WSIs (Bejnordi, Litjens, Hermsen, et al., 2015).

Maksoud et al. (2020) presented a method for the selective use of high-resolution processing based on prediction confidence at low-resolutions. Their system involves the use of a decision protocol to determine whether a low-resolution versus high-resolution network should be utilised. Their system was applied to 'liver-kidney-stomach' WSIs, where they were able to improve multi-class classification accuracy, whilst reducing inference time by a factor of 7.74 (when compared to other multi-scale approaches) (Maksoud et al., 2020). This approach is said to build on similar prior work carried out by Dong et al. (2018), where breast cancer segmentation is carried out in WSIs using low- or high-resolution images, as determined by a trained policy network (via reinforcement learning) which decides whether zooming in on regions of interest is required (Dong et al., 2018).

A very different multi-scale approach is presented by Hering and Kybic (2020) to detect cancerous tissue in histological images (a subset of the *Camelyon-16* data set). In their approach, several small regions of interest, which they call 'glimpses', are processed as opposed to the whole high-resolution WSI. The glimpses form a tree structure; glimpses at a low resolution determine the location of several higher resolution glimpses, continuing hierarchically until the highest resolution is reached. At the highest resolution, glimpses are classified and a whole-slide classification score is calculated (the maximum prediction score assigned to the highest resolution glimpses). They reported a mean AUC score between 0.92 and 0.95 (depending on their model parameters), which is comparable to the best AUC score (0.96) of the top performing Camelyon-16 team (Bejnordi et al., 2017; D. Wang et al., 2016). Their model inference time was between 1-1.5 seconds. This demonstrated that it is possible to perform high confidence classification having processed only a small proportion of the high-resolution image, leading to an important speedup with only a small performance deterioration (Hering & Kybic, 2020).

2.5.2 Single-Resolution, Low-Resolution Techniques

In terms of single-resolution models using low-resolution images, the following recent approaches are notable:

Patil et al. (2020) presented a CNN-based architecture for segmenting invasive breast cancer. The approach uses very low-resolution WSIs (320 x 320 pixels). The model is said to have achieved high accuracy when the cancerous regions took up a large proportion of the very low-resolution WSI, but performed significantly worse for smaller cancerous regions (Patil et al., 2020). This indicates that for this project, a developed low-resolution model may perform well on macro-metastases, which can be reasonably large in proportion to overall tissue, but give an unsatisfactory performance at localising micro-metastases.

Zormpas-Petridis et al. (2021) presented a computationally efficient framework ('*SuperHistopath*') designed to map global context features (by classifying and segmenting such features), reflecting the rich tumour morphological heterogeneity in WSIs (e.g., tumour tissue, stroma, necrosis, lymphocytes clusters, fat, haemorrhage and normal tissue). The approach involves the segmentation of low-resolution WSIs (at 5x magnification) into super-pixels, followed by classification of the super-pixels using a CNN ('Xception' and a custom CNN). It was shown that '*SuperHistopath*' is efficient for training and inference (∼5 minutes for classifying a WSI, and as low as ∼30 minutes for network training). For breast cancer WSIs, '*SuperHistopath*' classified sample regions into 6 predefined tissue categories of interest (tumour, necrosis, stroma, cluster of lymphocytes, fat, and lumen/empty space) with an overall accuracy of 93.1%, an average precision of 93.9%, and an average recall of 93.6% using 'Xception' (and 91.7%, 92.5%, 91.8% respectively using the custom CNN) over 10,349 super-pixels in an independent test set of five images (Zormpas-Petridis et al., 2021).

Finally, Shahidi (2021) presented a novel super-resolution approach using a generative adversarial network to generate super-resolution images from low-resolution WSIs. Since the hardware and storage requirements of WSIs are great, this paper is motivated by the idea of being able to store WSIs at low resolution with the intention that super-resolution techniques can be applied to achieve classification performance on the super-resolution images comparable to performance using the original high-resolution versions. Despite this, the main factor of relevance from this work concerns the performance comparison of controlled models (ResNeXt-101) using low- (64 x 64 pixel patches) and high-resolution (256 x 256 pixel patches) images for tumour classification. The high-resolution images gave better performance (99.49% accuracy score), but not dramatically so compared to low-resolution images (95.82% accuracy score) (Shahidi, 2021).

2.5.3 Low-Resolution Techniques Summary

In summary, single-resolution models using low-resolution images have seen some success for medical image analysis tasks. However, this does not indicate that the low-resolution model developed in this project will also give high confidence predictions; the differences in medical tasks being performed is a key factor. For example, a low-resolution model will be more capable at segmenting large tissue sub-types (especially when there is significant heterogeneity), than segmenting smaller homogeneous lesions (as is the case with the *Camelyon* data sets). Otherwise, research is focused on multi-scale approaches that use low-resolution images by default, but can resort to high-resolution processing when required. Since such approaches are designed to reduce inference time (on average) whilst maintaining high network capability, one may question the need for strictly low-resolution image models. However, it may be the case that strictly low-resolution models are sufficient for certain medical tasks, alleviating the hardware and storage requirements associated with maintaining high-resolution WSIs. Even when low-resolution model performance is inferior to multi-scale approaches, the low-resolution model may be exceptional for a certain class, warranting use. For example, a low-resolution model that exceptionally classifies benign tissue can filter the majority of negative cases, allowing pathologists to focus workload on challenging cases requiring attention.

Chapter 3

Method

In this chapter, high-level design decisions are explained and justified, specifically regarding the deep learning pipelines developed for whole-slide classification and tumour localisation. These decisions have been heavily motivated by the research conducted in the context survey (see chapter 2). Implementation details that are peculiar given the design specification are outlined where required. A testing summary for the deep learning pipeline implementations is given in section 8.5.

3.1 General Design Decisions

3.1.1 Programming Language

The `Python` ($\geq$ 3.0) programming language was selected over alternatives (e.g., `Java`) due to the availability of well-maintained open-source libraries for deep learning and image processing. These libraries mitigate the need for self-implementation and testing, which would hinder the feasibility of the project objectives.

3.1.2 Deep Learning Framework

Due to the considerable memory requirements of the data sets employed and the complexity of the deep CNN models developed, a graphics processing unit (GPU) has been used to provide increased computational power. A '*Nvidia GeForce GTX 1060 (6GB)*' GPU was provided by the School of Computer Science at the University of St Andrews. To take advantage of the computational capability provided by the GPU, a deep learning framework possessing CUDA support (for GPU acceleration) with CNN-based implementations has been selected: `PyTorch` (Paszke et al., 2019). `PyTorch` was chosen over similar alternatives, such as the more popular `TensorFlow (with Keras)` framework, due to familiarity with `PyTorch` by the parties involved in this project. Supplementary packages have been added to this framework, such as `TorchVision/SKImage` for image processing, `RayTune` for automated hyper-parameter optimisation, and `ASAP/OpenSlide` (which are provided by and recommended for the *Camelyon* challenges) for manipulating the high-resolution WSIs.

3.1.3 User Interface

A command-line interface (CLI) has been selected as sufficient, allowing for configurable execution via program arguments and flags. A full user manual for execution of the developed models (and supplementary tools) is included in section 8.4. Where model visualisations are beneficial, the popular `TensorBoard` package has been used for interpretation using a graphical user interface (GUI); example image outputs resulting from tissue detection and data augmentation are displayed, and model training is graphed (i.e., training/validation loss change per epoch), etc.

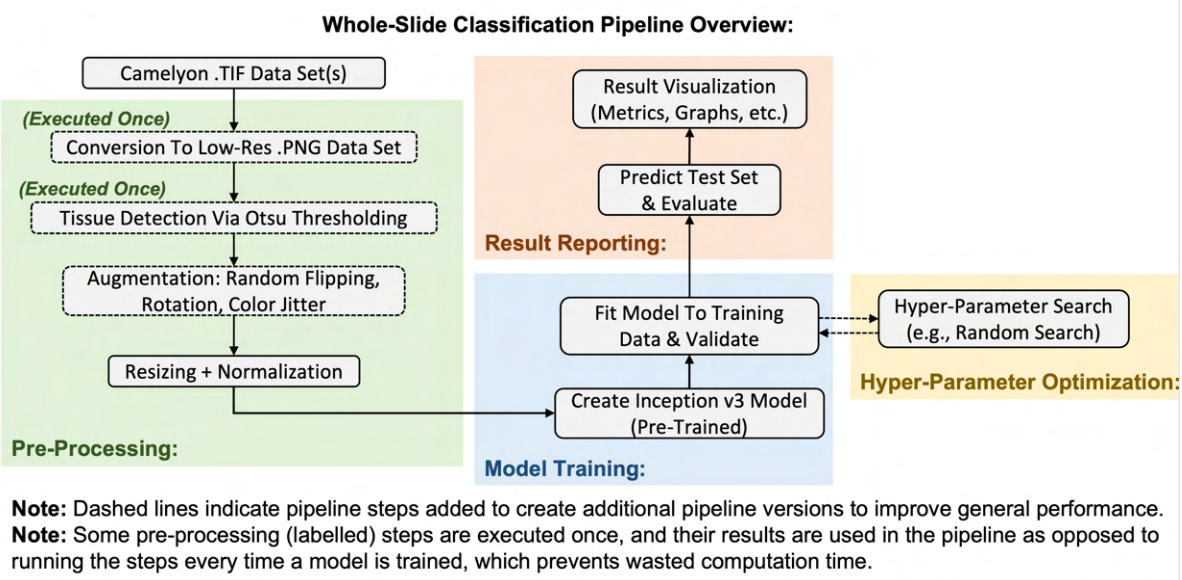
3.2 Whole-Slide Classification Design

This section outlines the deep learning pipeline design for the whole-slide classification task.

3.2.1 Pipeline Overview

The deep learning pipeline for whole-slide classification is characterised as simply reducing the high-resolution *Camelyon* data set images to a sufficiently small size such that they can be fed as input into a pre-trained CNN for binary classification (slides are cancerous or normal). However, additional techniques have also been employed to boost the general performance of this simple approach. The pipeline consists of four phases: pre-processing, model training, hyper-parameter optimisation, and result reporting (see Figure 3.1).

Figure 3.1: Whole-slide classification deep learning pipeline flowchart.



3.2.2 Data Set

This section describes the data set used and explains any manipulations of this data set to assist model training and evaluation.

Data Set Decision: Initially, the *Camelyon-16* data set was used to develop the low-resolution whole-slide classification model. This data set contains 399 WSIs total, comprising normal and cancerous cases. The cancerous slides contain micro- and macro-metastatic regions, of which there may be multiple per slide. This richness of cases in the cancerous slides, with varying ratios of cancerous tissue to total tissue area, benefits model training and evaluation; the data set is representative of real-world scenarios and includes difficult training instances, such as micro-metastatic regions that will likely be increasingly difficult to detect the smaller the input resolution.

However, the developed low-resolution models in this project gave poor general performance on the test set when using the *Camelyon-16* WSIs; the models struggled to separate cancerous and normal cases. The limited *Camelyon-16* training data was identified as the most likely cause. For perspective, the high-resolution methodologies employed by the *Camelyon-16* submissions allowed for the 270 training WSIs to be divided into minimally hundreds of thousand of training patches (millions in some submissions). However, the developed low-resolution models in this project use WSIs in their entirety, so the 270 training WSIs themselves comprise the training set. As a result, there is not enough example training data for the low-resolution models to learn features that perform well in general. To alleviate this issue, the training WSIs from the *Camelyon-17* challenge (500 labelled WSIs) were added to supplement training, increasing the number of training samples from 270 to 770 (∼285% increase). Note, this is still dramatically fewer training samples (even with augmentation) compared to the *Camelyon-16*

submissions and is a natural limitation of the pipeline methodology; using low-resolution WSIs in their entirety can result in dramatically lower inference times, but this is because each WSI is not first dissected into a multitude of training patches that are individually predicted and their results aggregated.

Many of the added *Camelyon-17* WSIs have a black background instead of the white background expected with the *Camelyon-16* WSIs. This has negative impacts on training; the differing backgrounds skew the computed mean and standard deviation for the supplemented training data, which results in poor normalisation. Thus, a basic thresholding of each of the RGB channels was applied to change the black background by converting black pixels to white pixels. Inspection of the WSIs revealed that tissue areas contained dominantly pink-red colours due to staining and would be minimally affected by a removal of black pixel content, which justifies use of the simple thresholding approach. The threshold value applied was empirically investigated via inspection of the pixel value distributions in the WSIs; any pixels with values lower than 45 in all three RGB channels are converted to white pixels. Whilst this converts the black background sufficiently, thin dark lines remain in areas where slide content (e.g., tissue) met the black background (see Figure 3.2). Multiple techniques were investigated to filter out these dark lines, such as morphological area closing and erosion operations, but ultimately this either resulted in blurring of tissue content or tissue content being cropped by the tissue detection process.

Figure 3.2: Demonstration of the black background/content filtering on an example image, '*patient_000_node_0.png*'. The original low-resolution image (left) has its background converted (right) but errant thin dark lines persist, which could not be effectively removed.

Finally, whilst the training WSIs from *Camelyon-17* were added to supplement the training data, the 500 testing WSIs from *Camelyon-17* were not also added to the testing set used; an appropriate performance comparison of this project's low-resolution models to the *Camelyon-16* submissions and pathologists involved requires evaluation on the same test set. Irregardless of this reasoning, the labels for the *Camelyon-17* test set were not publicly available at the time of this project, so the test set WSIs could not be used anyway.

Data Set Split: The *Camelyon-16* and *Camelyon-17* WSIs used, collectively referred to as the *Camelyon* data set, are provided pre-split into training and testing subsets. The training subsets contain 270 and 500 WSIs respectively, whilst the testing subsets contain 129 and 500 WSIs respectively (latter 500 test WSIs not used). This prevents data leakage (i.e., model training with knowledge of the testing set) that would invalidate the estimated general performance of the model - an overly optimistic performance would be reported.

The *Camelyon* training data (770 WSIs) was further split into training and validation partitions using stratified random sampling, which preserves the proportion of the output classes (see Table 3.1 and Table 3.2). An 80% training to 20% validation split was applied, which is common in machine learning and automated breast cancer detection (Yue et al., 2018). It was ensured that the stratified random sampling not only preserved the proportion of the binary output classes (i.e., cancer and normal),

but also the intra-class proportion for the positive cancer class. The distribution of the *Camelyon-16* positive class is $\sim$ 49% for micro-metastases and $\sim$ 51% for macro-metastases, whilst the distribution of the *Camelyon-17* positive class is $\sim$ 20% for ITCs, $\sim$ 32% for micro-metastases, and $\sim$ 48% for macro-metastases. Not ensuring the intra-class proportion could have resulted, for example, in the validation set containing a disproportionately large number of macro-metastases, which are likely easier to detect given their size, so the model would report performing better on the validation set than would be observed in general.

With the described partitions of the *Camelyon-16* and *Camelyon-17* WSIs utilised, the training and validation subsets were used for model training, and the testing subset was reserved for final model evaluation.

Table 3.1: Table summarising the data distribution in the initial *Camelyon-16* data set used.

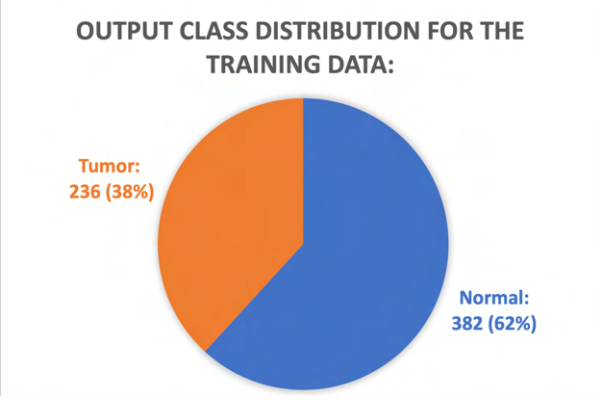
	Train	**Validate**	**Test**	**Total**
Normal	127	32	80	239
Tumour	89	22	49	160
Total	216	54	129	399

Table 3.2: Table summarising the data distribution of the final *Camelyon* data set used, where the *Camelyon-16* data set has been supplemented by adding the *Camelyon-17* training WSIs.

	Train	**Validate**	**Test**	**Total**
Normal	382	95	80	557
Tumour	236	57	49	342
Total	618	152	129	899

Output Class Balance: The model performs binary classification, so inspection of the output class distribution was crucial in determining whether re-balancing was required to prevent model bias towards one of the output classes (see Figure 3.3). The *Camelyon* training data has a positive (i.e., cancer) class proportion of $\sim$ 38%, which reflects the real-world statistic that 60-70% of extracted sentinel lymph nodes do not contain metastases (Kim et al., 2006). Whilst the observed output class imbalance is not severe, re-balancing prevents any minor class bias.

Figure 3.3: Pie chart showing the output class distribution between the positive tumour class and the negative normal class within the training data.



Various solutions to re-balance the data were considered. Under-sampling would involve dropping normal slides to equal the number of cancer slides, but this would diminish the limited amount of majority class data and, therefore, possibly discard useful features. Over-sampling the minority class through the use of, say, the synthetic minority over-sampling technique (SMOTE) (Chawla et al., 2002) would balance the output classes and increase the amount of useful information available to the model, but this is more computationally expensive compared to the simpler technique used: class weights. Class weighting applies greater importance to the under-represented class when computing the loss function

and does not require additional manipulation of the training data. The use of SMOTE for image data could be investigated in future work given its capability to meaningfully increase the size of the training data set, which is currently limited even with the additional *Camelyon-17* training WSIs.

The class weight assigned to the positive cancer class is given as the proportion of normal to cancer training samples:

$$\frac{382 \text{ normal slides}}{236 \text{ positive slides}} =\sim 1.62$$

Thus, the cancer class is weighted ∼62% more in the loss function as there are ∼62% more normal samples than cancer samples.

Label Encoding: The labels for the output classes are provided in categorical string format, but they were converted to numerical format, which is required for model training and inference. As the model performs binary classification, a simple binary encoding of the target classes has been used (see Table 3.3).

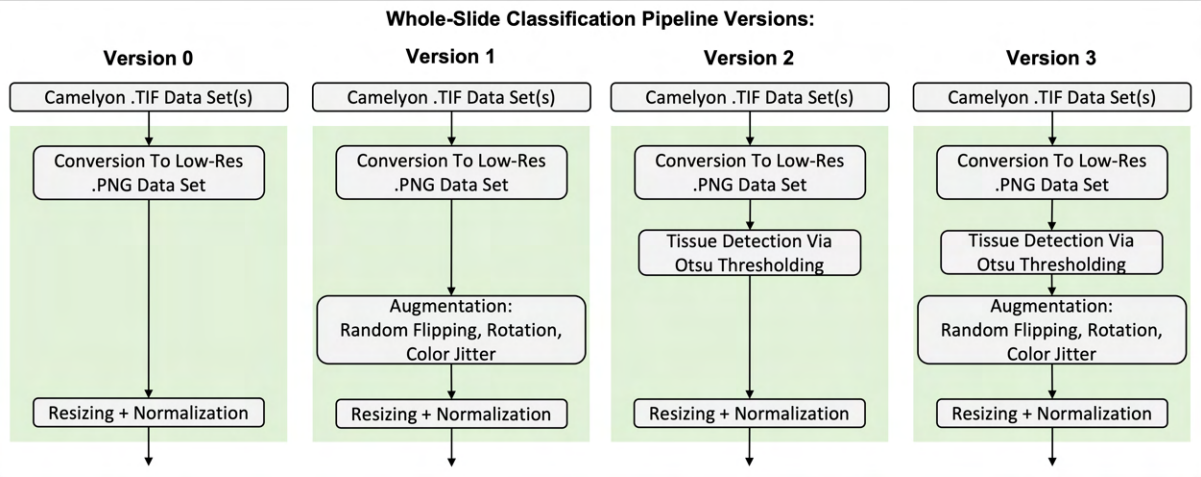
Table 3.3: Table showing binary encoding of the output classes.

Category Label	Binary Encoding
'*Normal*' (or '*None*' or '*Negative*')	0
'*Tumor*' (or '*ITC*' or '*Micro*' or '*Macro*')	1

3.2.3 Pre-Processing

This section outlines the design pertaining to pre-processing carried out on the WSIs before feeding the WSIs as input to the model (see Figure 3.4).

Figure 3.4: Pre-processing in the whole-slide classification deep learning pipeline from subsection 3.2.1. The pipeline versions, as given by the addition of pre-processing steps (green boxes), are individually illustrated. These versions are further explained in this section.



Data Set Conversion To Low-Resolution Images: The high-resolution `.TIF` WSIs were converted to low-resolution `.PNG` images as a one-time pre-processing step (i.e., not carried out by model training and inference every time). This allowed for faster model training (down-sampling of the very high-resolution images need only be processed once) and easier model debugging (the input images are more manageable and inspectable at low-resolution).

Initially, the high-resolution images were down-sampled to resolution level 8 for investigation of different model versions using 299 x 299 pixel input images. In such images, each pixel represents $62.2\mu m^2$ (see Table 2.3). Due to dimension differences between the high-resolution WSIs from different medical centres, not all WSIs had a resolution level 8 image down-sampling available. In these cases, the

level representing the next lowest resolution was selected. As a result, the low-resolution `.PNG` images have a maximum size of approximately 1000 x 1000 pixels. These `.PNG` images can, therefore, be passed in their entirety as input to the deep CNN model.

For later investigations of different input image resolutions up to 2048 x 2048 pixels, the high-resolution images had to be down-sampled instead to 6000 pixels in width and height (i.e., roughly resolution level 6) so that tissue detection would give tissue images of roughly 2048 x 2048 pixels, as desired. Input resolutions to be investigated lower than 2048 x 2048 pixels could be down-sampled from such tissue images. Using the initial data set down-sampled to resolution level 8 would have been pointless; the images would be up-sampled for all desired input resolutions greater than roughly 1000 x 1000 pixels, which gives no gain in information (i.e., up-scaling these images is not the same as 'zooming out' in magnification).

Data Augmentation: Data augmentation techniques were added iteratively based on intuition and context survey research to generate different model versions, each intending to improve on the last. There are four distinct model versions (see Figure 3.4). Augmenting the training WSIs acts as a form of regularisation to prevent over-fitting and, therefore, improve general model performance.

- **Version 0:** No data augmentation used. This model version provides a baseline performance by which later versions can be compared against to ensure the augmentation techniques improve the model.
- **Version 1:** This model version adds common image transformations observed in the *Camelyon* challenges to have boosted general model performance (see Figure 3.5).
  The WSI output labels are orientation-invariant as the slides have no canonical orientation, so random vertical and horizontal flipping of the images is utilised. Also, random rotations of the image are applied in 90 degree intervals (i.e., 0°, 90°, 180°, and 270°). Whilst image rotations could be applied in a continuous range (e.g., [0°, 360°]), this resulted in low-resolution images that either had a significant proportion of pixels cut-out by the rotation, or low-resolution images that kept all of the original image content but had a significant proportion of 'empty' pixels (see Figure 3.6). Both cases are especially detrimental to a low-resolution model that already has limited pixel content to utilise for classification.
  Finally, random colour jitter was also applied to the training images to add staining robustness by encouraging the model to learn colour-invariant features. Whilst staining standardisation algorithms were successful for *Camelyon-16* submissions, most successful *Camelyon-17* teams instead used colour augmentation; the additional *Camelyon-17* WSIs with variable staining from different medical centres made enforcing a reference standard difficult. This indicates that colour augmentation is a more robust augmentation technique for WSIs in general.
  It should be noted that the addition of these data augmentation techniques does not increase the size of the training data set used in the implementation; this is functionally equivalent when using random transformations to simply training for more epochs, which is a more elegant solution. For example, applying a single augmentation to a set of images doubles the number of images, which are used to train for n epochs; however, this is functionally equivalent to applying the transformation at random with 50% probability for $2n$ epochs. The implementation allows for the full breadth of images possible with the augmentations to be explored during training without having to pre-sample and store every possible image variation. As a result, model versions that use these augmentation techniques are permitted to train for at least twice as many epochs. Ideally, many more than twice as many epochs should be permitted since there are several random transformations being applied; though, a balance must be met with total training time given the number of model versions that need to be trained within the project time constraints.

Figure 3.5: Data augmentation examples for version 1 (and version 3) of the whole-slide classification model. An example image, '*tumor_003.png*' (at 299 x 299 pixels), has been used (far left). Horizontal flipping, vertical flipping, 90° (anti-clockwise) rotation, and color jitter on the original image has been demonstrated (from left to right respectively). Note that the color jitter image (far right) is noticeably darker appearing than the original (far left). Also note that rotations may be applied from [0°, 90°, 180°, 270°] even though only 90° rotation is shown. None or multiple augmentations may be applied to each training image at random.

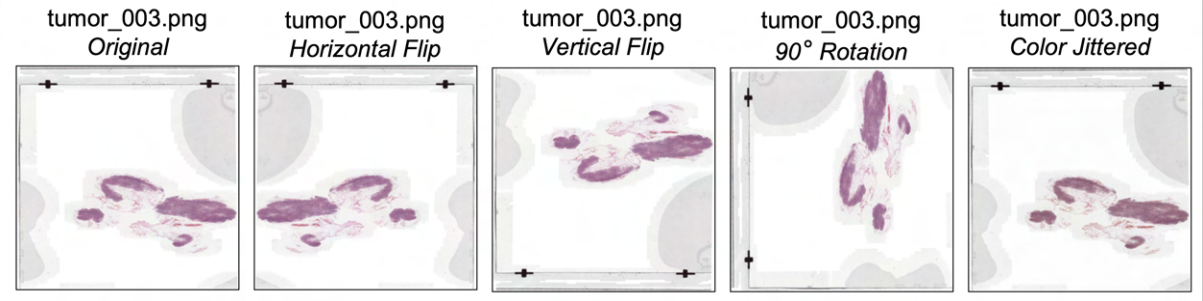
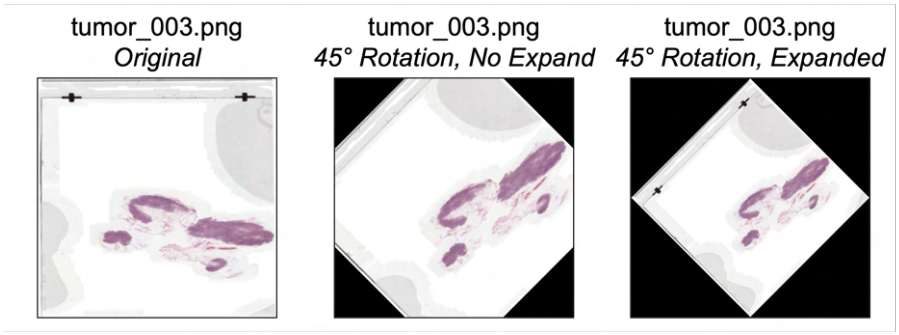


Figure 3.6: Demonstration of justification as to why rotation is applied solely in 90° intervals. An example image, '*tumor_003.png*' (at 299 x 299 pixels), has been used (left). With a continuous rotation range (i.e, 0° to 360°), 'gaps' created in the square image by non-90° interval rotations have to be handled. This results in one of two scenarios: either slide content is cropped out of the image (middle), which could inadvertently remove tissue area and affect the prediction, or keeping all image content results in an image with an undesirably low proportion of pixels relating to the slide content (right). In the worst case, a 45° rotation results in 50% of the image being occupied by 'gap filler' pixels that do not inform the output. This issue is emphasised by the fact that the image is low-resolution (299 x 299 pixels) and only the tissue region within the slide region informs the output, which is already a small proportion of the image. Furthering the issue is that the tumour region may be micro-metastases, which occupies a very small proportion of the tissue and so a very limited number of pixels (if any) would be useful for classification.



- **Version 2:** This model version adds tissue detection to improve model input (but without the transformation techniques from version 1). In previous versions, entire low-resolution slide images were given as input to the deep CNN. However, these slide images contain mostly background; only a small proportion of pixels correspond to the meaningful tissue area that informs the output. Therefore, tissue detection has been used to identify the tissue region within a low-resolution slide image, which is then extracted and used as input to the model (see Figure 3.7).
  Firstly, the input RGB image is converted to the HSV (Hue-Saturation-Value) colour space. Then, Otsu thresholding is carried out on the H and S channels to automatically determine optimal foreground threshold values. A binary image is created from the H and S channels using their corresponding optimal thresholds, giving a mask of the detected tissue regions in the image. Median filtering is applied to the tissue mask to remove spurious regions, though filtering is kept minimal to avoid over-filtering that may remove small tissue regions, which could be micro-metastases. These techniques have been used as they were greatly successful for the *Camelyon-16* submissions. Namely, the methodology of the top-performing '*HMS-MIT*' team has been replicated (D. Wang et al., 2016). Finally, bounding box co-ordinates for the detected tissue are calculated from the mask and used to extract the corresponding area from the original RGB image. Later, slight dilation was added to the detected tissue in the tissue masks just before calculating bounding box co-ordinates so the resulting cropping would not as tightly enclose the tissue area. This allowed for edges of the detected tissue, which define the edges of the bounding box, to be preserved in

instances where a slight (but significant at the cellular level) cropping out of tissue content was observed.

Originally, inspection of the produced tissue images revealed that image content, such as black crosses and text, produced undesirable tissue segmentation. However, the addition of the *Camelyon-17* training WSIs necessitated black content filtering to handle the majority of WSIs containing a black background. This gave raise to the added benefit that such filtering removes the described issues with black crosses and text, greatly improving tissue segmentation in most slides (see Figure 3.8). Despite this, the tissue detection process is still non-optimal; other undesirable image content exists, such as pink smearing (e.g., '*tumor_079*') and red pen marks (e.g., '*tumor_067*'), which are not simple to filter out due to their visual similarity with stained tissue (see Figure 3.9).

Figure 3.7: Demonstration of tissue detection for an example image, '*tumor_009.png*'. From left to right, the original RGB image is converted to the HSV colour space, then Otsu thresholded in the hue and saturation channels to create a binary mask, then median filtered to remove spurious small tissue distortions. Finally, a bounding box for the detected tissue is calculated and used to extract the corresponding region from the original RGB image. The result (far right) is observed to give good tissue detection for the example image (far left).

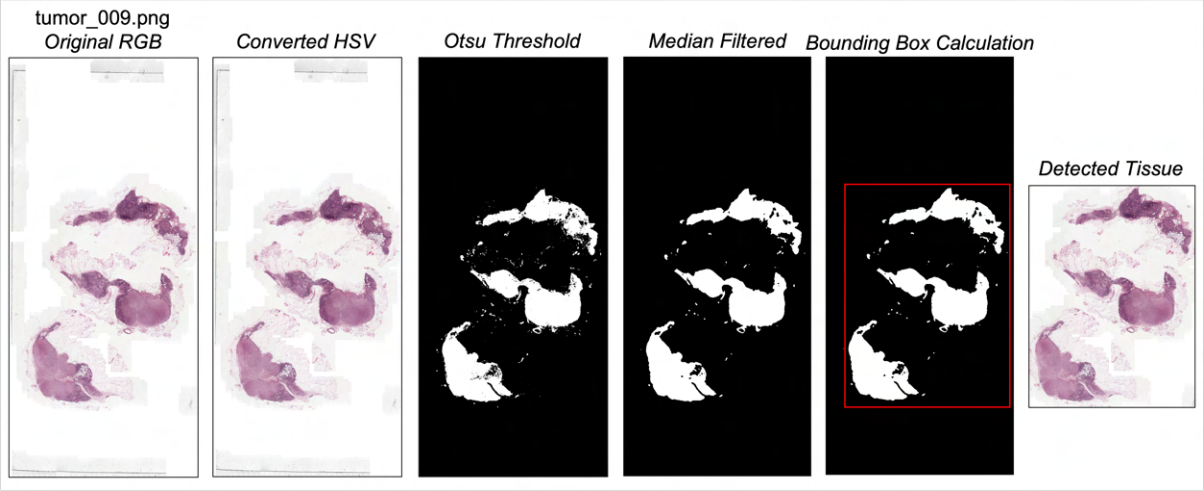


Figure 3.8: Demonstration of tissue detection improvement by adding black image content filtering. The results of performing tissue detection on '*tumor_001.png*' (left) are shown before (middle) and after (right) black image content filtering was added. The black content filtering allows for more robust localisation of tissue regions within the slides as slides containing black crosses or text receive better tissue detection.

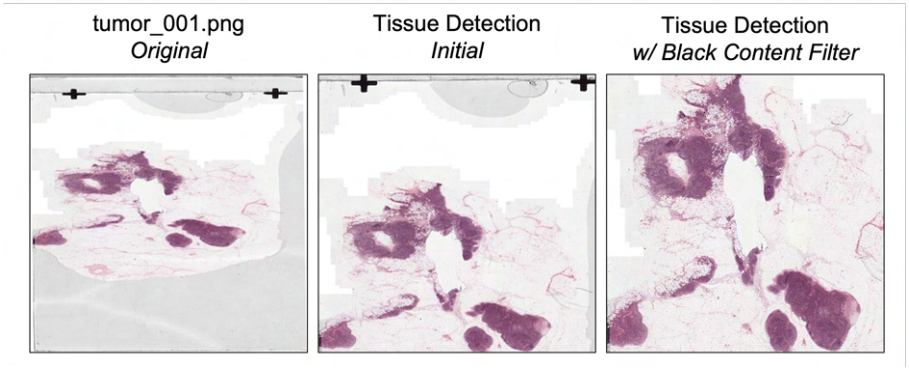
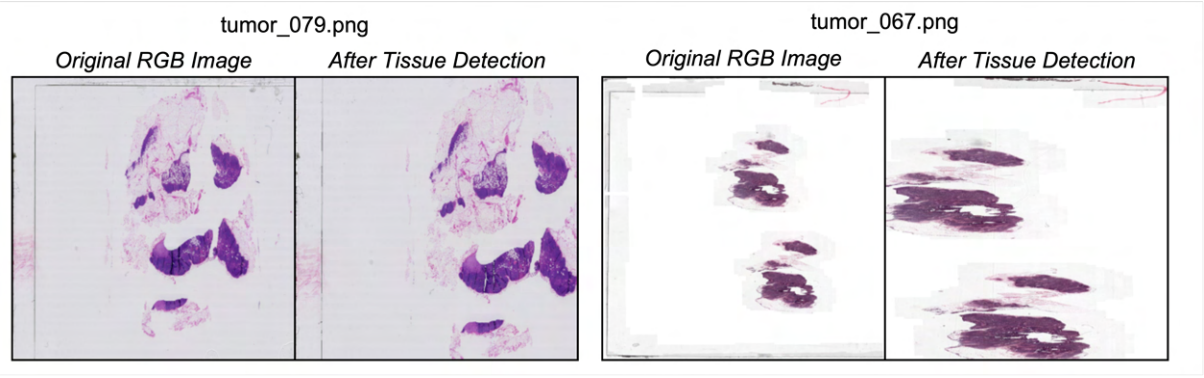


Figure 3.9: Demonstration of the limitations of the tissue detection process. For '*tumor_079.png*' (left), it is observed that tissue detection is non-optimal due to the presence of a pink smear on the far centre-left of the slide. Similarly, for '*tumor_067.png*' (right), tissue detection is non-optimal due to the presence of red-pink pen marks in the upper-right corner of the slide. Future work can improve the tissue detection technique by adding robustness to real-world annotations and imperfections present when the slides were digitised.



- **Version 3:** This model version combines the tissue detection process from version 2 with the transformation techniques from version 1. As a result, this model employs data augmentation for regularisation on low-resolution tissue detected images.

Data Preparation: Any data augmentation used is applied to the training images; however, data preparation is performed on all images (i.e., train, validate, and test) as a pre-processing step immediately before feeding the images as model input. Data preparation consists of image resizing and normalisation.

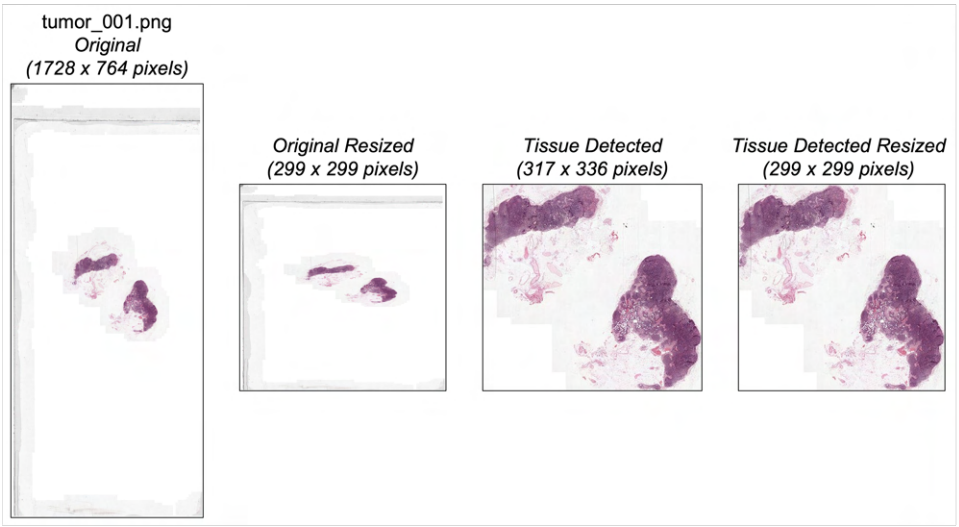
Image resizing involves converting the low-resolution `.PNG` images into target low-resolution dimensions desired for investigation. As an initial choice, image sizes of 299 x 299 pixels were used. This is the smallest input size permitted by the chosen Inception v3 architecture (see subsection 3.2.4). Based on the assumption that the model will likely perform worse as image resolution is decreased (discussed in the context survey - see chapter 2), investigating the input dimensions of 299 x 299 pixels will establish a lower-bound for model performance. The *Camelyon-16* submissions provide an upper-bound for model performance using high-resolution images. From this, the input resolution is explored further within these bounds with the intent to find an optimal resolution that maximises prediction confidence whilst minimising resource requirements.

The further image resolutions investigated are: 512 x 512, 1024 x 1024, and 2048 x 2048 pixels. Investigating greater input resolutions (4096 x 4096 pixels, etc.) is also justified as these resolutions are still low-resolution being thousands of times smaller than the original data set images. However, such resolutions are increasingly infeasible for investigation given the memory constraints they impose, ultimately requiring significant training time (larger image resolutions push the limit on the fact that these images are given in their entirety as input to a CNN for classification). To handle increasing input images resolutions (above the original 299 x 299 pixels), whilst maintaining the same batch size (at 16), an accumulated gradient approach was implemented for use with larger image resolutions that would otherwise create out-of-memory issues. The accumulated gradient approach essentially sets the batch size as 1 so that individual images are loaded into memory. However, the network is only updated once the required batch size amount of images (i.e., 16) have had their losses computed. This is effectively the same as just using the required batch size but memory management is utilised for practical purposes.

A noted issue of the image resizing is that images are morphed to a square (1:1) aspect ratio. As a result, the model versions seek to identify general features to classify cancer in 'squashed' WSIs. The tissue morphing negatively affects classification ability as any information on the shape of cell and multi-cellular structures is damaged. The issue is prolific in model versions 0 and 1, where the original rectangular low-resolution WSIs are used, but is mitigated in versions 2 and 3 as tissue detected regions mostly conform to a roughly square area (see Figure 3.10). However, the issue is not fully solved and is an inherent shortcoming of the pipeline methodology. The *Camelyon-16* submissions were able to use square inputs without issue as they used random cropping of high-resolution WSIs to create training patches;

cropping an image does not distort the image content. In this project, however, the WSIs themselves are used as input and do not all have uniform dimensions (required by the CNN architecture), so resizing must be applied. For versions 0 and 1, the issue of slide content morphing can be mitigated by using a rectangular aspect ratio that conforms better to the low-resolution slides, but the issue would still be present in a lesser form. For versions 2 and 3, extraction of the tissue areas (i.e. cropping of the slides) can be made to ensure a square aspect ratio but this would involve the inclusion of additional background pixels (to varying degrees), which undermines the benefit of applying tissue detection in the first place; tissue detection seeks to minimise the amount of background present which does not inform the output.

Figure 3.10: The issue of slide morphing in model versions 0 and 1 is visualised, along with its mitigation in versions 2 and 3. For an example slide, '*normal_001.png*', the original WSI used for versions 0 and 1 (far left) receives significant squishing when resized as model input (middle left). However, the tissue detected version of this slide used for versions 2 and 3 (middle right) receives very minor morphing (far right) when prepared as model input since the tissue detected region is roughly square. This was observed for most *Camelyon* WSIs used.

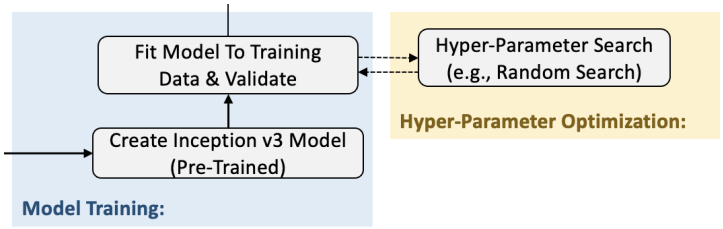


Finally, the training images are normalised, which is a common pre-processing step to improve training by increasing model numerical stability and/or speeding up the training process (Russell & Norvig, 2020; Yue et al., 2018). The model was initially trained without normalisation, then with the normalisation values recommended for the `PyTorch` pre-trained Inception v3 model, and finally with normalisation values calculated for the *Camelyon* low-resolution training WSIs (i.e., a tool was developed for calculating the per channel mean and standard deviation across the *Camelyon* WSIs). The *Camelyon* normalisation values were used as they are more suitable than those provided for the pre-trained Inception v3 model. The *Camelyon* WSIs (containing monotonous stained tissue) are fundamentally different from the '*ImageNet*' images used to train Inception v3 (1000 classes including goldfish, umbrella, etc.), so computing the mean and standard deviation for the *Camelyon* data set is more appropriate. Furthermore, when using tissue detection (i.e., versions 2 and 3 of the model), the mean and standard deviation computed for the tissue detected images is used for normalisation instead since the input data to the model is changed significantly.

3.2.4 Model Training

This section outlines the design pertaining to model training carried out using the pre-processed WSIs (see Figure 3.11).

Figure 3.11: Model training in the whole-slide classification deep learning pipeline from subsection 3.2.1.



Deep CNN Architecture: The majority of successful submissions to the *Camelyon* challenges used a select few deep learning architectures, primarily GoogLeNet, AlexNet, or VGG-Net. Whilst each of these architectures are viable choices for this project's low-resolution models, ultimately, a GoogLeNet based architecture was selected as this was used by the top performing '*HMS-MIT*' team for the whole-slide classification task. Specifically, the Inception v3 architecture (a newer version of GoogLeNet, also known as Inception v1) was selected as this provides several iterations of improvement upon the original GoogLeNet design.

The Inception v3 architecture could have been substituted for the other viable architectures (e.g., AlexNet and VGG-Net) to determine concretely which architecture is best suited for breast cancer detection using low-resolution images. Furthermore, the predictions of these different models could have been aggregated to form an ensemble, which was a technique observed with many *Camelyon-16* submissions, to determine if general performance could be even further improved. Assuming the ensemble models could infer via parallel processing, and that there is no significant disparity between their inference times, the use of an ensemble could improve prediction performance with little expense to the overall average inference time.

Transfer Learning: A pre-trained Inception v3 architecture has been utilised, which was originally trained on the 1000-class 'ImageNet' data set (Russakovsky et al., 2015). This utilises transferred learning, allowing for classification knowledge from a general domain to be leveraged. This is especially useful in this project where the small size of the training data set limits learning capability; the scarce training data is better utilised by fine-tuning the weights of an existing network, as opposed to training an entirely new one. A fully-connected layer has been used at the end of the Inception v3 architecture and consists of a single output neuron. The output neuron has Sigmoid activation, so the model provides a single value between 0 and 1, which represents the probability that the input image belongs to the positive cancer class. The fully connected layer described replaces the original pre-trained fully-connected layer (instead of simply appending to it) as the original output classes for the pre-trained network (i.e., 1000 'ImageNet' outputs) are very different from the output classes in this medical application. The pre-trained feature extraction layers are useful, but the pre-trained classification layers far less so. Overall, this converts the architecture from 1000-class classification to a binary classifier for the whole-slide classification task.

Loss Function: The selected loss function is binary cross entropy loss (also known as log loss), which is a standard machine learning choice for binary classification tasks (Litjens et al., 2017). Binary cross entropy compares each predicted output probability to its corresponding actual output. An overall score is calculated which penalises the predicted probabilities based on distance from their expected output. Usefully, loss increases exponentially the more incorrect (greater the distance) a predicted probability is from the expected output (i.e., large penalties deter the model from predicting low probabilities for positive class samples).

Optimiser: Due to the complexity involved with the deep nature of the CNN model, it is key to minimise the number of hyper-parameters to tune. Adaptive learning rate optimisation algorithms provide good out-of-the-box performance via fast convergence and little parameter fine-tuning when

compared with traditional optimisation algorithms, such as stochastic gradient decent (SGD). The most generally used adaptive optimiser is adaptive moment estimation (Adam), which combines momentum (for larger steps in the direction of the steepest gradient) and root mean square propagation (for higher acceleration on steep slopes). Adam was selected as a suitable optimiser for these qualities, though many different optimisers and many variations of Adam itself exist. An improvement would be to perform an empirical investigation as to which optimiser yields the best performance for the *Camelyon* data set. This was not prioritised due to the complex interplay of model variables, where a change to one variable may alter the best choice for all other variables, so great effort would need to be expended for what would be tantamount to a limited gain in terms of the project objectives.

Early Stopping: As discussed, the original training data was separated into training and validation partitions with an 80%/20% split respectively. At the end of each training epoch, model predictions are performed on the validation set to obtain an epoch validation loss. Training is halted if this validation loss has not decreased significantly within some number of epochs to prevent over-fitting on the training data. The number of epochs permitted without improvement was configured as 10% of the maximum number of epochs for training. This proportion balances the need to prevent over-fitting, whilst permitting sufficient patience as to not prematurely stop training due to local minima.

An improvement to this design would be to use K-fold cross validation, which allows for training using all of the available training data (especially useful given the limited size of the training data set), prevention of validation set over-fitting, and more accurate reporting of model performance on unseen data. However, K-fold cross validation divides the training data into K subsets and effectively trains K different models (each training on different $K-1$ subsets and evaluating on the final K^{th} subset). Due to project time constraints, the required time to train a model (in the order of hours and emphasised by the presence of several model versions requiring hyper-parameter optimisation with a broad search space), and an unfamiliarity with its implementation in `PyTorch`, the use of K-fold cross validation was not prioritised and a simple validation set was opted for.

Hyper-Parameter Optimisation: The tuning of model hyper-parameters is an important step for ensuring the capability of the deep learning pipeline is realised. The hyper-parameters tuned are batch size, learning rate, and maximum number of epochs.

An automated hyper-parameter optimisation approach has been used. First, search of a broad hyper-parameter space is performed on the base model (i.e., version 0 with no data augmentation) to establish decent hyper-parameter values that each of the model versions can be trained with for version comparison. Then, search using a finer hyper-parameter space is performed on the best model version to fine-tune the hyper-parameter values and maximise the whole-slide classification performance. None of the model versions showed particular promise initially when using 299 x 299 pixel input images (see subsection 4.2.1). Thus, optimisation was performed on model version 3 since this version is theoretically best, although this could not be empirically ensured.

Random search provides a reasonable compromise between execution time and finding sufficient hyper-parameter values by randomly sampling from the hyper-parameter search space. An alternative, grid search, guarantees that the hyper-parameter values presented are optimal by exhaustively exploring the search space but requires significant time, especially for larger search spaces. Despite this, the automated hyper-parameter optimisation implemented allowed for both forms of search to be leveraged; grid search could be used for hyper-parameters with few possible values (i.e., batch size), whilst remaining hyper-parameters with a broader range of values could be sampled using random search (i.e., learning rate and maximum number of epochs).

3.2.5 Whole-Slide Classification Design Summary

The following provides a design summary of the deep learning pipeline for whole-slide classification.

1. Data Set:
    - *Camelyon-16* data set was selected, which initially contained 399 WSIs but was supplemented with the 500 training WSIs from the *Camelyon-17* data set. Thus, 770 training WSIs and 129 testing WSIs were used.

- The training WSIs were split into 80% training and 20% validation partitions using stratified random sampling.
- Training partition output classes were balanced using class weights as there is a majority negative (i.e., normal) class.
- The expected output labels were binary encoded ('normal' $\rightarrow$ 0, 'tumour' $\rightarrow$ 1).

2. Pre-Processing:

   - Prior, one-time conversion of high-resolution WSIs (at resolution level 0) to low-resolution slide images was executed.
   - Black content filtering was applied across all low-resolution WSIs to handle background differences between *Camelyon-16* and *Camelyon-17* WSIs.
   - Model version 0 simply uses the original low-resolution WSIs as input to the deep CNN model (baseline for future version comparison).
   - Model version 1 includes data augmentation via random horizontal flipping, vertical flipping, rotation, and colour jitter.
   - Model version 2 adds prior tissue detection of low-resolution WSIs but without the version 1 transformation techniques.
   - Model version 3 combines tissue detection from version 2 and the image transformation techniques from version 1.
   - Data preparation includes resizing to a target image resolution and normalising using mean and standard deviation values computed for the input data set across each of the RGB channels.

3. Model Training:

   - Model architecture was selected as Inception v3 using a new full-connected layer for binary output.
   - Transfer learning via pre-trained model on 'ImageNet' with weights fine-tuned.
   - The optimisation algorithm is adaptive moment estimation (Adam).
   - The loss function is binary cross entropy loss.
   - Early stopping when validation loss has not improved significantly within 10% of maximum epochs.
   - Hyper-parameter optimisation via automated hybrid of grid search and random search.

3.2.6 Whole-Slide Classification Design Improvements

Several techniques were investigated to improve the general performance of the initial low-resolution whole-slide classification model (i.e., version 0). Namely, data augmentations (e.g., flipping, rotations, and colour jitter) and tissue detection were the primary techniques investigated (resulting in model versions 1, 2 and 3). Furthering this, the training data was supplemented by adding the 500 *Camelyon-17* training WSIs out of recognition that the initial poor general performance observed was likely due to there being an insufficient amount of training examples in the initial *Camelyon-16* data set.

Despite this, there are several further improvements that could be investigated in the future (outwith those already mentioned) to improve the developed low-resolution models which could not be prioritised in this project. The following discusses several areas of possible exploration, though these are not an exhaustive list.

Hard-negative mining is a technique where images classified as false positives could be fed back into model training more often (i.e., as duplicates) to pull the model weights towards learning these hard cases. This technique was popular amongst *Camelyon-16* submissions and was used by many of the top performing teams for whole-slide classification (e.g., '*HMS-MIT*').

Furthermore, several limitations were identified in the tissue detection methodology implemented. It was observed that additional filters would need to be added each time a new real-world issue was identified in the slides preventing adequate tissue detection: black crosses, black versus white slide backgrounds, slide bubbles, pen marks, smears, etc. Some issues within the slides cannot be solved via simple filtering, such as the presence of duplicate tissue areas which creates redundancy; in such cases roughly half of the limited pixels in a tissue detected image are wasted by repeated data that does not provide additional information for classification. Instead of attempting to develop a suite of filters to achieve ideal tissue detection in all cases, a separate CNN could be trained solely for the prior task of tissue segmentation. Whilst this was used with limited success by the *Camelyon-16* submissions (e.g., '*Warwick-QU*'), this would make the developed pipeline an end-to-end deep learning pipeline, minimising the amount of medical domain knowledge required to improve model performance.

Finally, test time augmentations were observed to have been used by some submissions in the *Camelyon-17* challenge. This involves applying all implemented augmentations to each input image at test time, performing a model prediction for each image, and then aggregating the results to create a final probability score. This technique, interestingly, has the potential to further extract model performance after training and may steady the predictions of unstable trained models (i.e., where small input image variations dramatically affect the predicted probability).

3.3 Tumour Localisation Design

Due to project time constraints and the observed poor performance for the whole-slide classification task, the initial project objectives were re-prioritised. Initially, the project aims were to develop basic models for both whole-slide classification and tumour localisation. Then, these models would be improved and a range of input resolutions investigated. However, training of the initial whole-slide classification model (i.e., model version 0) revealed that the classifier not only gave poor performance, but was effectively skill-less at the lowest input resolution (see subsection 4.2.1) even when the training set was supplemented with more training WSIs from *Camelyon-17*. Intuitively, if a pipeline cannot tell if cancer is present in a low-resolution slide, then it seems unreasonable that a similar methodology would be able to locate cancer within a slide any easier. As a result, instead of developing two simple attempts at the two medical machine learning tasks, a single in-depth attempt at whole-slide classification was prioritised.

It may be the case that a tumour localisation pipeline can effectively locate cancerous regions in low-resolution slides. If so, this could present a more effective basis for whole-slide classification than the whole-slide classification pipeline developed in this project. This remains as possible future work to extend that of this project.

In this future work, a similar tumour localisation methodology could be developed to the whole-slide classification methodology. The pipeline would be characterised by feeding entire low-resolution WSIs as input to a deep CNN architecture for tumour localisation, supplemented by additional techniques to improve general model performance. This pipeline would effectively replace the Inception v3 model for binary classification with a segmentation model for tumour localisation by using region-based CNNs, such as Faster R-CNN or Mask R-CNN. Use of these models would require significant alteration to the *Camelyon* WSIs to make them suitable for training (i.e., metastasis bounding box co-ordinates would need to be produced for all cancerous WSIs as positive training samples). Also, further research into unfamiliar aspects such as loss functions that evaluate segmentation ability would be required.

Chapter 4

Results

In this chapter, the metrics used to evaluate the low-resolution models are defined and the corresponding results for the whole-slide classification task are presented and analysed.

4.1 Evaluation Metrics

The main metric used to evaluate this project's low-resolution models was used by the *Camelyon-16* challenge (Bejnordi et al., 2017), allowing for meaningful performance comparisons to the *Camelyon-16* submissions and actual pathologists involved. This is slide-based evaluation for the whole-slide classification task. Additional metrics are used for resource requirement comparisons, or to support further analysis of the developed models. For example, secondary metrics are formed from the confusion matrices of the binary classifiers (precision, recall, etc.) to analyse the limitations of the whole-slide classification models, where appropriate.

- **Slide-Based Evaluation:** The metric is the area under the receiver operating characteristic curve (i.e., Area Under ROC or AUC). This will be referred to as the AUC score. The ROC curve is a plot of the true positive rate (i.e., sensitivity) versus the false positive rate for all classification threshold values.

$$\text{True Positive Rate} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Negatives}}$$

$$\text{False Positive Rate} = \frac{\text{False Positives}}{\text{False Positives} + \text{True Negatives}}$$

  Furthermore, percentile bootstrapping has been used to provide 95% confidence intervals with presented AUC scores (this practice was also followed by the *Camelyon-16* submissions). Providing such confidence intervals aims to estimate differences in performance observed by using different seeds when training the models and performing inference (the developed pipeline uses seeding to ensure reproducibility of results but seed choice affects model performance).

- **Average Inference Time:** The average inference time of the developed models is of particular importance to this project; the aim is to improve the clinical viability of automated breast cancer detection techniques. The average inference time for the developed low-resolution models includes any time taken to convert the original high-resolution images into low-resolution model inputs, as well as the time taken for the model to use the input to make a prediction. In all cases, average inference times are reported where a '*Nvidia GeForce GTX 1060 (6GB)*' GPU was used for execution.

4.2 Whole-Slide Classification Results

This section details the results generated by the low-resolution models developed for the whole-slide classification task.

4.2.1 Model Version Comparison Results

Hyper-parameter optimisation was performed on model version 0 (i.e., no data augmentation) to establish a decent initial set of hyper-parameter values. Then, subsequent version models were trained using these values for comparison. All hyper-parameter values were kept equal, except that the maximum number of epochs was doubled for later model versions using data augmentations (i.e., flipping, rotation, and colour jitter) to permit sufficient use of the augmented images. The implementation does not increase the data set size, but rather applies the transformations at random, which is functionally equivalent when more epochs are permitted.

The average (per image) inference time in all cases includes the average conversion time from a high-resolution `.TIF` image to a low-resolution `.PNG` image with black image content filtering applied (∼360 milliseconds), as well as the average time required for the model to infer a low-resolution image. Where tissue detection is employed, the average (per image) inference time also includes the average time taken to convert an original low-resolution `.PNG` image to a low-resolution tissue detected image (∼190 milliseconds).

The following table presents the results for each model version when predicting the test data set at the initial input resolution of 299 x 299 pixels. The test data set is that of *Camelyon-16*, containing 129 low-resolution WSIs (80 normal and 49 cancerous).

Table 4.1: Comparison of AUC score and average inference time for the different model versions when using 299 x 299 pixel input images. *The confidence interval for the final model version appears not an interval since this version's model predictions all begin with 0.5 and differ only after the thousands decimal column.

Model Version	**AUC Score (With 95% Confidence Interval)**	**Average Per Image Inference Time (Seconds)**
0 - *No Data Augmentation, Original Low-Res Images*	0.537 [0.429 - 0.638]	0.437
1 - *Data Augmentation, Original Low-Res Images*	0.373 [0.274 - 0.474]	0.437
2 - *No Data Augmentation, Tissue Detected Images*	0.437 [0.339 - 0.543]	0.627
3 - *Data Augmentation, Tissue Detected Images*	0.500 [0.500 - 0.500]	0.627

The presented AUC scores are confusing and misleading in their representation of general performance for the different model versions; the AUC score should be between 0.5 and 1. Further investigation of the developed low-resolution models revealed that none of these models have any significant skill in determining whether metastases exists within the 299 x 299 pixel images; the developed low-resolution models cannot separate the metastatic and normal slides (see Figure 4.1 and Figure 4.2). There are several reasons for the poor capability of the models which primarily concern either a defective training process or deficient training samples. However, high confidence in the correctness of the pipeline implementation (see section 8.5), as well as the fact that these models are observed to converge to a validation loss minimum, leads to the following conclusion: the very low input resolution of 299 x 299 pixels is simply insufficient for meaningful classification with the developed pipeline.

Figure 4.1: Figures showing the predicted probabilities for the test data set images (at 299 x 299 pixels) coloured by their actual output class for model versions 0 and 1. No model version (shown below in their respected order) exhibits skill. A skilled model would have many negative (blue) samples close to 0 and many positive (red) samples close to 1, with some overlap being observed for difficult samples. However, no separation of the output classes is observed. The models instead predict some dominant probability in most cases (∼0 for versions 0 and 1).

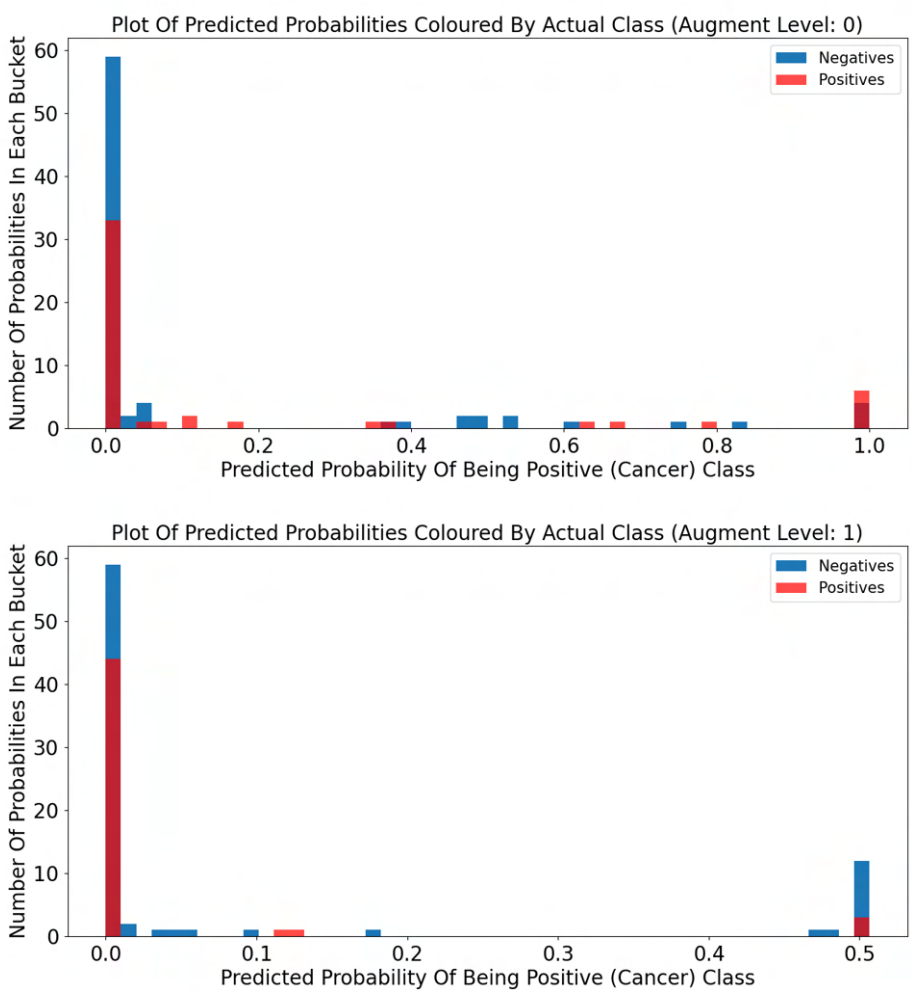
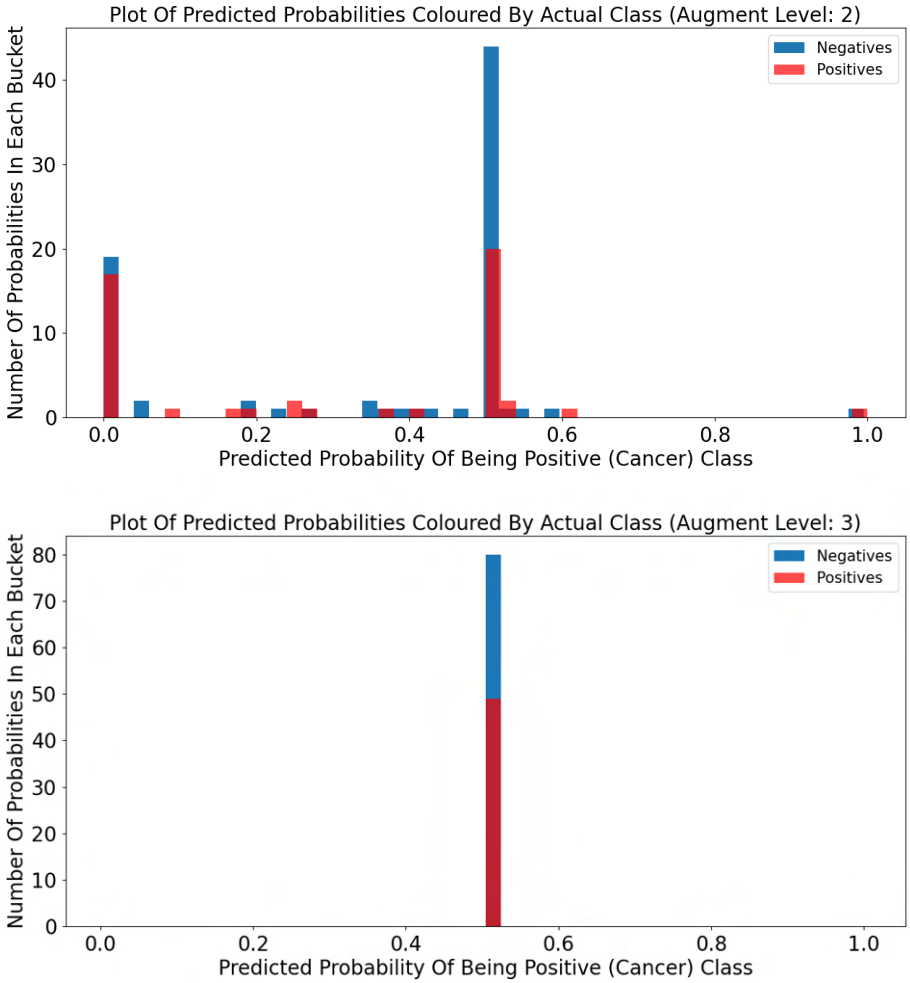


Figure 4.2: Figures showing the predicted probabilities for the test data set images (at 299 x 299 pixels) coloured by their actual output class for model versions 2 and 3. Likewise to model versions 0 and 1, these models are skill-less and predict some dominant probability in most cases (∼0.5).



In figures 4.1 and 4.2, model versions 0 and 2 (the versions that do not use the flipping, rotation, and colour jitter augmentations) have a greater spread of predicted probabilities. The greater spread indicates that these models are trying to make informed decisions as opposed to simply always predicting one of the classes (although all models dominantly predict some probability). This could indicate that the data augmentation strategy is insufficient. However, the opposite may also be true. By applying data augmentations, model versions 1 and 3 may have learnt orientation- and colour-invariant features that increase the certainty of predictions.

Model version 3, which applies data augmentations to tissue detected images, only predicts that both classes are roughly equally likely with little variance (Figure 4.2). Contrarily, this means the model for version 3 has either learnt nothing from training or has learnt that it cannot predict with any certainty given the limited pixel information available within the 299 x 299 images.

As an aside, many result analysis techniques to aid understanding of the developed models were implemented, chiefly, the generation of confusion matrices with the class distribution of false negative samples being provided (i.e., macro-metastasis, micro-metastasis, and ITCs). These were intended to analyse the types of slide and cancer the models have poor performance classifying. However, as the models exhibit no separation of the output classes, a suitable threshold to generate the confusion matrices could not be selected (so the analysis described could not be performed).

4.2.2 Input Resolution Comparison Results

It was observed in subsection 4.2.1 that the model versions developed in this project were effectively skill-less at classifying the existence of cancer in low-resolution images at 299 x 299 pixels. This is likely caused by an inherent limitation of the methodology: there are too few training images to permit sufficient training, the images themselves do not possess sufficient information for classification, or both factors share responsibility. Little can be done to investigate the former issue as more training samples simply do not exist for use. Thus, this section investigates model performance at different input resolutions, all still considered to be low-resolution when compared to the original high-resolution `.TIF` images. These different input resolutions will, therefore, still provide significant inference time improvement to the high-resolution techniques of the *Camelyon* submissions.

The additional image resolutions investigated are 512 x 512, 1024 x 1024, and 2048 x 2048 pixels (299 x 299 pixels was initially used). Across all of these input image resolutions, pipeline version 3 was selected for use. Version 3 should theoretically provide the greatest general performance (although this could not be empirically shown in subsection 4.2.1) by making use of regularisation via data augmentations and higher quality input images via tissue detection. Also, the hyper-parameter values used were the same as those used for version 3 with 299 x 299 pixel inputs to ensure any increase in model performance results from the increased resolution of the input images.

The following table presents the results for model version 3 when predicting the test set at different input resolutions.

Table 4.3: AUC score and average inference time for model version 3 across different input image resolutions. *Training for the model using 2048 x 2048 image inputs did not reach convergence within reasonable time (allotted time of 72 hours), so this model could not be evaluated. This primarily resulted from a significant slowdown of the training process for this model when accommodating batches of larger resolution images in memory.

Input Resolution (Pixels)	**AUC Score (With 95% Confidence Interval)**	**Average Per Image Inference Time (Seconds)**
299 x 299	0.500 [0.500 - 0.500]	0.627
512 x 512	0.530 [0.429 - 0.628]	12.070
1024 x 1024	0.540[0.452 - 0.629]	12.100
2048 x 2048	Timed Out*	Timed Out*

As expected, there is an increase in inference time for larger input resolutions to infer the original high-resolution WSIs. However, the reported inference times (largest is 12.100 seconds) are significantly lower than those of the *Camelyon* submissions; a method reproduction for the top-performing '*HMS-MIT*' team found that their high-resolution method requires roughly 6 minutes and 35 seconds per high-resolution WSI on average[1].

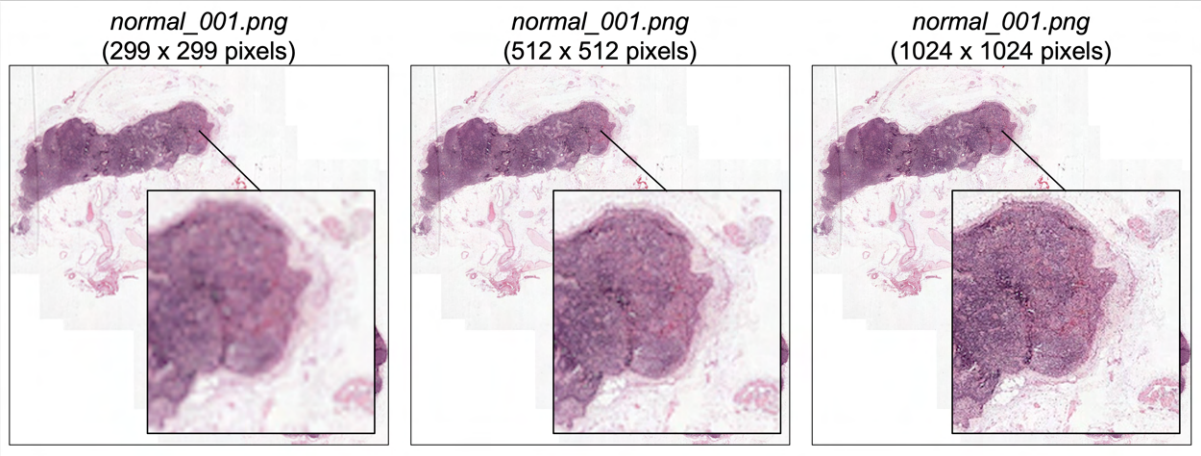
The large increase in average inference time between 299 x 299 pixels (0.627 seconds) and 512 x 512 pixels (12.070 seconds) and very small increase between 512 x 512 and 1024 x 1024 pixels (12.100 seconds) is expected. The 299 x 299 pixel model derives its inputs from the original WSIs down-sampled to resolution level 8. The one-time pre-processing operations of black content filtering and tissue detection occurs on images of roughly 1000 x 1000 pixels, which are then reduced to tissue images at the desired input resolution. However, the 512 x 512 and 1024 x 1024 models both derive their inputs from tissue detection carried out on the original WSIs instead down-sampled to roughly resolution level 6. Thus, the one-time pre-processing occurs on images of roughly 6000 x 6000 pixels (which takes significantly longer) in order to produce the desired tissue images of the required input resolution. Ensuring that input images were derived by down-sampling and pre-processing from higher than required resolutions

[1]The reproduced *HMS-MIT* method was developed by the *iCAIRD* team (of Christina Fell, Mahnaz Mohammadi and David Morrison) at the University of St Andrews. Note that *Camelyon-16* submissions did not present their average inference times per WSI as this was not pertinent to the challenge, hence the use of reproduced results. The presented average inference time per WSI was calculated over 4 separate runs (different model configurations each run) across 215 WSIs.

was necessary given that image up-scaling is not equivalent to an increase in magnification (i.e., the operation does not give the added detail, the images are just increased in size).

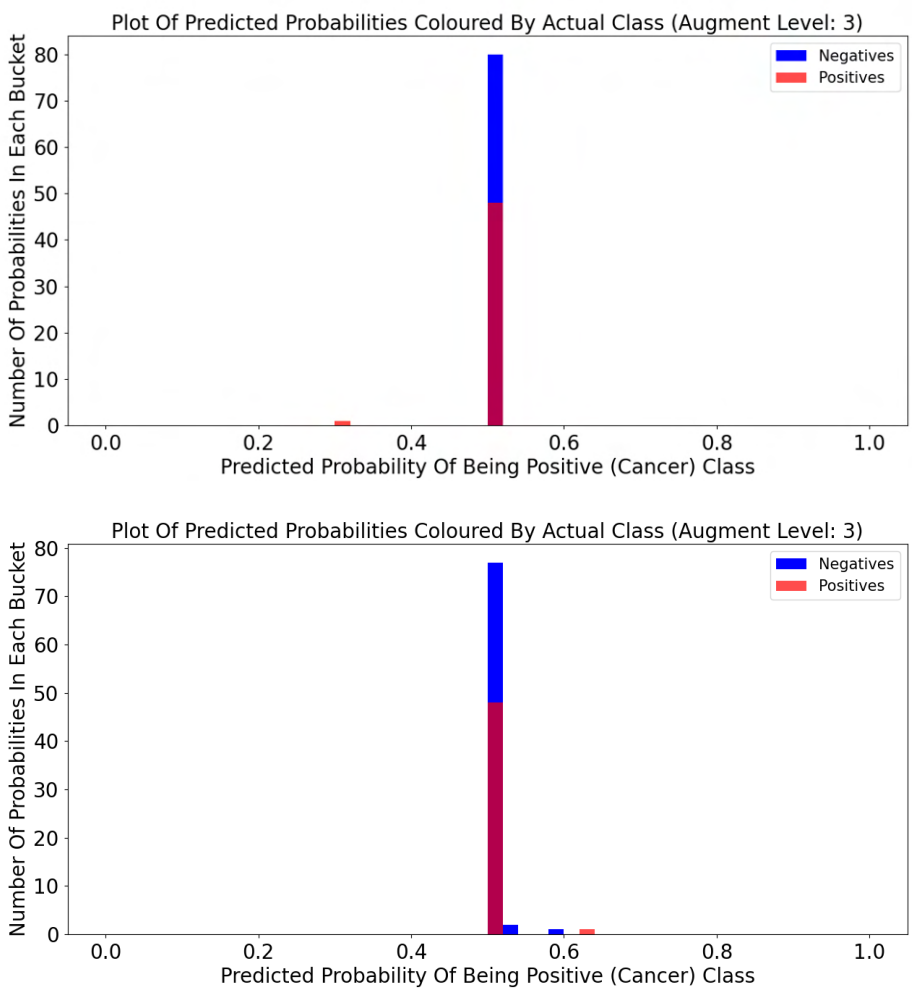
Initial inspection of the produced AUC scores in Table 4.3 leads to the conclusion that model performance improves as the input resolution is increased; the AUC score has increased slightly from 0.500 at 299 x 299 pixels to 0.540 at 1024 x 1024 pixels. This supports the intuition that, as input resolution increases, the model should become more capable at discerning cancerous tissue due to the increased spatial (i.e., cellular) detail available (demonstrated in Figure 4.3).

Figure 4.3: Figure illustrating the increase in spatial detail given by using higher input resolutions. In the following images, '*normal_001.png*' has been used as an example. There is a clear difference in spatial detail given by increasing the resolution. At 299 x 299 pixels (far left), only the macro-cellular structures of the tissue can be discerned and with difficulty due to prevalent blur. However, at 1024 x 1024 pixels, macro-cellular structures can be perceived more clearly and even individual cells can be discerned (though with blur).



However, the improvement in average inference time is superficial since further inspection reveals that these models are also incapable at separating the output classes in the test data set (see Figure 4.4), as observed before when investigating the different model versions. As AUC score is increasing, the models using higher input resolutions can be said to be slightly less incapable at separating the output classes, though marginally. Irregardless, these models are still insufficient for use as they essentially predict at random; the models at all input resolutions investigated dominantly predict scores close to 0.5. As before, this could mean the models have insufficient training and have not been able to learn the general features required for confident classification (i.e., not enough training samples). Alternatively, the models may have learnt that no prediction of any certainty can be made given the limited information within the input images; even though 1024 x 1024 inputs give more than 10 times the amount of pixel content compared to 299 x 299 inputs (i.e., a magnified view of the slide), critical information that informs on cancer presence is still indiscernible. For example, the density of cells in a given area and the regularity of the cell shapes are indicators of cancer, especially regarding micro-metastasis and individual tumour cells, but these factors are obfuscated by blurring.

Figure 4.4: Figures showing the predicted probabilities for the test data set images coloured by their actual output class for model version 3 when using 512 x 512 and 1024 x 1024 input images, respectively. As before, neither of the models display any capability at separating the output classes in the test data set and instead dominantly predict the perfectly uncertain score of 0.5 with little variance.



In summary, the developed deep learning pipeline has been shown to be insufficient even when the input resolution has been increased to 512 x 512 and 1024 x 1024 pixels. A slight observed increase in AUC score (as the input resolution has been increased) indicates that the models are becoming less incapable at separating the outputs, suggesting that some as yet uninvestigated minimum resolution is required for class separation and better than chance predictions. However, the benefit of further investigation to find such an input resolution is questionable given the increasing in-feasibility of training CNNs using these larger resolution images entirely as input. For example, attempts to investigate 2048 x 2048 pixel inputs were fruitless as the model did not reach convergence after several days of training. Although, the feasibility of model training depends on available equipment, allocated training time, and perceived benefit of the resulting model, so no claims can be made that future work investigating larger input resolutions would not be beneficial.

Chapter 5

Discussion

The over-arching result of this project is the developed deep learning pipeline for binary cancer classification is effectively skill-less at the investigated low input resolutions, even when additional techniques were implemented to improve general capability.

Notably, differences in pipeline methodology between this project and most *Camelyon-16* submissions (i.e., using entire low resolution images as input versus the patch-based probability map approach) means that although many similar pipeline techniques have been used (the pre-processing and model training strategies that have been used are directly inspired by the *Camelyon-16* submissions), the resulting models possess effectively opposite capabilities. This project's models have no classification capability, whilst the top-performing '*HMS-MIT*' (D. Wang et al., 2016) team achieved an AUC score of 0.994, outperforming time constrained pathologists (0.810) and performed comparably to a pathologist without time constraints (0.966). As this project's models are skill-less, direct comparisons of AUC score to the *Camelyon-16* submissions is unsuitable and has not been undertaken. Furthermore, even though this project presents an order of improvement in average inference time (seconds as opposed to minutes), ultimately, this improvement is not beneficial as classification capability is the significantly more important clinical factor; a slow model that predicts with high confidence can be useful whilst a fast model that predicts at random is useless.

Even more damning is the fact that some multi-resolution approaches in the literature exhibit similar or better average per image inference times and have significant classification capability. For example, Hering and Kybic (2020) presented a multi-resolution approach that achieves average inference times between 1-1.5 seconds with AUC scores between 0.92-0.95 on the *Camelyon-16* test set. Currently, such results only support the redundancy of single-resolution, low-resolution approaches.

This project's deep learning pipeline may have increasing proficiency at larger input resolutions than those investigated, possibly revealing some optimal resolution that captures sufficient prediction performance and average inference time. However, this seems unlikely given that there appears to be an inherent trade-off between the two desired attributes, requiring an unsatisfactory decision to prioritise either model performance or inference speed to define the input resolution to use. This is not the case with the discussed multi-resolution approaches. Currently, this project's models suffer from the 'garbage-in, garbage-out' characteristic attributed to the black box nature of deep learning models. The low-resolution input images, even at 1024 x 1024 pixels, are significantly blurred at the cellular level, making the identification of necessary features for cancer detection (cell density, shape irregularity, discolouring, etc.) questionable even by trained pathologists.

It may also be the case that the pipeline is inherently bottle-necked at all input resolutions by the limited amount of training samples available. Simply, there is not enough labelled training samples for the model to learn how to discern metastases within homogeneous appearing tissue, especially given that there are multiple types of cancer that the model is attempting to consolidate for classification (i.e., micro-metastases, macro-metastases, ITCs). This is an inherent issue of using WSIs as inputs in their entirety, since the size of the training data set becomes linked to the number of WSIs that can reasonably be exhaustively annotated by trained medical personnel.

Chapter 6

Conclusions

In this chapter, the accomplished objectives, project limitations, and possible areas of future work are outlined.

6.1 Project Achievements

The accomplishments of this project are the completion of the primary and secondary project objectives. In summary, a comprehensive context survey has been produced which establishes the historiography of breast cancer detection, analyses state-of-the-art techniques via the *Camelyon-16* and *Camelyon-17* challenges, and explores other relevant low-resolution methods from the wider medical image analysis literature. Then, a deep learning pipeline for binary cancer detection in low-resolution pathology images has been developed, optimised, and evaluated. Multiple techniques have been implemented to improve the general performance of the pipeline, and multiple input resolutions have been investigated. Details for a similar pipeline performing intra-slide tumour localisation are also provided.

Overall, the primary achievement of this project is in establishing the following conclusion: the developed deep learning pipeline is insufficient at performing binary cancer classification at the low input resolutions investigated, even when additional techniques were added to improve general performance. As a result, the deep learning pipeline is shown to be unsuitable for use as a form of filter to quickly classify some number of easy pathology cases before resorting to a *Camelyon* submission-like method. This negative result is disappointing but expected.

This project also demonstrated that there is an inverse relationship between classification performance and inference time when altering the input resolution used in the developed pipeline, which is a suspected intuition. This has been shown in the most extreme regard, where the lowest input resolutions result in effectively skill-less models. This relationship is not fully mapped in this project as the input resolutions investigated represent just a proportion of the possible image resolutions that can be input to a CNN for training. An optimal, higher input resolution with an acceptable balance of this relationship may exist, but this would require careful definition of what is sufficient prediction confidence and inference time for a clinically viable cancer detection system.

6.2 Limitations And Future Work

The primary limitation of this project is that the design and implementation of the low-resolution cancer detection models are mainly theoretically reasoned, as opposed to empirically investigated.[1] The developed pipeline has been justified using best practices observed with the *Camelyon* challenge submissions and relevant cancer detection models in the general field of medical image analysis. However, a robust empirical investigation of key pipeline attributes (model architecture, optimisation algorithm, stain handling, tissue detection techniques, etc.) could establish the optimal pipeline for cancer detection

[1] Whilst the design is not empirically-based, some empirical investigation contributed to the design decisions. For example, the 'Adam' optimisation strategy was selected only after training with stochastic gradient decent, and its variants using momentum, were found to be overly sensitive to the hyper-parameters.

at low-resolutions, especially given that best practices observed in the literature may not be sufficient given the context shift to classifying low-resolution instead of very high-resolution pathology images. It stands that future work should carry out empirical investigations of core pipeline aspects, allowing for the capability of the methodology used (i.e., simply reducing the high-resolution images to be sufficiently small enough that they can be input to a CNN) to be fully realised.

A secondary limitation of this project is that a single methodology for performing cancer detection is investigated. Whilst multiple techniques have been explored to improve the deep learning pipeline, the methodology is constant in that a binary cancer classification CNN is applied to resolution reduced pathology images. Possible future work has been detailed (section 3.3) about an alternative methodology that performs tumour localisation within pathology slides instead as a basis for a cancer detection pipeline. This was not prioritised in this project; if cancer detection models perform poorly on low-resolution images with limited pixel information, then models for intra-slide tumour localisation would also likely not perform well. Future work may prove the contrary.

Furthermore, this project is limited in that the developed low-resolution models perform binary classification, inferring slides to be either cancerous or normal. However, there is an intra-positive class distribution containing several cancer types (i.e., macro-metastasis, micro-metastasis, and ITCs). Future work extending the developed pipeline to perform multi-class classification for these cancer types may reveal further insights. For example, the methodology of simply reducing the WSIs to a small size and feeding the whole images as input to a CNN could lead to an effective classifier of large macro-metastases, which could be used to quickly classify cases of this type of cancer.

Finally, this project aimed to improve on the clinical viability of the *Camelyon* challenge submissions. Optimistically, this project's developed models aimed to replace them. More realistically, this project aimed to created a model that could first be applied to filter out easily classifiable cases before using some existing *Camelyon* model for the remaining difficult cases. In such a scenario, the rapidness of using a low-resolution model to filter out most cases could justify the longer inference times incurred by the high-performing *Camelyon* submissions. However, neither of these scenarios were realised, emphasising the conclusion that future work is required investigating multi-resolution approaches. These approaches encapsulate the described need for swift low-resolution processing of easy pathology cases in tandem with high-confidence, high-resolution processing for difficult pathology cases.

Chapter 7

Bibliography

Abraham, T., Todd, A., Orringer, D. A., & Levenson, R. (2021). Applications of artificial intelligence for image enhancement in pathology. *Artificial intelligence and deep learning in pathology* (pp. 119–148). Elsevier.

Akakin, H. C., & Gurcan, M. N. (2012). Content-based microscopic image retrieval system for multi-image queries. *IEEE transactions on information technology in biomedicine*, *16*(4), 758–769.

Annuscheit, J., & Hufnagl, P. (2016). *Htw-berlin, berlin, germany* [Accessed: 11 October 2021]. https://camelyon16.grand-challenge.org/Result_HTW/

Bandi, P., Geessink, O., Manson, Q., Van Dijk, M., Balkenhol, M., Hermsen, M., Bejnordi, B. E., Lee, B., Paeng, K., Zhong, A., et al. (2018). From detection of individual metastases to classification of lymph node status at the patient level: The camelyon17 challenge. *IEEE transactions on medical imaging*, *38*(2), 550–560.

Bejnordi, B. E., Litjens, G., Hermsen, M., Karssemeijer, N., & van der Laak, J. A. (2015). A multi-scale superpixel classification approach to the detection of regions of interest in whole slide histopathology images. *Medical Imaging 2015: Digital Pathology*, *9420*, 94200H.

Bejnordi, B. E., Litjens, G., Timofeeva, N., Otte-Höller, I., Homeyer, A., Karssemeijer, N., & van der Laak, J. A. (2015). Stain specific standardization of whole-slide histopathological images. *IEEE transactions on medical imaging*, *35*(2), 404–415.

Bejnordi, B. E., Veta, M., Johannes van Diest, P., van Ginneken, B., Karssemeijer, N., Litjens, G., van der Laak, J. A., Hermsen, M., Manson, Q. F., Balkenhol, M., & et al. (2017). Diagnostic assessment of deep learning algorithms for detection of lymph node metastases in women with breast cancer. *JAMA*, *318*(22), 2199–2210. https://doi.org/10.1001/jama.2017.14585

Berseth, M. (2016). *Nlp logix, jacksonville, florida, us* [Accessed: 11 October 2021]. https://camelyon16.grand-challenge.org/Result_LOGIX/

Breiman, L. (2001). Random forests. *Machine learning*, *45*(1), 5–32.

Camelyon-16 [Accessed: 19 September 2021]. (2015). https://camelyon16.grand-challenge.org/

Camelyon-17 [Accessed: 19 September 2021]. (2016). https://camelyon17.grand-challenge.org/

Cancer information summaries (pdq) [Accessed: 11 October 2021]. (2002). https://www.ncbi.nlm.nih.gov/books/NBK65951/figure/CDR0000638198__2/

Cancer survival in england - adults diagnosed [Accessed: 28 September 2021]. (2019). https://www.ons.gov.uk/peoplepopulationandcommunity/healthandsocialcare/conditionsanddiseases/datasets/cancersurvivalratescancersurvivalinenglandadultsdiagnosed

Cascetta, K. (2021). *What does it mean if breast cancer spreads to your lymph nodes?* [Accessed: 29 September 2021]. https://www.healthline.com/health/breast-cancer/breast-cancer-lymph-nodes

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). Smote: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, *16*, 321–357.

Chen, H., Lin, H.-J., Dou, Q., & Heng, P.-A. (2016). *Department of computer science and engineering, the chinese university of hong kong, sha tin, hong kong* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_CULAB/

Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine learning*, *20*(3), 273–297.

Dalal, N., & Triggs, B. (2005). Histograms of oriented gradients for human detection. *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, *1*, 886–893.

Dong, N., Kampffmeyer, M., Liang, X., Wang, Z., Dai, W., & Xing, E. (2018). Reinforced auto-zoom net: Towards accurate and fast breast cancer segmentation in whole-slide images. *Deep learning in medical image analysis and multimodal learning for clinical decision support* (pp. 317–325). Springer.

Fernández-Carrobles, M. M., Serrano, I., Bueno, G., & Déniz, O. (2016). Bagging tree classifier and texture features for tumor identification in histological images. *Procedia Computer Science*, *90*, 99–106.

Fukushima, K. (1980). A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position. *Biol. Cybern.*, *36*, 193–202.

Géron, A. (2019). *Hands-on machine learning with scikit-learn, keras, and tensorflow: Concepts, tools, and techniques to build intelligent systems*. O'Reilly Media.

Ginsburg, S. B., Lee, G., Ali, S., & Madabhushi, A. (2015). Feature importance in nonlinear embeddings (fine): Applications in digital pathology. *IEEE transactions on medical imaging*, *35*(1), 76–88.

Griffin, J., & Treanor, D. (2017). Digital pathology in clinical use: Where are we now and what is holding us back? *Histopathology*, *70*(1), 134–145. https://doi.org/10.1111/his.12993

Haralick, R. M., Shanmugam, K., & Dinstein, I. H. (1973). Textural features for image classification. *IEEE Transactions on systems, man, and cybernetics*, (6), 610–621.

Hass, C., Sanchez, U., Vasilev, I., Mey, T., & Bruni, E. (2016). *Exb research and development, germany* [Accessed: 11 October 2021]. https://camelyon16.grand-challenge.org/Result_EXBdev/

He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. *Proceedings of the IEEE conference on computer vision and pattern recognition*, 770–778.

Hering, J., & Kybic, J. (2020). Multiple instance learning via deep hierarchical exploration for histology image classification. *2020 IEEE 17th International Symposium on Biomedical Imaging (ISBI)*, 235–238.

Jorns, J. M., Visscher, D., Sabel, M., Breslin, T., Healy, P., Daignaut, S., Myers, J. L., & Wu, A. J. (2012). Intraoperative frozen section analysis of margins in breast conserving surgery significantly decreases reoperative rates: One-year experience at an ambulatory surgical center. *American journal of clinical pathology*, *138*(5), 657–669.

Kawahara, J., & Hamarneh, G. (2016). Multi-resolution-tract cnn with hybrid pretrained and skin-lesion trained layers. *International workshop on machine learning in medical imaging*, 164–171.

Kendall, A., Badrinarayanan, V., & Cipolla, R. (2015). Bayesian segnet: Model uncertainty in deep convolutional encoder-decoder architectures for scene understanding. *arXiv preprint arXiv:1511.02680*.

Khvatkov, V., & Vylegzhanin, A. (2016). *Smart imaging technologies co., us* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_Vitali2/

Kim, T., Giuliano, A. E., & Lyman, G. H. (2006). Lymphatic mapping and sentinel lymph node biopsy in early-stage breast carcinoma: A metaanalysis. *Cancer*, *106*(1), 4–16.

Kovalev, V., Kalinovsky, A., & Liauchuk, V. (2016). Deep learning in big image data: Histology image classification for breast cancer diagnosis. *Big Data and Advanced Analytics, Proc. 2nd International Conference, BSUIR, Minsk*, 44–53.

Kraus, O. (2016). *University of toronto, electrical and computer engineering, canada* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_Toronto2/

Krizhevsky, A., Sutskever, I., & Hinton, G. E. (2012). Imagenet classification with deep convolutional neural networks. *Advances in neural information processing systems*, *25*, 1097–1105.

LeCun, Y. (1998). The mnist database of handwritten digits. *http://yann.lecun.com/exdb/mnist/*.

Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., & Zitnick, C. L. (2014). Microsoft coco: Common objects in context. *European conference on computer vision*, 740–755.

Litjens, G., Bandi, P., Bejnordi, B. E., Geessink, O., Balkenhol, M., Bult, P., Halilovic, A., Hermsen, M., van de Loo, R., Vogels, R., & et al. (2018). 1399 h&e-stained sentinel lymph node sections of breast cancer patients: The camelyon dataset. *GigaScience*, *7*(6). https://doi.org/10.1093/gigascience/giy065

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A., Ciompi, F., Ghafoorian, M., van der Laak, J. A., van Ginneken, B., & Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. *Medical Image Analysis*, *42*, 60–88. https://doi.org/10.1016/j.media.2017.07.005

Liu, Y., Gadepalli, K., Norouzi, M., Dahl, G. E., Kohlberger, T., Boyko, A., Venugopalan, S., Timofeev, A., Nelson, P. Q., Corrado, G. S., et al. (2017). Detecting cancer metastases on gigapixel pathology images. *arXiv preprint arXiv:1703.02442*.

Lo, S.-C., Lou, S.-L., Lin, J.-S., Freedman, M. T., Chien, M. V., & Mun, S. K. (1995). Artificial convolution neural network techniques and applications for lung nodule detection. *IEEE transactions on medical imaging*, *14*(4), 711–718.

Lowe, D. G. (2004). Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, *60*(2), 91–110.

Madabhushi, A., & Lee, G. (2016). Image analysis and machine learning in digital pathology: Challenges and opportunities. *Medical Image Analysis*, *33*, 170–175. https://doi.org/10.1016/j.media.2016.06.037

Maksoud, S., Zhao, K., Hobson, P., Jennings, A., & Lovell, B. C. (2020). Sos: Selective objective switch for rapid immunofluorescence whole slide image classification. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3862–3871.

Martin, D. T. (2016). *Camelyon challenge 2016 - detection of metastases in sentinel lymph nodes* [Accessed: 13 October 2021]. https://grand-challenge-public-prod.s3.amazonaws.com/f/challenge/65/25d5c40f-3210-4119-8a36-5e32e61a1fe7/Camelyon16_Radboudumc.pptx

Martin, L. J. (2020). *Breast biopsy: Procedures, risks, recovery, and results* [Accessed: 28 September 2021]. https://www.webmd.com/breast-cancer/breast-biopsy

Meyer, F. (1994). Topographic distance and watershed lines. *Signal processing*, *38*(1), 113–125.

Mormont, R., Geurts, P., & Marée, R. (2018). Comparison of deep transfer learning strategies for digital pathology. *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, 2262–2271.

Munjal, B., George, A., Albarqouni, S., Demirci, S., & Navab, N. (2016). *Technische universitat munchen, computer aided medical procedure (camp), munich, germany* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_CAMP2/

Ojala, T., Pietikainen, M., & Maenpaa, T. (2002). Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on pattern analysis and machine intelligence*, *24*(7), 971–987.

Öner, M. Ü. (2016). *Metastasis detection and localization in lypmh nodes by using convolutional neural networks* (Doctoral dissertation). Middle East Technical University.

Otsu, N. (1979). A threshold selection method from gray-level histograms. *IEEE transactions on systems, man, and cybernetics*, *9*(1), 62–66.

Pantanowitz, L. (2010). Digital images and the future of digital pathology. *Journal of pathology informatics*, *1*.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., ... Chintala, S. (2019). Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, & R. Garnett (Eds.), *Advances in neural information processing systems 32* (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf

Patil, S. M., Tong, L., & Wang, M. D. (2020). Generating region of interests for invasive breast cancer in histopathological whole-slide-image. *2020 IEEE 44th Annual Computers, Software, and Applications Conference (COMPSAC)*, 723–728.

Phoulady, H. A. (2016). *University of south florida, tampa, usa* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_Phoulady/

Reinhard, E., Adhikhmin, M., Gooch, B., & Shirley, P. (2001). Color transfer between images. *IEEE Computer graphics and applications*, *21*(5), 34–41.

Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *International Conference on Medical image computing and computer-assisted intervention*, 234–241.

Rother, C., Kolmogorov, V., & Blake, A. (2004). "grabcut" interactive foreground extraction using iterated graph cuts. *ACM transactions on graphics (TOG)*, *23*(3), 309–314.

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., Huang, Z., Karpathy, A., Khosla, A., Bernstein, M., et al. (2015). Imagenet large scale visual recognition challenge. *International journal of computer vision*, *115*(3), 211–252.

Russell, S., & Norvig, P. (2020). *Artificial intelligence: A modern approach*. Pearson.

Saha, S. (2018). *A comprehensive guide to convolutional neural networks - the eli5 way* [Accessed: 1 October 2021]. https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

Sentinel lymph node biopsy [Accessed: 11 October 2021]. (2019). https://www.cancer.gov/publications/dictionaries/cancer-terms/def/sentinel-lymph-node-biopsy

Sethi, A., Sha, L., Vahadane, A. R., Deaton, R. J., Kumar, N., Macias, V., & Gann, P. H. (2016). Empirical comparison of color normalization methods for epithelial-stromal classification in h and e images. *Journal of pathology informatics*, *7*.

Shaban, M., Qaiser, T., Awan, R., Sirinukunwattana, K., Tsang, Y.-W., & Rajpoot, N. (2016). *Camelyon challenge contest: The warwick-qu approach* [Accessed: 13 October 2021]. https://grand-challenge-public-prod.s3.amazonaws.com/f/challenge/65/0440cbb8-2810-4231-8f29-8ef8584a29d4/Camelyon16_isbi_WarwickQU.pdf

Shahidi, F. (2021). Breast cancer histopathology image super-resolution using wide-attention gan with improved wasserstein gradient penalty and perceptual loss. *IEEE Access*, *9*, 32795–32809.

Simonyan, K., & Zisserman, A. (2014). Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*.

Sutton, R. S., & Barto, A. G. (2018). *Reinforcement learning: An introduction*. MIT press.

Szegedy, C., Liu, W., Jia, Y., Sermanet, P., Reed, S., Anguelov, D., Erhan, D., Vanhoucke, V., & Rabinovich, A. (2015). Going deeper with convolutions. *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. https://doi.org/10.1109/cvpr.2015.7298594

Tokunaga, H., Teramoto, Y., Yoshizawa, A., & Bise, R. (2019). Adaptive weighting multi-field-of-view cnn for semantic segmentation in pathology. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 12597–12606.

Valkonen, M., Kartasalo, K., Liimatainen, K., Nykter, M., Latonen, L., & Ruusuvuori, P. (2017). Metastasis detection from whole slide images using local features and random forests. *Cytometry Part A*, *91*(6), 555–565.

van Diest, P. J., van Deurzen, C. H., & Cserni, G. (2010). Pathology issues related to sn procedures and increased detection of micrometastases and isolated tumor cells. *Breast Disease*, *31*(2), 65–81. https://doi.org/10.3233/bd-2010-0298

Venâncio, R., Cheikh, B. B., Coron, A., & Racoceanu, D. (2016). *Sorbonne universites, upmc univ paris 06, cnrs, inserm, laboratoire imagerie biomedicale (lib), paris, france* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_Sorbonne/

Vestjens, J., Pepels, M., de Boer, M., Borm, G. F., van Deurzen, C. H., van Diest, P. J., Van Dijck, J., Adang, E., Nortier, J. W., Rutgers, E. T., et al. (2012). Relevant impact of central pathology review on nodal classification in individual breast cancer patients. *Annals of oncology*, *23*(10), 2561–2566.

Veta, M., Heng, Y. J., Stathonikos, N., Bejnordi, B. E., Beca, F., Wollmann, T., Rohr, K., Shah, M. A., Wang, D., Rousson, M., & et al. (2019). Predicting breast tumor proliferation from whole-slide images: The tupac16 challenge. *Medical Image Analysis*, *54*, 111–121. https://doi.org/10.1016/j.media.2019.02.012

Wang, D., Khosla, A., Gargeya, R., Irshad, H., & Beck, A. H. (2016). Deep learning for identifying metastatic breast cancer. *arXiv preprint arXiv:1606.05718*.

Wang, H., Roa, A. C., Basavanhally, A. N., Gilmore, H. L., Shih, N., Feldman, M., Tomaszewski, J., Gonzalez, F., & Madabhushi, A. (2014). Mitosis detection in breast cancer pathology images by combining handcrafted and convolutional neural network features. *Journal of Medical Imaging*, *1*(3), 034003.

Watanabe, S., Seno, S., Takenaka, Y., & Matsuda, H. (2016). *Osaka university, department of bioinformatic engineering* [Accessed: 11 October 2021]. https://camelyon16.grand-challenge.org/Result_Osaka/

Wong, Q. (2016). *Independent participant - quincy wong* [Accessed: 11 October 2021]. https://grand-challenge-public-prod.s3.amazonaws.com/f/challenge/65/58865d9b-841b-42d7-901f-94a08c3ea8a3/Camelyon16_QuincyWong.pdf

Xu, T. (2016). *Deepcare inc.* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_DeepCare/

Yue, W., Wang, Z., Chen, H., Payne, A., & Liu, X. (2018). Machine learning with applications in breast cancer diagnosis and prognosis. *Designs*, *2*(2), 13. https://doi.org/10.3390/designs2020013

Zhong, A., & Li, Q. (2016). *Gordon center for medical imaging, massachusetts general hospital, harvard medical school, usa* [Accessed: 13 October 2021]. https://camelyon16.grand-challenge.org/Result_MGH3/

Zhu, X., & Goldberg, A. B. (2009). Introduction to semi-supervised learning. *Synthesis lectures on artificial intelligence and machine learning*, *3*(1), 1–130.

Zormpas-Petridis, K., Noguera, R., Ivankovic, D. K., Roxanis, I., Jamin, Y., & Yuan, Y. (2021). Superhistopath: A deep learning pipeline for mapping tumor heterogeneity on low-resolution whole-slide digital histopathology images. *Frontiers in oncology*, *10*, 3052.

Chapter 8

Appendices

8.1 Ethical Approval Document

Figure 8.1: Ethics approval letter from the University of St Andrews Teaching and Research Ethics Committee.

University of St Andrews | FOUNDED 1413

School of Computer Science Ethics Committee

18 November 2021

Dear Nathan,

Thank you for submitting your ethical application which was considered by the School Ethics Committee.

The School of Computer Science Ethics Committee, acting on behalf of the University Teaching and Research Ethics Committee (UTREC), has approved this application:

Approval Code:	CS15846	**Approved on:**	18.11.2021	**Approval Expiry:**	18.11.2026
Project Title:	Breast Cancer Detection in Low Resolution Images				
Researcher(s):	Nathan Poole				
Supervisor(s):	Ms Christina Fell and Dr David Harris-Birtill				

The following supporting documents are also acknowledged and approved:

1. Application Form

Approval is awarded for 5 years, see the approval expiry data above.

If your project has not commenced within 2 years of approval, you must submit a new and updated ethical application to your School Ethics Committee.

If you are unable to complete your research by the approval expiry date you must request an extension to the approval period. You can write to your School Ethics Committee who may grant a discretionary extension of up to 6 months. For longer extensions, or for any other changes, you must submit an ethical amendment application.

You must report any serious adverse events, or significant changes not covered by this approval, related to this study immediately to the School Ethics Committee.

Approval is given on the following conditions:

- that you conduct your research in line with:
  - the details provided in your ethical application
  - the University's Principles of Good Research Conduct
  - the conditions of any funding associated with your work
- that you obtain all applicable additional documents (see the 'additional documents' webpage for guidance) before research commences.

You should retain this approval letter with your study paperwork.

Yours sincerely,

Wendy Boyter

SEC Administrator

School of Computer Science Ethics Committee
Dr Juan Ye/Convenor, Jack Cole Building, North Haugh, St Andrews, Fife, KY16 9SX
Telephone: 01334 463252 Email: ethics-cs@st-andrews.ac.uk
The University of St Andrews is a charity registered in Scotland: No SC013532

8.2 *Camelyon-16* Teams

Table 8.1: Table relating team pseudonyms with their full team names and citations.

Team Pseudonym	Challenge Team Name
Camp-Tum	Technical University of Munich (CAMP), Germany (Munjal et al., 2016)
CULab	The Chinese University of Hong Kong (CU lab), Hong Kong (Chen et al., 2016)
DeepCare	DeepCare Inc, China (Xu, 2016)
ExB	EXB Research and Development co., Germany (Hass et al., 2016)
HMS-MGH	Harvard Medical School, Gordon Center for Medical Imaging, MGH (Zhong & Li, 2016)
HMS-MIT	Harvard Medical School and MIT (D. Wang et al., 2016)
HTW	HTW-BERLIN, Germany (Annuscheit & Hufnagl, 2016)
LIB	Sorbonne Universites, Laboratoire d'Imagerie Biomdicale, France (Venâncio et al., 2016)
METU	Middle East Technical University, Departments of EEE, NSNT and HS, Turkey (Öner, 2016)
Minsk	United Institute of Informatics Problems, Belarus (Kovalev et al., 2016)
NLPLogix	NLP LOGIX co., USA (Berseth, 2016)
OsakaUniv	Osaka University, Department of Bioinformatic Engineering, Japan (Watanabe et al., 2016)
QuincyWong	Independent participant, Germany (Wong, 2016)
RadboudUMC	Radboud University Medical Center (DIAG), Netherlands (D. T. Martin, 2016)
SmartImaging	Smart Imaging Technologies co., USA (Khvatkov & Vylegzhanin, 2016)
Tampere	BioMediTech, University of Tampere, Finland (Valkonen et al., 2017)
UnivSFlorida	University of South Florida, Computer Science and Engineering, USA (Phoulady, 2016)
UnivToronto	University of Toronto, Electrical and Computer Engineering, Canada (Kraus, 2016)
VisiLab	VISILAB, University of Castilla-La Mancha, Spain (Fernández-Carrobles et al., 2016)
Warwick-QU	The Warwick-QU Team, United Kingdom (Shaban et al., 2016)

8.3 *Camelyon-17* Teams

Table 8.3: Table relating team pseudonyms with their full team names, as used in the *Camelyon-17* challenge. Knowledge of team submissions was acquired through the leader-board page of the challenge website ("Camelyon-17", 2016).

Team Pseudonym	**Challenge Team Name**
Lunit	Lunit Inc.
HMS-MGH-CCDS	Harvard Medical School, Mass. General Hospital, Center for Clinical Data Science
VCA-TUe	Electrical Engineering Department, Eindhoven University of Technology
MIL-GPAT	The University of Tokyo, Tokyo Medical and Dental University
Indica	Indica Labs
Chengshenghua	Huazhong University of Science and Technology, Britton Chance Center for Biomedical Photonics
DTU	Technical University of Denmark
IMT-CUHK	Imsight Medical Technology, Chinese University of Hong Kong, Xiamen University
Desuto	ContextVision
METU-Vision	Middle East Technical University
Proscia	Proscia Inc., Carnegie Mellon University, Moffit Cancer Center
ML-KA	Karlsruhe Institute of Technology

8.4 User Manual

This section provides instructions to execute the developed `Python` scripts which implement the deep learning pipeline. The environment requirements to permit program execution are specified. First, execution of the supplementary tools is explained, which carry out one-time prior processing on the WSIs expected by the model training and inference implementations. Then, execution of model training and inference is explained.

8.4.1 Environment Requirements

`Python` version 3.0 or higher is required. Additional `Python` packages are required which are either default `Python` packages (e.g., `os`, `sys`, and `numpy`) or external packages requiring installation. Required external packages are specified in the provided `requirements.txt` file but are listed as follows for convenience: `openslide-python`, `torch`, `torchvision`, `scikit-image`, `tensorboard`, `scikit-learn`, and `ray[tune]`. All of these packages can be installed simply using `pip`. A GPU device and `CUDA` support to allow for GPU acceleration is highly recommended but not required.

8.4.2 Instructions - Supplementary Tools

Supplementary tools are provided in the `src/tools/` directory. The supplementary tools aim to convert the original *Camelyon-16* and *Camelyon-17* data sets to inputs usable by the model training and inference implementations. These data sets are required to replicate all of the deep learning pipeline stages. However, due to the size of these data sets (hundreds of gigabytes each), relevant low-resolution data set versions are provided in the `data/` directory.

The supplementary tools are numbered according to their recommended order of execution as each tool alters the input data set. The scripts themselves contain documentation as to their purpose and usage, but this is provided in the following sections for completeness.

- `1_generate_lowres_images.py` - see subsubsection 8.4.2.1
- `2_generate_image_labels.py` - see subsubsection 8.4.2.2
- `3_filter_images_black_content.py` - see subsubsection 8.4.2.3
- `4_generate_tissue_images.py` - see subsubsection 8.4.2.4
- `5_generate_images_mean_std.py` - see subsubsection 8.4.2.5

8.4.2.1 Generating Low-Resolution Image Data Set

The '`1_generate_lowres_images.py`' tool converts a directory of high-resolution `.TIF` images (from the *Camelyon-16* and *Camelyon-17* data sets) to low-resolution `.PNG` images of a specified size. From this, training, validation, and testing partitions can be created as described in the whole-slide classification design. The following instructions define usage of this tool:

```
python3 1_generate_lowres_images.py <width|-l> <height|res_level> <input_dir>
<output_dir>
```

- Use '<`width`> <`height`>' to specify desired image dimensions in pixels. Note that these dimensions are considered as maximums since the conversion will maintain the original aspect ratio of the slide images at high-resolution.
- Alternatively, use '`-l` <`res_level`>' to specify one of the default resolution levels (i.e., degree of magnification) between 0 (highest resolution) to 8 (lowest resolution) inclusive (see Table 2.3).
- <`input_dir`> is a path to a directory containing high-resolution `.TIF` images to be converted.
- <`output_dir`> is a path to a directory to save the generated low-resolution images to.

8.4.2.2 Renaming Low-Resolution Images To Include Labels

The '`2_generate_image_labels.py`' tool will rename a directory of images to include their associated output label, as determined by a provided CSV file. This tool was initially implemented for convenience of model debugging and analysis but became necessary when the *Camelyon-17* WSIs were added to the data set used; the training images in *Camelyon-17* do not contain labels (i.e., 'tumor' or 'normal') as is the case with the *Camelyon-16* WSIs. This tool should be used to rename all images in the input data set (i.e., the training, validation, and testing subsets) and may therefore have to be executed multiple times for different sub-directories of images in the data set. The following instructions define the usage of this tool:

```
python3 2_generate_image_labels.py -i <input_dir> -c <csv_file> [-h]
```

- <input_dir> is a path to a directory containing the low-resolution `.PNG` images to rename with their associated labels.
- <csv_file> is a path to a CSV file containing the file names in the input directory and their associated labels which can be extracted for renaming. Such CSV files are provided with the *Camelyon-16* and *Camelyon-17* data sets.
- `-h` is an optional flag indicating to provide help with program execution.

8.4.2.3 Filtering Black Content Within The Low-Resolution Images

The '`3_filter_images_black_content.py`' tool is used to handle the fact that *Camelyon-16* WSIs have a white background, whilst the *Camelyon-17* WSIs mostly have a black background. This tool will apply black content filtering to a directory of images, which converts black pixels to white pixels, and save the resulting images. This tool should be used to convert all of the renamed low-resolution WSIs in the data set, effectively overwriting the original low-resolution `.PNG` images. The resulting images of this tool are used directly as input to model versions 0 and 1. Usage of the tool is defined as follows:

```
python3 3_filter_images_black_content.py -i <input_dir> -o <output_dir> [-h]
```

- <input_dir> is a path to a directory containing low-resolution `.PNG` images to perform black content filtering on.
- <output_dir> is a path to a directory to save the resulting filtered images to.
- `-h` is an optional flag indicating to provide help with program execution.

8.4.2.4 Generating Low-Resolution Tissue Detected Images

The '`4_generate_tissue_images.py`' tool is used to generate a modified version of the low-resolution WSI data set, where all of the WSIs have had tissue detection performed on them. The resulting low-resolution images contain just the identified tissue regions from within the slides and are used as inputs to model versions 2 and 3. This tool should be executed on all WSIs in the data set (i.e., training, validation, and testing subsets) to allow for use of versions 2 and 3 of whole-slide classification model training and inference (may have to be executed multiple times when sub-directories are present). The following instructions define usage of this tool:

```
python3 4_generate_tissue_images.py -i <input_dir> -o <output_dir> [-c] [-h]
```

- <input_dir> is a path to a directory containing the low-resolution `.PNG` images to perform tissue detection on.
- <output_dir> is a path to a directory to save the generated tissue detected images to.
- `-c` is an optional flag indicating to produce side-by-side comparison images (as opposed to just the tissue extracted images) such that the results of the tissue detection process can be visually verified.
- `-h` is an optional flag indicating to provide help with program execution.

8.4.2.5 Calculating Data Set Mean And Standard Deviation

The '5_generate_images_mean_std.py' tool was used to calculate the per channel (low-resolution images have RGB format) mean and standard deviation across the training images. Specifically, this tool was executed on the training images resulting from the '3_filter_images_black_content.py' tool so that the input images for model versions 0 and 1 could be normalised. The tool was also executed for the tissue detected training images so that the input images for model versions 2 and 3 could be normalised. The resulting per channel means and standard deviations were added to the classification data set implementation so that recalculation did not have to occur each time a model was trained or used for inference. Use of the tool is defined as follows:

```
python3 5_generate_images_mean_std.py -i <input_dir> -width <image_width>
-height <image_height> [-h]
```

- <input_dir> is a path to a directory containing training .PNG images to calculate the per channel mean and standard deviation for.
- <image_width> is the desired width to convert images to before calculating the per channel mean and standard deviation.
- <image_height> is the desired height to convert images to before calculating the per channel mean and standard deviation. The mean and standard deviation of the training images will change according to how they are resized as model input, which is determined by the target input resolution being investigated (initially 299 x 299 pixels).
- -h is an optional flag indicating to provide help with program execution.

8.4.3 Instructions - Model Training And Inference

Having performed the required one-time prior processing stages explained above, there should be two data set versions available for model training and inference. The first contains training, validation, and testing partitions of low-resolution (black content filtered) WSIs (all renamed to include their associated output label) for use with whole-slide classification model versions 0 and 1. The second contains training, validation, and testing partitions of low-resolution tissue images (all renamed to include their associated output label) for use with model versions 2 and 3.

The following sections explain usage of the training and inference implementations which make use of the described data sets to create and evaluate low-resolution whole-slide classification models. Note that the following data set directory structure is expected:

```
root/
├── train/
│   ├── ...
│   └── image_N[_label].png
├── eval/
│   ├── ...
│   └── image_N[_label].png
└── test/
    ├── ...
    └── test_N[_label].png
```

8.4.3.1 Model Training

The 'classification_train.py' script is used for training a whole-slide classification model using the developed low-resolution image data sets according to the outlined model design. Usage of this script is defined as follows:

```
python3 classification_train.py -i <input_dir> -o <output_file> -a <augment_level>
[-hpo] [-h] [-r <input_res>]
```

- `-i <input_dir>` is a path to a directory containing all input data (i.e., contains 'train', 'eval', and 'test' sub-directories of data set images).
- `-o <output_file>` is a path to an output file to save the trained model to.
- `-a <augment_level>` is an integer specifying the data augmentation level (i.e., model version) to use.
  - 0 = No data augmentation using the low-resolution (black filtered) WSIs.
  - 1 = Data augmentation (i.e., flipping, rotation, and colour jitter) using the low-resolution (black filtered) WSIs (`<input_dir>` should be a path to such images).
  - 2 = No data augmentation using the tissue detected low-resolution images.
  - 3 = Data augmentation using the tissue detected low-resolution images (`<input_dir>` should be a path to such images).
- `-hpo` is an optional flag indicating to train using automated hyper-parameter optimisation.
- `-h` is an optional flag indicating to provide help with program execution.
- `-r <input_res>` is an optional argument specifying the model input image resolution to use, which is 299 x 299 pixels by default. It should be specified as one of 299, 512, 1024, or 2048.

8.4.3.2 Model Inference

The '`classification_infer.py`' script is used for inferring a directory of images (e.g., the test set for evaluating the model). Usage of this script is defined as follows:

```
python3 classification_infer.py -i <input_dir> -m <model_file> -o <output_file>
-a <augment_level> [-v] [-h] [-r <input_res>]
```

- `-i <input_dir>` is a path to a root directory which itself contains a 'test' sub-directory of images to infer.
- `-m <model_file>` is a path to a file containing the trained model to use for inference.
- `-o <output_file>` is a path to a file to save the image file names (from `<input_dir>`) and corresponding model predictions to in CSV format.
- `-a <augment_level>` is an integer specifying the data augmentation level (i.e., model version) used when the model was trained.
  - 0 = No data augmentation using the low-resolution (black filtered) WSIs.
  - 1 = Data augmentation (i.e., flipping, rotation, and colour jitter) using the low-resolution (black filtered) WSIs (`<input_dir>` should be a path to such images).
  - 2 = No data augmentation using the tissue detected low-resolution images.
  - 3 = Data augmentation using the tissue detected low-resolution images (`<input_dir>` should be a path to such images).
- `-v` is an optional flag indicating to instead infer the validation subset (i.e., 'eval' sub-directory) of images.
- `-h` is an optional flag indicating to provide help with program execution.
- `-r <input_res>` is an optional argument specifying the model input image resolution expected. It should be one of 299, 512, 1024, or 2048.

8.5 Testing Summary

Manual module testing was performed to ensure the correctness of the fundamental components of the deep learning systems (and supplementary tools). Most low-level functionality is carried out by third-party libraries; thus, testing seeks to verify at a high-level that these functionalities are executed as intended.

Test cases are provided for the core components of the project in the following sections and have been selected to demonstrate correctness at a high-level without having to individually provide all test cases for all methods. To this end, screenshots of program behaviour are given as evidence.

8.5.1 Auxiliary Tools Testing

This section outlines the high-level correctness testing performed on the auxiliary scripts created to aid development of the low-resolution models.

8.5.1.1 Generating Low-Resolution Images

The following details testing performed on the implementation for generating low-resolution images from the high-resolution *Camelyon* data set WSIs: `generate_lowres_images.py` (see Table 8.5).

Table 8.5: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that program arguments must be given as expected so that image conversion can take place correctly.	Figure 8.2
Test that a single high-resolution input image is correctly converted to a low-resolution image of a specified resolution level.	Figure 8.3
Test that a single high-resolution input image is correctly converted to a low-resolution image of specified width and height.	Figure 8.4
Test that all images in a given input directory are automatically converted and saved to a specified output directory.	Figure 8.5

Figure 8.2: The following screenshot shows that a valid width and height, or valid resolution level, accompanied by an existing input image directory must be provided to `generate_lowres_images.py` for image conversion to occur.

```
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py
Invalid number of arguments given.
Usage: 'python3 generate_lowres_images <width|-l> <height|res_level> <input_dir> <output_dir>'.
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py -l not_a_res_level ../testdata/ ../testdata/
Error: Resolution level must be an integer.
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py -l -1 ../testdata/ ../testdata/
Error: Invalid resolution level requested - must be between 0 and 9 inclusive.
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py -l 8 not_a_dir ../testdata/
Error: Given input directory does not exist.
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py not_int not_int ../testdata/ ../testdata/
Error: Width and height must be an integer.
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py -1 -1 ../testdata/ ../testdata/
Error: Invalid width/height provided - must be greater then 0.
```

Figure 8.3: The following screenshot illustrates a high-resolution `.TIF` image (‘*normal_001.png*’) being correctly converted to a low-resolution image at resolution level 8. In this example, the output image is 382 x 864 pixels in size at level 8.

```
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py -l 8 ../testdata/ ../testdata/
Generating Low-Resolution Images:
-----------------------------------------------------------------------------------------------
Images Generated At Resolution Level:  8
-----------------------------------------------------------------------------------------------
Working On Slide: ' normal_001.tif '.
Saved ' normal_001.png ' to ' ../testdata/ ' (Width:  382 , Height:  864 ).
```

Figure 8.4: The following screenshot illustrates a high-resolution `.TIF` image (‘*normal_001.png*’) being correctly converted to a low-resolution image with height and width specified as 500 pixels. The output image is 221 x 500 pixels wide, which is expected as resizing the images uses the given height and width as maximum values so that aspect ratio is preserved.

```
np57@a6110cb757b8:/data/CS5199/src$ python3 generate_lowres_images.py 500 500 ../testdata/ ../testdata/
Generating Low-Resolution Images:
-----------------------------------------------------------------------------------------------------------
Images Generated As Thumbnails With Dimensions (Width, Height):  500 ,  500
-----------------------------------------------------------------------------------------------------------
Working On Slide: ' normal_001.tif '.
Saved ' normal_001.png ' to ' ../testdata/ ' (Width:  221 , Height:  500 ).
```

Figure 8.5: The following screenshot illustrates that conversion from high- to low-resolution images can be automatically carried out for a directory of many input images. In the screenshot, some images (‘*normal_155.png*’, ‘*normal_022.png*’, and ‘*normal_078.png*’) have been converted and saved, with other input images being processed (the screenshot is mid-execution).

```
np57@52a2f7ecc1e8:/data/np57/CS5199/src$ python3 generate_lowres_images.py -l 8 ../../../datasets/camelyon16/raw/training/normal/
Output directory (' ../testdata/testnormal/ ') does not exist - creating directory.
Generating Low-Resolution Images:
-----------------------------------------------------------------------------------------------
Images Generated At Resolution Level:  8
-----------------------------------------------------------------------------------------------
Working On Slide: ' normal_155.tif '.
Saved ' normal_155.png ' to ' ../testdata/testnormal/ ' (Width:  448 , Height:  308 ).
Working On Slide: ' normal_022.tif '.
Saved ' normal_022.png ' to ' ../testdata/testnormal/ ' (Width:  382 , Height:  852 ).
Working On Slide: ' normal_078.tif '.
Saved ' normal_078.png ' to ' ../testdata/testnormal/ ' (Width:  354 , Height:  838 ).
```

8.5.1.2 Generating Output Labels In File Names For An Image Directory

The following details testing performed on the implementation for renaming files (i.e., images) such that they include their expected output label: `generate_image_labels.py` (see Table 8.7).

Table 8.7: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that valid program arguments must be given as expected for file name relabelling to occur.	Figure 8.6
Test that correct labels are extracted for a set of images in the input directory.	Figure 8.7
Test that correct labels are added to the file name of the input images.	Figure 8.8

Figure 8.6: The following screenshot shows that a valid directory (containing `.PNG` images to be renamed) must be provided, as well as the corresponding CSV file (as given by the *Camelyon* challenges) containing the output labels for said images.

```
nmpoole@Nathans-MBP tools % python3 generate_image_labels.py
usage: generate_image_labels.py [-h] -i INPUT_DIR -c CSV_FILE
generate_image_labels.py: error: the following arguments are required: -i/--input_dir, -c/--csv_file
nmpoole@Nathans-MBP tools % python3 generate_image_labels.py -i not_a_dir -c not_a_csv
Error: Given input directory does not exist.
nmpoole@Nathans-MBP tools % python3 generate_image_labels.py -i ../../testdata/test/ -c not_a_csv
Error: Invalid CSV file given.
nmpoole@Nathans-MBP tools % python3 generate_image_labels.py -i ../../testdata/test/ -c ../../testdata/test/reference.csv
Renaming Images To Include Labels:
```

Figure 8.7: The following screenshot shows that correct labels are extracted from the given CSV file for the input directory images. For a set of images (‘*test_001.png*’, ‘*test_002.png*’, and ‘*test_003.png*’), the corresponding row from the CSV file is shown, along with the label that has been identified for inclusion in the file renaming. Note that finer labels (e.g., ‘micro’ and ’macro’) are used instead of the more general binary labels (e.g., ‘tumor’) since this will afford better model performance analysis, though these different labels are considered semantically equivalent by the developed models.

```
Iterating CSV File: ../../testdata/test/reference.csv.
File: 'test_001', CSV Row: '['test_001', 'Tumor', 'IDC', 'Macro']', Label Extracted: 'macro'.
File: 'test_002', CSV Row: '['test_002', 'Tumor', 'ILC', 'Macro']', Label Extracted: 'macro'.
File: 'test_003', CSV Row: '['test_003', 'Normal', 'DCIS', 'None']', Label Extracted: 'none'.
```

Figure 8.8: The following screenshot shows that identified labels for images within the input directory are added to the file names of the images, as expected.

```
Input Directory ('../../testdata/test/') Num PNG Images: 3.
-----------------------------------------------------------------------------------------------
Iterating CSV File: ../../testdata/test/reference.csv.
    File Found In Input Dir: 'test_001'. Renaming With Label: 'macro'
        Renamed '../../testdata/test/test_001.png' To '../../testdata/test/test_001_macro.png'.
    File Found In Input Dir: 'test_002'. Renaming With Label: 'macro'
        Renamed '../../testdata/test/test_002.png' To '../../testdata/test/test_002_macro.png'.
    File Found In Input Dir: 'test_003'. Renaming With Label: 'none'
        Renamed '../../testdata/test/test_003.png' To '../../testdata/test/test_003_none.png'.
-----------------------------------------------------------------------------------------------
Finished Renaming: 3 Files Renamed (i.e., Labelled).
-----------------------------------------------------------------------------------------------
```

8.5.1.3 Filtering Black Image Content

The following details testing performed on the implementation for filtering out black image content (i.e., converting black backgrounds to white) from the low-resolution WSIs: `filter_images_black_content.py` (see Table 8.9).

Table 8.9: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that program arguments must be given as expected so that image black pixel filtering can take place correctly.	Figure 8.9
Test that filtering of black pixels and conversion to white background is as expected for a single image.	Figure 8.10
Test that filtering can be applied to multiple images from a directory.	Figure 8.11

Figure 8.9: The following screenshot shows that a valid input directory of images must be provided to the program. A valid output directory may also be given, but is created if it does not exist.

```
nmpoole@Nathans-MBP tools % python3 3_filter_images_black_content.py
usage: 3_filter_images_black_content.py [-h] -i INPUT_DIR -o OUTPUT_DIR
3_filter_images_black_content.py: error: the following arguments are required: -i/--input_dir, -o/--output_dir
nmpoole@Nathans-MBP tools % python3 3_filter_images_black_content.py -i not_a_dir -o ../../testdata/test_filtering/
Error: Given input directory does not exist.
nmpoole@Nathans-MBP tools % python3 3_filter_images_black_content.py -i ../../testdata/train/ -o ../../testdata/test_filtering/
Output directory (' ../../testdata/test_filtering/ ') does not exist - creating directory.
```

Figure 8.10: Demonstration of the black background/content filtering on an example image, '*patient_000_node_0.png*'. The original low-resolution image (left) has its background converted (right) as expected. Errant thin dark lines persist, which could not be effectively removed (discussed in the design section).

Figure 8.11: The following screenshot shows that the desired filtering can be carried out on multiple images from the specified input directory, and the results are saved to the output directory given.

```
Output directory (' ../../testdata/test_filtering/ ') does not exist - creating directory.
Saved 'normal_003.png' to '../../testdata/test_filtering/' (Total: 0).
Saved 'normal_002.png' to '../../testdata/test_filtering/' (Total: 1).
Saved 'normal_001.png' to '../../testdata/test_filtering/' (Total: 2).
Saved 'tumor_002.png' to '../../testdata/test_filtering/' (Total: 3).
Saved 'tumor_003.png' to '../../testdata/test_filtering/' (Total: 4).
Saved 'tumor_001.png' to '../../testdata/test_filtering/' (Total: 5).
Saved 'tumor_009.png' to '../../testdata/test_filtering/' (Total: 6).
Average Conversion Time (h:m:s) = 0:00:00.346243
```

8.5.1.4 Generating Tissue Detected Images

The following details testing performed on the implementation for generating tissue detected images from the low-resolution WSIs: `generate_tissue_images.py` (see Table 8.11).

Table 8.11: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that program arguments must be given as expected so that image tissue detection can take place correctly.	Figure 8.12
Test that tissue detection is performed as expected for a single input image.	Figure 8.13
Test that tissue detection is performed as expected on several images from a directory.	Figure 8.14
Test that tissue detected images are saved to the indicated output directory.	Figure 8.14
Test that comparison images between the original and tissue detected images are produced when specified.	Figure 8.15
Test that average conversion time for a set of images is reported as expected.	Figure 8.16

Figure 8.12: The following screenshot shows that a valid input directory of images must be provided to the program. A valid output directory may also be given, but is created if it does not exist. The optional argument, '*c*' may be given, indicating to generate comparison images so the tissue detection results can be verified.

```
nmpoole@Nathans-MBP tools % python3 generate_tissue_images.py
usage: generate_tissue_images.py [-h] -i INPUT_DIR -o OUTPUT_DIR [-c]
generate_tissue_images.py: error: the following arguments are required: -i/--input_dir, -o/--output_dir
nmpoole@Nathans-MBP tools % python3 generate_tissue_images.py -i not_a_dir -o not_a_dir
Error: Given input directory does not exist.
nmpoole@Nathans-MBP tools % python3 generate_tissue_images.py -i ../../testdata/train -o not_a_dir
Output directory (' not_a_dir ') does not exist - creating directory.
```

Figure 8.13: The following screenshot shows the result of running tissue detection on a single image, '*tumor_009.png*' (far left). The tissue detection process functions as described in the design process, yielding good tissue segmentation in this case (far right).

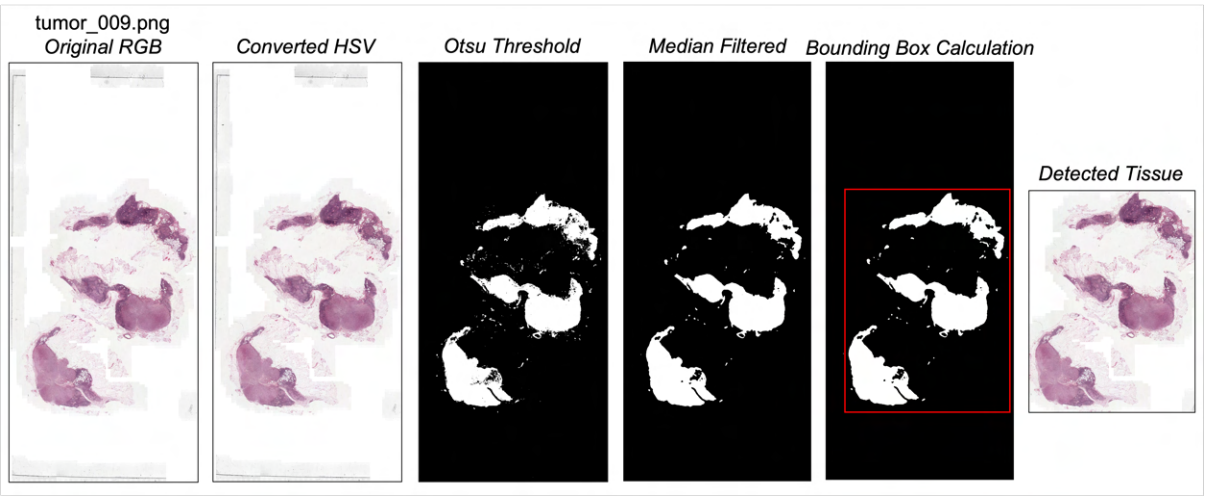


Figure 8.14: The following screenshot shows that the tissue detection process is executed on each (`.PNG`) image within the given input directory. The resulting images have their names changed to indicate that they represent just the tissue region extracted from the original slide, and are saved to the given output directory.

```
nmpoole@Nathans-MBP tools % python3 generate_tissue_images.py -i ../../testdata/train/ -o ../../testdata/train_tissue
Saved 'normal_003_tissue.png' to '../../testdata/train_tissue' (Total: 1).
Saved 'normal_002_tissue.png' to '../../testdata/train_tissue' (Total: 2).
Saved 'normal_001_tissue.png' to '../../testdata/train_tissue' (Total: 3).
Saved 'tumor_002_tissue.png' to '../../testdata/train_tissue' (Total: 4).
Saved 'tumor_003_tissue.png' to '../../testdata/train_tissue' (Total: 5).
Saved 'tumor_001_tissue.png' to '../../testdata/train_tissue' (Total: 6).
Saved 'tumor_009_tissue.png' to '../../testdata/train_tissue' (Total: 7).
```

Figure 8.15: The following screenshots show that comparison images can be generated for a directory of (`.PNG`) images (top). In this case, a grid image comprising the original and tissue detected image is created for each image in the input directory, allowing for immediate side-by-side comparisons and result verification (bottom screenshot shows the comparison image generated for '*normal_001.png*').

```
nmpoole@Nathans-MBP tools % python3 generate_tissue_images.py -i ../../testdata/train/ -o ../../testdata/train_tissue -c
Saved 'normal_003_tissue_compare.png' to '../../testdata/train_tissue' (Total: 1).
Saved 'normal_002_tissue_compare.png' to '../../testdata/train_tissue' (Total: 2).
Saved 'normal_001_tissue_compare.png' to '../../testdata/train_tissue' (Total: 3).
Saved 'tumor_002_tissue_compare.png' to '../../testdata/train_tissue' (Total: 4).
Saved 'tumor_003_tissue_compare.png' to '../../testdata/train_tissue' (Total: 5).
Saved 'tumor_001_tissue_compare.png' to '../../testdata/train_tissue' (Total: 6).
Saved 'tumor_009_tissue_compare.png' to '../../testdata/train_tissue' (Total: 7).
```

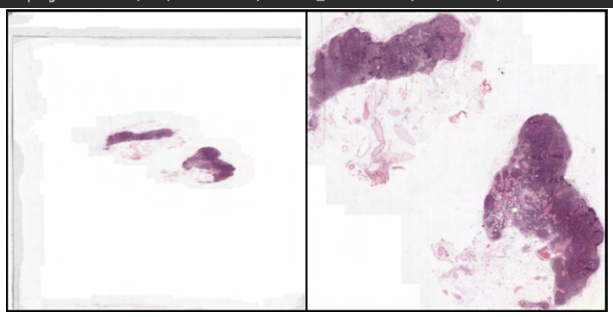


Figure 8.16: The following screenshot shows that the average time to convert a low-resolution `.PNG` image into its tissue detected version is output in a comprehensible format at the end of program execution.

```
Saved 'tumor_009_tissue_compare.png' to '../../testdata/train_tissue' (Total: 7).
Average Conversion Time (h:m:s.ms) = 0:00:00.411045
```

8.5.1.5 Generating Mean And Standard Deviation For An Image Directory

The following details testing performed on the implementation for calculating the per channel mean and standard deviation across a directory of images: `generate_images_mean_std.py` (see Table 8.13).

Table 8.13: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that program arguments must be given as expected so that mean and standard deviation calculation can occur.	Figure 8.17
Test that per channel mean and standard deviation calculation is correct for a single image.	Figure 8.18
Test that per channel mean and standard deviation normalises original low-resolution (black filtered) WSIs to have a mean of 0 and a standard deviation of 1.	Figure 8.19
Test that per channel mean and standard deviation normalises tissue detected low-resolution images to have a mean of 0 and a standard deviation of 1.	Figure 8.20

Figure 8.17: The following screenshot shows that a valid input directory (containing `.PNG` images) must be provided, as well as the dimensions to convert images to before calculating the mean and standard deviation. Note, target image dimensions used by the developed models affect the data set mean and standard deviation.

```
nmpoole@Nathans-MBP tools % python3 generate_images_mean_std.py
usage: generate_images_mean_std.py [-h] -i INPUT_DIR -width WIDTH -height HEIGHT
generate_images_mean_std.py: error: the following arguments are required: -i/--input_dir, -width, -height
nmpoole@Nathans-MBP tools % python3 generate_images_mean_std.py -i not_a_dir -width 0 -height 0
Error: Given input directory does not exist.
nmpoole@Nathans-MBP tools % python3 generate_images_mean_std.py -i ../../testdata/train/ -width 0 -height 0
Error: Given width/height must be a positive integer.
nmpoole@Nathans-MBP tools % python3 generate_images_mean_std.py -i ../../testdata/train/ -width 299 -height 0
Error: Given width/height must be a positive integer.
nmpoole@Nathans-MBP tools % python3 generate_images_mean_std.py -i ../../testdata/train/ -width 299 -height 299
Whole-Slide Classification - Computing Mean And Std:
```

Figure 8.18: The following screenshot shows that the implementation of mean and standard deviation calculation for a single input image (‘*normal_001.png*’) matches the outputs of the provided `mean()` and `std()` methods in `PyTorch`, showing correctness.

```
Whole-Slide Classification - Computing Mean And Std:
Input Directory Num Images: 1 (from '../../testdata/mean_std_test/').
------------------------------------------------------------------------
Calculating Mean and Std...
------------------------------------------------------------------------
Input Directory Image Stats (at 299 x 299 pixels):
Channel Mean = tensor([0.9766, 0.9725, 0.9752])
Channel Std = tensor([0.0620, 0.0829, 0.0674])
Torch.Mean() Channel Mean = tensor([0.9766, 0.9725, 0.9752])
Torch.Std() Channel Std = tensor([0.0620, 0.0829, 0.0674])
```

Figure 8.19: The following screenshot shows that the normalisation values computed for the low-resolution (black filtered) WSIs normalises all of such images to have an approximate mean of 0 and an approximate standard deviation of 1.

```
Mean and Std Of Normalised Images In '../data/Camelyon/L8_filt/train/':
Normalisation Using Mean = [0.9159, 0.8941, 0.9208], Std = [0.1193, 0.1636, 0.1091]
Mean = tensor([ 2.0784e-05,  1.3678e-04, -2.8655e-04])
Std = tensor([0.8166, 0.8385, 0.8396])
```

Figure 8.20: The following screenshot shows that the normalisation values computed for the low-resolution tissue detected images normalises all of such images to have an approximate mean of 0 and an approximate standard deviation of 1.

```
Mean and Std Of Normalised Images In '../data/Camelyon/L8_tissue/train/':
Normalisation Using Mean = [0.8572, 0.8084, 0.8592], Std = [0.1545, 0.2144, 0.1475]
Mean = tensor([-2.5691e-04,  1.5680e-04, -3.0974e-05])
Std = tensor([0.8788, 0.8952, 0.8755])
```

8.5.2 Whole-Slide Classification Testing

This section details high-level correctness testing performed on the whole-slide classification deep learning pipeline.

8.5.2.1 Whole-Slide Classification Data

The following details testing performed on the data set implementation for the whole-slide classification task: `classification_data.py` (see Table 8.15).

Table 8.15: Set of tests with their corresponding proof of correctness.

Test Description	**Correctness Proof**
Test that data sets can be created for each of the three modes: train, validate, and test.	Figure 8.21
Test that the data sets in each mode have the correct size.	Figure 8.21
Test that the data sets in each mode have the correct images.	Figure 8.21
Test that the correct transforms are selected when training using augment levels 0, 1, 2 and 3.	Figure 8.22
Test that the correct transforms are selected when validating using augment levels 0, 1, 2 and 3.	Figure 8.23
Test that the correct transforms are selected when testing using augment levels 0, 1, 2 and 3.	Figure 8.24
Test that the correct label of '0' is given for images of the 'normal' class, where file names may contain 'normal', 'none', or 'negative' to indicate such.	Figure 8.25
Test that the correct label of '1' is given for images of the 'tumor' class, where file names may contain 'tumor', 'itc', 'micro', or 'macro' to indicate such.	Figure 8.26
Test that the image, as well as corresponding label and file name, are returned when a given index of the data set is requested.	Figure 8.27

Figure 8.21: In the following screenshot, it is demonstrated that data sets can be created for the purposes of training, validating, and testing. The level 8 '*Camelyon*' data has been used as input. The training set has 618 images, the validation set has 152 images, and the testing set has 129 images, all of which are as expected. Also, example image names within the data sets have been sampled at random and shown; the correct images are being selected from the expected sub-directory within the given input directory.

```
Creating Training Data Set From '../data/Camelyon/L8/train'.
Train Data Set Size: 618.
Example Data: ['patient_000_node_2_negative.png', 'normal_100.png', 'patient_072_node_0_itc.png']...
Creating Validation Data Set From '../data/Camelyon/L8/eval'.
Validate Data Set Size: 152.
Example Data: ['normal_128.png', 'normal_114.png', 'patient_022_node_4_macro.png']...
Creating Test Data Set From '../data/Camelyon/L8/test'.
Test Data Set Size: 129.
Example Data: ['test_086_none.png', 'test_087_none.png', 'test_014_none.png']...
```

Figure 8.22: The following screenshots show the transformations selected for the training data set under the different augmentation levels. Only the preparation transformation (i.e., resizing and normalisation) is selected when no augmentation is specified (i.e., versions 0 and 2) (top). The data augmentation transform (i.e., random flipping, rotation, and colour jitter) and the data preparation transform is applied for versions 1 and 3, as expected (bottom). Note that versions 2 and 3 use a different mean and standard deviation, which has been calculated for the tissue detected images data set.

```
Training Augmentations For Augment Level 0 and 2:
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
```

```
Training Augmentations For Augment Level 1 and 3:
Compose(
    Compose(
    RandomVerticalFlip(p=0.5)
    RandomHorizontalFlip(p=0.5)
    Rotate90Intervals()
    ColorJitter(brightness=[0.75, 1.25], contrast=[0.75, 1.25], saturation=[0.75, 1.25], hue=[-0.1, 0.1])
)
    Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
)
Compose(
    Compose(
    RandomVerticalFlip(p=0.5)
    RandomHorizontalFlip(p=0.5)
    Rotate90Intervals()
    ColorJitter(brightness=[0.75, 1.25], contrast=[0.75, 1.25], saturation=[0.75, 1.25], hue=[-0.1, 0.1])
)
    Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
)
```

Figure 8.23: The following screenshot shows the transformations selected for the validation data set under the different augmentation levels. The validation set only uses data preparation (i.e., resizing and normalisation), as expected.

```
Validation Augmentations For Augment Level 0 and 1:
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
Validation Augmentations For Augment Level 2 and 3:
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
```

Figure 8.24: The following screenshot shows the transformations selected for the testing data set under the different augmentation levels. The testing set only uses data preparation (i.e., resizing and normalisation), as expected.

```
Test Augmentations For Augment Level 0 and 1:
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.6418, 0.62, 0.6467], std=[0.4077, 0.4085, 0.4082])
)
Test Augmentations For Augment Level 2 and 3:
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
Compose(
    Resize(size=(299, 299), interpolation=bilinear, max_size=None, antialias=None)
    ToTensor()
    Normalize(mean=[0.8572, 0.8084, 0.8592], std=[0.1545, 0.2144, 0.1475])
)
```

Figure 8.25: The following screenshot shows that the correct negative class label, '0', is given for files associated with the negative class, regardless of whether their file name contains 'normal', 'none', or 'negative'.

```
File Name: 'patient_000_node_2_negative.png', Label: '0'.
File Name: 'normal_100.png', Label: '0'.
File Name: 'test_086_none.png', Label: '0'.
```

Figure 8.26: The following screenshot shows that the correct positive class label, '1', is given for files associated with the positive class, regardless of whether their file name contains 'tumor', 'itc', 'micro', or 'macro'.

```
File Name: 'tumor_073.png', Label: '1'.
File Name: 'patient_072_node_0_itc.png', Label: '1'.
File Name: 'patient_081_node_2_micro.png', Label: '1'.
File Name: 'patient_013_node_2_macro.png', Label: '1'.
```

Figure 8.27: The following screenshot shows that the image, label, and file name is returned for requested indices from the data sets. The images are tensors with RGB channels (i.e., 3) and 299 x 299 pixel dimensions.

```
Requested Index = 128.
Image: 'torch.Size([3, 299, 299])', File Name: 'tumor_076.png', Label: '1'.
Requested Index = 256.
Image: 'torch.Size([3, 299, 299])', File Name: 'tumor_003.png', Label: '1'.
Requested Index = 512.
Image: 'torch.Size([3, 299, 299])', File Name: 'patient_009_node_1_micro.png', Label: '1'.
```

8.5.2.2 Whole-Slide Classification Training

The following details testing performed on the model training implementation for the whole-slide classification task: `classification_train.py` (see Table 8.17).

Table 8.17: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that valid program arguments (i.e., input directory path, output file path, augmentation level, and optional hyper-parameter optimisation) must be given to `classification_train.py`.	Figure 8.28
Test that the model, loss function, and optimiser are initialised correctly.	Figure 8.29
Test that the training and validation data sets are initialised correctly and have the correct sizes.	Figure 8.30
Test that training occurs correctly for all batches in the training data during an epoch.	Figure 8.31
Test that validation occurs correctly for all batches in the validation data during an epoch.	Figure 8.32
Test that training and validation loss is reported for each epoch.	Figure 8.33
Test that training and validation loss is reported to `TensorBoard` every epoch.	Figure 8.34
Test that the current model is saved when the validation loss decreases within an epoch.	Figure 8.35
Test that training is terminated when the early stopping condition is met.	Figure 8.36
Test that total training time is reported when training is terminated.	Figure 8.37
Test that automated hyper-parameter optimisation can be executed and runs correctly.	Figure 8.38

Figure 8.28: The following screenshot shows that valid program arguments must be provided to the `classification_train.py` script, otherwise, an error message is displayed indicating any issues.

```
nmpoole@Nathans-MBP src % python3 classification_train.py
usage: classification_train.py [-h] -i INPUT_DIR -o OUTPUT_FILENAME -a AUGMENT [-hpo]
classification_train.py: error: the following arguments are required: -i/--input_dir, -o/--output_filename, -a/--augment
nmpoole@Nathans-MBP src % python3 classification_train.py -i not_a_dir -o not_a_dir/output.pth -a -1
Error: Given input directory does not exist.
nmpoole@Nathans-MBP src % python3 classification_train.py -i ../data/Camelyon/L8/ -o not_a_dir/output.pth -a -1
Error: Invalid location of output file specified.
nmpoole@Nathans-MBP src % python3 classification_train.py -i ../data/Camelyon/L8/ -o ../data/output.pth -a -1
Error: Augmentation level must be 0, 1, 2, or 3.
```

Figure 8.29: The following screenshot shows the program state during execution, which was investigated to ensure the model, loss function, and optimiser were being instantiated with `PyTorch` as intended. As shown, the 'Inception v3' model is being used, along with the 'Adam' optimiser using binary cross entropy loss. The internal state of the model, loss function, and optimiser were checked to verify aspects such as output layer, positive class weight, etc. were being used as expected.

```
> criterion = {BCEWithLogitsLoss} BCEWithLogitsLoss()
> model = {Inception3} Inception3(\n  (Conv2d_1a_3x3): BasicConv2d(\n    (conv): Conv2d(3, 32, kernel_size=(3, 3), stride=(2, 2), bias=False)\n
> optimiser = {Adam} Adam (\nParameter Group 0\n    amsgrad: False\n    betas: (0.9, 0.999)\n    eps: 1e-08\n    lr: 0.001\n    weight_decay: 0\n)
```

Figure 8.30: The following screenshot shows that the training and validation data sets are being instantiated correctly during model training. The functionality of these data sets was checked for correctness when the `classification_data.py` script was tested. The training data set has 618 WSIs, whilst the validation set has 152 WSIs, which is shown to be the case.

```
Training Dataset Size: 618 (from '../data/Camelyon/L8/train').
Training Dataset Augmentation Level: 0
Evaluation Dataset Size: 152 from '../data/Camelyon/L8/eval').
```

Figure 8.31: The following screenshot shows that training occurs for all batches in the training data set; (19 batches x 32 WSIs per batch) + 10 WSIs in the last batch = 618 training WSIs.

```
Epoch 1 of 500:
    Training Batch 1 of 20: Batch Size = 32
    Training Batch 2 of 20: Batch Size = 32
    Training Batch 3 of 20: Batch Size = 32
    Training Batch 4 of 20: Batch Size = 32
    Training Batch 5 of 20: Batch Size = 32
    Training Batch 6 of 20: Batch Size = 32
    Training Batch 7 of 20: Batch Size = 32
    Training Batch 8 of 20: Batch Size = 32
    Training Batch 9 of 20: Batch Size = 32
    Training Batch 10 of 20: Batch Size = 32
    Training Batch 11 of 20: Batch Size = 32
    Training Batch 12 of 20: Batch Size = 32
    Training Batch 13 of 20: Batch Size = 32
    Training Batch 14 of 20: Batch Size = 32
    Training Batch 15 of 20: Batch Size = 32
    Training Batch 16 of 20: Batch Size = 32
    Training Batch 17 of 20: Batch Size = 32
    Training Batch 18 of 20: Batch Size = 32
    Training Batch 19 of 20: Batch Size = 32
    Training Batch 20 of 20: Batch Size = 10
```

Figure 8.32: The following screenshot shows that validation occurs for all batches in the validation data set; (4 batches x 32 WSIs per batch) + 24 WSIs in the last batch = 152 training WSIs.

```
    Evaluating Batch 1 of 5: Batch Size = 32
    Evaluating Batch 2 of 5: Batch Size = 32
    Evaluating Batch 3 of 5: Batch Size = 32
    Evaluating Batch 4 of 5: Batch Size = 32
    Evaluating Batch 5 of 5: Batch Size = 24
```

Figure 8.33: The following screenshot shows that training and validation loss is reported at the end of every epoch. The reported values were manually verified for several epochs to ensure correctness.

```
    Epoch Training Loss: 0.8855060855933378
    Epoch Evaluation Loss: 1.007372165981092
```

Figure 8.34: The following screenshots show that the training (left) and validation/evaluation (right) loss is reported to `TensorBoard` every epoch for visualisation purposes. Note that these graphs were generated by running model training on a small subset of data for testing the implementation and are therefore not representative of model performance - they are just for testing purposes.

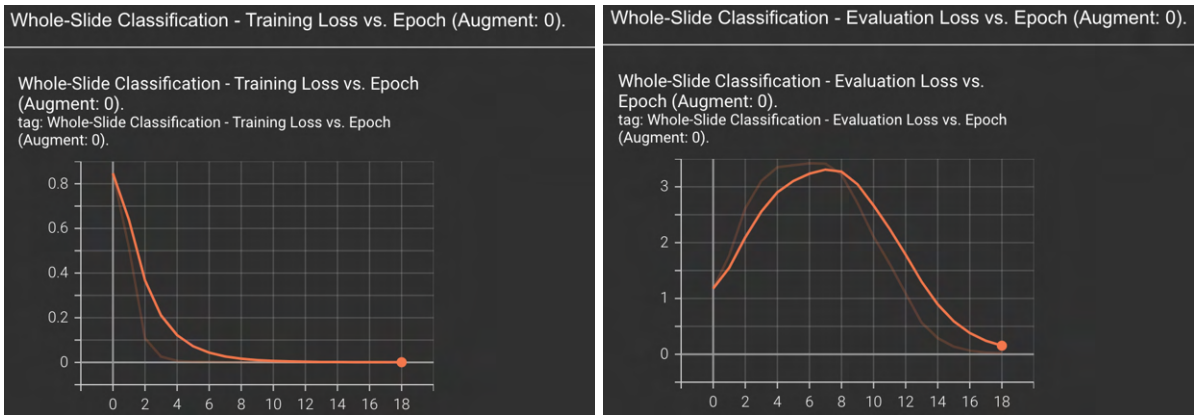


Figure 8.35: The following screenshot shows that the current model is saved when the validation loss in an epoch is lower than any previously reported validation loss. In the example shown, the model from the first epoch is saved, which is always the case as the first epoch will give the lowest validation loss so far; it is the reference for all following epochs.

```
Validation Loss Decreased (inf --> 1.007372).
Saved Trained Model (State Dict) To: ' ../testdata/testmodel.pth '.
```

Figure 8.36: The following screenshot shows that training is stopped early when the stopping condition is met. In the example, the stopping condition is set as 50 epochs without validation loss improvement, which is 10% of the maximum amount of epochs specified for training (500).

```
Stopping Training: Early stopping condition met - 50 epochs without criterion improvement.
```

Figure 8.37: The following screenshot shows that the training time is reported after training has completed.

```
Training Complete In 167m 12s.
```

Figure 8.38: The following screenshot shows that automated hyper-parameter optimisation via `RayTune` can be executed when the '*-hpo*' optional argument is specified. Here, 10 trials have been initialised with a randomly chosen configuration from the hyper-parameter search space.

```
Resources requested: 4.0/4 CPUs, 1.0/1 GPUs, 0.0/18.56 GiB heap, 0.0/9.28 GiB objects (0.0/1.0 accelerator_type:G)
Result logdir: /data/np57/CS5199/src/ray_results/classification_train_a0_1637974274.7409186
Number of trials: 10/10 (9 PENDING, 1 RUNNING)
+---------------------+----------+-----------------+--------------+-----------------+--------------+------------+
| Trial name          | status   | loc             |   batch_size |   learning_rate |   max_epochs |   momentum |
|---------------------+----------+-----------------+--------------+-----------------+--------------+------------|
| DEFAULT_20c35_00000 | RUNNING  | 172.17.0.2:270 |           16 |           0.001 |          100 |        0.9 |
| DEFAULT_20c35_00001 | PENDING  |                 |           16 |           0.1   |          300 |        0.9 |
| DEFAULT_20c35_00002 | PENDING  |                 |           16 |           0.0001 |          500 |        0.3 |
| DEFAULT_20c35_00003 | PENDING  |                 |            8 |           0.001 |          500 |        0.1 |
| DEFAULT_20c35_00004 | PENDING  |                 |            8 |           0.01  |          700 |        0.9 |
| DEFAULT_20c35_00005 | PENDING  |                 |            8 |           0.0001 |          100 |        0.7 |
| DEFAULT_20c35_00006 | PENDING  |                 |           32 |           0.01  |          100 |        0.3 |
| DEFAULT_20c35_00007 | PENDING  |                 |            8 |           0.01  |          700 |        0.3 |
| DEFAULT_20c35_00008 | PENDING  |                 |           32 |           0.001 |          300 |        0.7 |
| DEFAULT_20c35_00009 | PENDING  |                 |           32 |           0.001 |          300 |        0.9 |
+---------------------+----------+-----------------+--------------+-----------------+--------------+------------+
```

8.5.2.3 Whole-Slide Classification Inference

The following details testing performed on the model inference implementation for the whole-slide classification task: `classification_infer.py` (see Table 8.19).

Table 8.19: Set of tests with their corresponding proof of correctness.

Test Description	Correctness Proof
Test that valid program arguments (i.e., input image directory, input model path, output CSV file, and augment level) must be provided to `classification_infer.py`.	Figure 8.39
Test that the test data set is instantiated correctly from the provided input directory.	Figure 8.40
Test that the model is loaded correctly from the provided model path.	Figure 8.41
Test that inference occurs in batches for all test data as expected.	Figure 8.42
Test that the AUC score is calculated and output by the program.	Figure 8.43
Test that the average inference time is calculated and output by the program.	Figure 8.44
Test that predictions are wrote to the output file as required by the `Camelyon-16` challenge.	Figure 8.45

Figure 8.39: The following screenshot shows that valid program arguments must be provided to the `classification_infer.py` script, otherwise, an error message is displayed indicating any issues.

```
nmpoole@Nathans-MBP src % python3 classification_infer.py
usage: classification_infer.py [-h] -i INPUT_DIR -m MODEL_FILE -o OUTPUT_FILE -a AUGMENT [-v]
classification_infer.py: error: the following arguments are required: -i/--input_dir, -m/--model_file, -o/--output_file, -a/--augment
nmpoole@Nathans-MBP src % python3 classification_infer.py -i not_a_dir -m not_a_model -o not_a_dir/output.csv -a -1
Error: Given input directory does not exist.
nmpoole@Nathans-MBP src % python3 classification_infer.py -i ../data/Camelyon/L8/ -m not_a_model -o not_a_dir/output.csv -a -1
Error: Given model file does not exist.
nmpoole@Nathans-MBP src % python3 classification_infer.py -i ../data/Camelyon/L8/ -m ../testdata/testmodel.pth -o not_a_dir/output.csv -a -1
Error: Invalid location of output file specified.
nmpoole@Nathans-MBP src % python3 classification_infer.py -i ../data/Camelyon/L8/ -m ../testdata/testmodel.pth -o ../testdata/output.csv -a -1
Error: Augmentation level must be 0, 1, 2, or 3.
```

Figure 8.40: The following screenshot shows that the test data set is initialised as expected. In this test, the test data set has 3 WSIs, which is the expected size (the implementation testing directory, '`testdata/`', has a test set of 3 images).

```
Whole-Slide Classification - INFER:
-------------------------------------------------
Inference Device:  cpu
Test Dataset Size: 3 (from '../testdata/test').
```

Figure 8.41: The following screenshot shows that a model is loaded without issues from a specified model file path generated by the model training script. In this test, the model used is a simple one generated from the implementation testing directory training data in '`testdata/`'. This model was manually inspected during program execution to make sure the weights of the model were initialised correctly using those set by model training.

```
Inception v3 Model Created - State Dict Loaded From: '../testdata/testmodel.pth'.
```

Figure 8.42: The following screenshot shows that inference is executed in batches. However, in this test, the `testdata/` training set used for implementation testing only has 3 WSIs, so there is a single batch in the screenshot (batch size is 32), as expected.

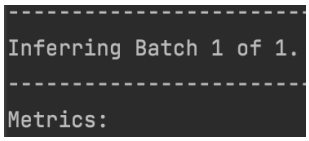


Figure 8.43: The following screenshot shows that the AUC score for a model is output after all test data has been inferred. In the screenshot, the `testdata/` test set has just three images (two tumour WSIs and one normal WSI as illustrated by the confusion matrix given), which are all correctly predicted by the model, hence the perfect AUC score.

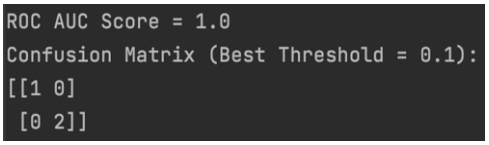


Figure 8.44: The following screenshot shows that the average model inference time per image is output by the script, which is required for calculation of the inference time metric used in the results analysis. In this example, the model takes roughly 200 milliseconds on average to infer each input image (which have no augmentation applied in this case).

Figure 8.45: The following screenshots show that the model predictions are output to the specified output file. The top screenshot shows that the program indicates where it saves the predictions (i.e., the output file). The bottom screenshot shows the contents of the output CSV file for this test, which has the format expected by the *Camelyon* challenges (i.e., file name then probability of positive class).