Master's thesis

NTNU
Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

Sølvi Knapstad

Automated Segmentation of Metastatic Lymph Nodes in Lymphoma Patients

Master's thesis in Applied Physics and Mathematics
Supervisor: Pål Erik Goa
Co-supervisor: Anum Masood
June 2022

NTNU
Norwegian University of
Science and Technology

Sølvi Knapstad

Automated Segmentation of Metastatic Lymph Nodes in Lymphoma Patients

Master's thesis in Applied Physics and Mathematics
Supervisor: Pål Erik Goa
Co-supervisor: Anum Masood
June 2022

Norwegian University of Science and Technology
Faculty of Natural Sciences
Department of Physics

NTNU
Norwegian University of
Science and Technology

Preface

This master thesis concludes my final work as a student in the five-year master's degree program of Applied Physics and Mathematics at the Norwegian University of Science and Technology. The master thesis is a continuation of the specialization project conducted in the autumn of 2021, whereas the presented work was completed during the spring of 2022. In the process of writing both my thesis and specialization project, there are several people that deserve acknowledgments for their support and assistance during the last year.

First and foremost, I am grateful to both my supervisors Pål Erik Goa and Anum Masood. I want to thank them for their support, guidance, and help during the last year. Their aid and advice with both the work and writing of my thesis has been profoundly appreciated. Pål Erik has let me be a part of his research group which has provided me with a broader and new perspective of the field of biophysics and medical technology. This has been a great motivation and I have gained helpful advice from the group discussions. Anum, thank you for quickly answering my questions and assisting me when errors appeared. I would also like to thank Dr. Håkon Johansen for his guidance, training, and validations regarding the manual segmentations of the lymphoma patients. Mattijs Elschott has provided valuable feedback and encouragement through the last year which I am appreciative of. A special thank you to Live Eikenes for providing access to the lymphoma dataset and the information regarding the image acquisition performed during the lymphoma study.

I will also like to thank my friends and fellow students whom have contributed to making my years as a student so memorable. I am especially grateful for my collective whom I have spent the last years living through a pandemic with and who has filled my spare time with joy and laughter. At last, I would like to thank my family who has always believed, supported, and encouraged me to reach my goals.

Sølvi Knapstad
Trondheim, June 10, 2022

Abstract

Using the deep learning artificial neural network 2D U-Net, this project tests the accuracy of the 2D U-Net for the purpose of automatically segmenting malignant lesions in PET/MR images of patients with metastatic lymphoma.

For Hodgkin and Non-Hodgkin lymphoma, the FDG PET/MRI segmentations are important for prognosis, staging, and response assessment of lymphoma patients. However, manually segmentations are time-consuming and difficult in complex patient cases and for high disease burden. The aim of this project is to develop an automated method for segmentation of cancer-affected lymph-nodes in PET/MRI using a deep neural network. FDG PET/MRI baseline, interim, and End-Of-Treatment (EOT) images of Hodgkin and Non-Hodgkin lymphoma patients were analyzed. Two groups of radiologist and nuclear medicine physicians have contributed with clinical reading of the PET/MR images following standardized protocols. However, the segmentation ground truth was missing from the lymphoma dataset, and it was crucial for implementing the deep learning network for an automated segmentation process. The manual segmentation required has therefore been performed by the author and validated by a nuclear medicine physician from St. Olavs Hospital.

The neural network model was taught how to perform classification tasks directly from images, i.e., the network was trained to recognize patterns from a dataset consisting of 64 PET/MRI examinations. A 3-channel multi-modal image, i.e., an RGB image, consisting of a PET, a T2-HASTE, and a DWI with b = 800 s/mm^2 was used as input for the algorithm. The model was trained to replicate the segmentations of the ground truth by using a 2D U-Net architecture.

Furthermore, the lymphoma dataset was divided in a 85/15 ratio for training and testing consisting of 53 and 11 PET/MRI examinations, respectively. Both a 4-fold and 13-fold cross-validation were performed for the training of the model. The validation resulted in average dice scores of 0.61 and 0.63 respectively for the 4-fold and 13-fold trained models. Several other metrics such as loss, accuracy, precision, recall, Negative Predictive Value (NPV), and specificity were included for the voxel level analysis. The scores were 0.011, 0.97, 0.83, 0.11, 0.97, 0.99, respectively for the 4-fold validation and 0.065, 0.97, 0.90, 0.10, 0.97, 0.99, respectively for the 13-fold validation. The average dice score of the testing patient were 0.29 and 0.32 respectively for the 4-fold and 13-fold cross-validation which suggested an inferior performance on unseen patients compared to the PET/MRI

examinations used in the validation. Despite the overall high scores for the evaluation metrics, the voxel based analysis did not give a great indication of how well the model managed to segment cancer lesions due to the majority of the voxels in a patient being classified as true negative. Therefore, a lesion-based analysis were conducted and it revealed that the model often segmented fewer lesions than in the ground truth. This indicated that the model's main limitation was the number of false negative predicted lesions. As a consequence, the model performs better on the validation data than for the testing dataset which was excluded from the training.

In conclusion, the trained 2D U-Net model automatically segments malignant lymph node lesions in the 3-channel multi-modal images. However, future research should focus on improving the dice score, co-registration, decrease the number of undetected tumor lesions, and increase the dataset to ensure a larger variation in the cohort. This will benefit the training and yield better results.

Sammendrag

Ved å bruke det kunstige nevrale nettverket 2D U-Net, tester denne masteroppgaven nøyaktigheten til 2D U-Net med formålet om å automatisk segmentere ondartede svultser i PET/MR-bilder av pasienter med metastatisk lymfom.

For Hodgkin- og Non-Hodgkin-lymfom er FDG PET/MR-segmenteringene viktige for prognose, stadieinndeling (staging) og responsvurdering av lymfompasienter. Manuelle segmenteringer er imidlertid tidkrevende og vanskelige i komplekse pasienttilfeller der en har høy sykdomsbyrde. Målet med dette prosjektet er å utvikle en automatisert metode for segmentering av kreftrammede lymfeknuter i PET/MR ved bruk av dyp læring (deep learning), nærmere bestemt et dypt kunstig nevralt nettverk. FDG PET/MR-baseline, interim- og behandlingsavslutningsbilder (EOT) av Hodgkin- og Non-Hodgkin-lymfompasienter ble analysert. To grupper radiologer og nukleærmedisinere har bidratt med klinisk lesing av PET/MR-bildene etter standardiserte protokoller. Imidlertid manglet de faktiske segmenteringene, m.a.o segmenterings fasiten, fra lymfomdatasettet, og disse var avgjørende for å implementere dyplæringsnettverket for en automatisert segmenteringsprosess. De manuelle segmenteringene som krevdes ble utført av forfatteren og validert av en nukleærmedisiner fra St.Olavs Hospital.

Den nevrale nettverksmodellen ble lært hvordan man utfører klassifiseringsoppgaver direkte fra bilder, dvs. nettverket ble opplært til å gjenkjenne mønstre fra et datasett bestående av 64 PET/MR-undersøkelser. Et 3-kanals multimodalt bilde, et RGB-bilde, bestående av PET, T2-HASTE og DWI med b $= 800$ s/mm^2 ble brukt som input for algoritmen, og modellen ble lært til å gjenskape segmenteringene i grunnsannheten (segmenterings fasiten) ved å bruke en 2D U-Net-arkitektur.

Videre ble lymfomdatasettet delt inn i et 85/15-forhold for trening og testing som bestod av henholdsvis 53 og 11 PET/MR-undersøkelser. Både en 4-fold og 13-fold kryssvalidering ble utført for opplæringen av modellen. Valideringen resulterte i gjennomsnittlige Dice-score (overlappingsmål) på henholdsvis $0,61$ og $0,63$ for 4-fold og 13-fold modellene. Flere andre evaluaringer som tap, nøyaktighet, presisjon, tilbakekalling, negativ prediktiv verdi (NPV) og spesifisitet ble inkludert for voxel-nivåanalysen. Resultatene var $0,011$, $0,97$, $0,83$, $0,11$, $0,97$, $0,99$, henholdsvis for 4-fold valideringen og $0,065$, $0,97$, $0,90$, 0.10, $0,97$, $0,99$, henholdsvis for 13-fold valideringen. Den gjennomsnittlige dice scoren til testpasientene var henholdsvis $0,29$ og $0,32$ for 4-fold og 13-fold kryssvalidering, noe

som antyder en dårligere ytelse på nye og usette pasienter sammenlignet med PET/MR-undersøkelsene brukt i valideringen. Til tross for de generelt høye verdiene for evalueringsmetodene, ga den voxelbaserte analysen ingen god indikasjon på hvor nøyaktig modellen klarte å segmentere kreftlesjoner ettersom flertallet av vokslene i pasientene ble klassifisert som ekte negative (TN). Derfor ble det utført en lesjonsbasert analyse, og den avslørte at modellen ofte segmenterte færre lesjoner enn det som var tilstede i grunnsannheten. Dette indikerte at modellens hovedbegrensning var antallet falske negative predikerte kreftsvultser. Som en konsekvens, presterer modellen bedre på valideringsdataene enn for testdatasettet som ble ekskludert fra opplæringen.

For å konkludere, så segmenterer den trente 2D U-Net-modellen automatisk ondartede lymfeknutesvultser i 3-kanals multimodale bilder. Fremtidig arbeid bør fokusere på å forbedre overlappingsmålet dice score, co-registreringen, redusere antall uoppdagede tumorlesjoner, samt øke datasettet for å sikre en større variasjon i kohorten. Dette kan forbedre både treningen og resultatene.

Acronyms

AI Artificial Intelligence.

ANN Artificial Neural Network.

cHL Classic Hodgkin Lymphoma.

CMR Complete Metabolic Response.

CNN Convolutional Neural Network.

CT Computed Tomography.

DICOM Digital Imaging and Communications in Medicine.

DL Deep Learning.

DLBCL Diffuse Large B-Cell Lymphoma.

DS Dice Score.

DWI Diffusion Weighted Imaging.

EOT End of Treatment.

FDG Fluorodeoxyglucose.

FDR False Discovery Rate.

FID Free Induction Decay.

FN False Negative.

FNR False Negative Rate.

FP False Positive.

GPU Graphics Processing Unit.

HASTE Half Fourier Acquired Single-shot Turbo spin Echo.

LOOCV Leave-One-Out Cross-Validation.

LOR Line Of Response.

LYRIC Lymphoma Response to Immunomodulatory Therapy Criteria.

MRI Magnetic Resonance Imaging.

NHL Non-Hodgkin Lymphoma.

NiFTI Neuroimaging Informatics Technology Initiative.

NMR Nuclear Magnetic Resonance.

NMR No Metabolic Response.

NPV Negative Predictive Value.

OSEM Ordered Subset Expectation Maximization.

PET Positron Emission Tomography.

PMD Progressive Metabolic Disease.

PMR Partial Metabolic Response.

PPV Positive Predictive Value.

ReLU Rectified Linear Unit.

RF Radio Frequency.

RGB Red Green Blue color model.

ROI Region of Interest.

RS Reed-Sternberg-cells.

SPECT Single-Photon Emission Computerized Tomography.

SUV Standardized Uptake Value.

TE Echo Time.

TIRM Turbo Inversion Recovery Magnitude.

TN True Negative.

TOF Time Of Flight.

TP True Positive.

TPR True Positive Rate.

Contents

Chapter 1

Introduction

This chapter will introduce the thesis written this spring which is an continuation of the specialization project completed in Autumn 2021. The sections in this chapter will present the motivation behind the thesis and give an overview of the description, goals and contributions. The last section will provide an outline for the structure used in this thesis.

1.1 Motivation

Today, cancer is the leading cause of death worldwide and accounts for nearly one in six deaths [1]. One of the most common cancers in adolescents is called lymphoma which is a cancer that attacks the lymphatic system [2]. Lymphoma can easily metastasize to other organs and regions because it is present in the lymph system throughout the body and spreads outside the affected lymph nodes. Fortunately, several types of lymphomas are potentially curable when the patient is diagnosed and treated correctly.

Subsequently, in order to determine the best response assessment for a lymphoma patient, a correct staging of the disease is crucial for the outcome. In 2014, PET/CT was introduced as the standard imaging modality due to lymphoma tumors being FDG-avid [3]. However, studies show that PET/MRI has proven just as effective as PET/CT in the disease staging as it additionally provides improved soft-tissue contrast [4]. To further determine the best response assessment for a patient, the physicians need to detect and delineate the cancerous lymph nodes in the PET/CT and PET/MRI. This is often performed in teams of two, where two sets of nuclear physicians and oncologists annotate the cancer lesions for the same patient, which is both a time consuming process and tedious work to perform.

As manual segmentations of cancerous lymph nodes are resource-intensive, introducing an automated segmentation method can save both the physician and oncologist from hours of work. An AI model can do the segmentation efficient and accurately. It can therefore be an alternative for one of the two teams of nuclear physicians and oncologists. However, even the best AI cannot replace the physicians. The AI model is only meant to be a second reader, i.e., a tool to assist.

The segmentations performed by the AI needs to be validated because it does not always manage to differentiate between physiological and pathological uptake as it only sees pixel values.

Currently, the physicians at St. Olavs Hospital do not have access to programs, algorithms, or models that automatically segment lymphoma cancer nodes in multi-modal images. The work performed in this thesis can therefore open doors and introduce physicians to the benefits of deep learning techniques and automated segmentation methods.

1.2 Project Description and Goals

The overall goal of this thesis is to develop an automated method for segmentation of cancerous lymph nodes in lymphoma patients in multi-modal PET/MR images using the deep neural network 2D U-Net.

This thesis is a deep learning (DL) project which aims to provide an automated segmentation method for lymphoma patients. This will be achieved by implementing the neural network 2D U-Net and thereafter the model will be trained by using PET/MR data of metastatic lymphoma cancer patient. Unfortunately, the ground truth was missing from the data set, and therefore one of the project goals was to generate the ground truth. The manual segmentations were validated by Håkon Johansen, a nuclear medicine physician at St. Olavs Hospital.

The manually segmented patient data were used in the training, validation, and testing of the CNN model. The deep convolutional 2D U-Net design used in this project was developed and tested by a previous master student, Eivind Lysheim [5]. The network code was extended to handle multi-modal data-sets with three channels (PET, T2-HASTE, DWI with $b = 800\ s/mm^2$). This corresponds in moving from a gray-scale image to a colored one, in other words, an RGB image needs to be created where each color represents one channel. In order to create a larger dataset for the training, data augmentation was performed on the multi-modal images due to limited number of patients. Thereafter, the 2D U-Net was used to train the model and evaluation was done on both the validation and testing datasets.

Artificial neural networks have proven to be a great asset in imaging analysis as it provides great precision when it comes to detecting cancer. For this reason, the 2D U-Net model is expected to provide accurate segmentations of malignant lesions and clearly mark them in the multi-modal images. Future research will focus on implementing and using Artificial Intelligence (AI) and Deep Learning (DL) algorithms to better characterizes malignant lesions based on the annotation provided by the physician and author. The algorithm may be used for different imaging modalities and be modified to segment other malignant cancers where an automated segmentation process will be beneficial.

1.3 Contributions

The work performed in this thesis has contributed to the following research groups and institutions: the MR Physics group at NTNU, the Nuclear Medicine department at St. Olavs Hospital and the cancer group 180°N WP5: Machine Learning [6]. After completing this thesis, the MR Physics group will have access to 64 segmented and validated lymphoma patients. Additionally, they will have a Deep Learning model using the 2D U-Net network which can segment cancerous lymph nodes in a three channel multi-modal image. Moreover, the network can also benefit the 180°N WP5 group which can use this automatic segmentation method together with other projects for a fully automatic detection and staging of lymphoma patients. What is more is that the method can be utilized and tested for other types of cancers.

1.4 Report Structure

The thesis is divided into six chapters in addition to the preface, abstract, acronyms, and appendixes. Chapter 2 introduces the relevant theory and basics behind Lymphoma disease, PET, MRI, and Deep learning. Chapter 3 addresses the material and methods implemented. Whereas Chapter 4 presents the results from the training and testing of the model which were obtained this year. Chapter 5 grants a discussion of the results presented, and thereafter Chapter 6 provides conclusions based on the findings and results presented in the aforementioned sections. At last, the appendixes present additional information that may benefit the reader about the method, results, and the code implemented in both MATLAB and Python.

Chapter 2

Theory

This chapter contains four different sections that are important for the understanding of the thesis. The reader will be introduced to Lymphoma cancer as a disease, and thereafter PET and MR imaging will be explained. Lastly, deep learning and the neural network 2D U-Net will be presented. The subsequent sections are based on information presented in the specialization project conducted in the autumn of 2021: section 2.1 about Lymphoma Disease, section 2.2 about PET excluding subsection 2.2.3 and 2.2.4, section 2.3 about MRI physics excluding subsection 2.3.5 about hybrid PET/MRI, and lastly section 2.4 regarding deep learning excluding the subsections concerning neural network training and evaluating a neural network.

2.1 Lymphoma Disease

Globally, over six hundred thousand new cases of lymphoma cancer incidences were registered in 2020 [7]. In fact, lymphoma is one of the most common types of cancers to attack the body's circulatory and lymph system [8]. These tumor cells originate from lymphocytes, the immune system's own infection-fighting cells [9]. The lymphocytes are for instance located in the lymph nodes, spleen, thymus, and bone marrow [9], and thus, these regions are common for the cancer to metastasize to. Figure 2.1 provides an illustration of the different lymph node regions in the human body.

Lymphoma is divided into two broad groups: Classical Hodgkin Lymphoma (cHL) and Non-Hodgkin Lymphoma (NHL). Hodgkin lymphoma is characterized by Reed-Sternberg-cells (RS) or large Hodgkin cells in an inflammatory environment [8]. These cells appears to be resisting signals provided from the immune system to force the cancer cells to undergo apoptosis. NHL does neither consist of RS nor large Hodgkin cells. Therefore, a discovery of these cells are essential in order to give the patient a correct diagnosis [8].

The World Health Organization (WHO) International Classification of Disease reports of more than 50 different sub-types of lymphoma based on histopathologic, immunohistochemical, cytogenetic, and molecular analyses [10]. However,

in clinical practice, only a few subgroups of lymphoma account for the majority of the cancer incidences reported [10]. There is an overall high survival rate for patients diagnosed with Hodgkin Lymphoma, which is the most common category of cancer diagnosed in adolescents (age 15-19 years) [2]. cHL is a highly curable malignancy where the five year survival rate for patients diagnosed with cHL at age 0-19 years is 96.4% and as high as 89.8% for patients diagnosed between ages 20 and 64 years [2]. Non-Hodgkin Lymphoma on the other hand, is a quite aggressive group of lymphomas where Diffuse Large B-Cell Lymphoma (DLBCL) is the most common and consist of 33% of the diagnosed cases. The occurrence of DLBCL can appear in both childhood and adolescence, although it is most frequently diagnosed in patient that are over the age of 60 [10].

Today, several sub-types of lymphomas are curable when treated and diagnosed correctly. The treatment plan consists of either chemotherapy in combination with radiation therapy, immunochemotherapy, or chemotherapy alone [10]. The goal of the combined modality treatments have been to preserve efficacy of treatment while reducing the toxicity exposure to patients [2]. Since Hodgkin and Non-Hodgkin lymphoma involves different lymphocyte cells, it follows as a consequence that every type of lymphoma grows at a different rate and responds differently to treatment [9]. It is therefore important to diagnose the patient correctly in order to provide the best treatment plan with the use of the different treatment regimens mentioned above.

2.1.1 Staging and Response Assessment

The staging of cancer defines the location and extent of disease. In addition, it suggests prognostic information that can be used as a baseline against which the disease progression or the response to treatment should be compared to [11]. The most important factors influencing therapeutic decisions and prognosis are the histologic subtypes and the extent of the lymphoma disease. The majority of cases with cHL and DLBCL are treated with immunochemotherapy, and some with radiation in addition. For the diagnosis of lymphoma, PET/CT is today the standard for staging and assessment due to the fact that both cHL and some NHL subgroups are fluorodeoxyglucose (FDG)- avid. It is therefore recommended for initial staging and End-Of-Treatment (EOT) evaluation in patients with cHL and NHL subtypes like DLBCL [11]. The use of interim PET/CT as a predictive tool to identify early non-responders has been validated in advanced cHL, whereas for NHL it is not recommended [4]. The use of FDG in PET imaging provides an indication of the metabolic and proliferative activity within the tumor, and has therefore been an important contribution to the evolution of staging and response assessment of lymphoma [12].

In addition to standard morphological images, advanced MRI sequences like diffusion weighted imaging (DWI) provides functional information. PET/MRI produce improved soft-tissue contrast, the acquisition of truly simultaneous multiparametric images that yield morphological, molecular, and functional informa-

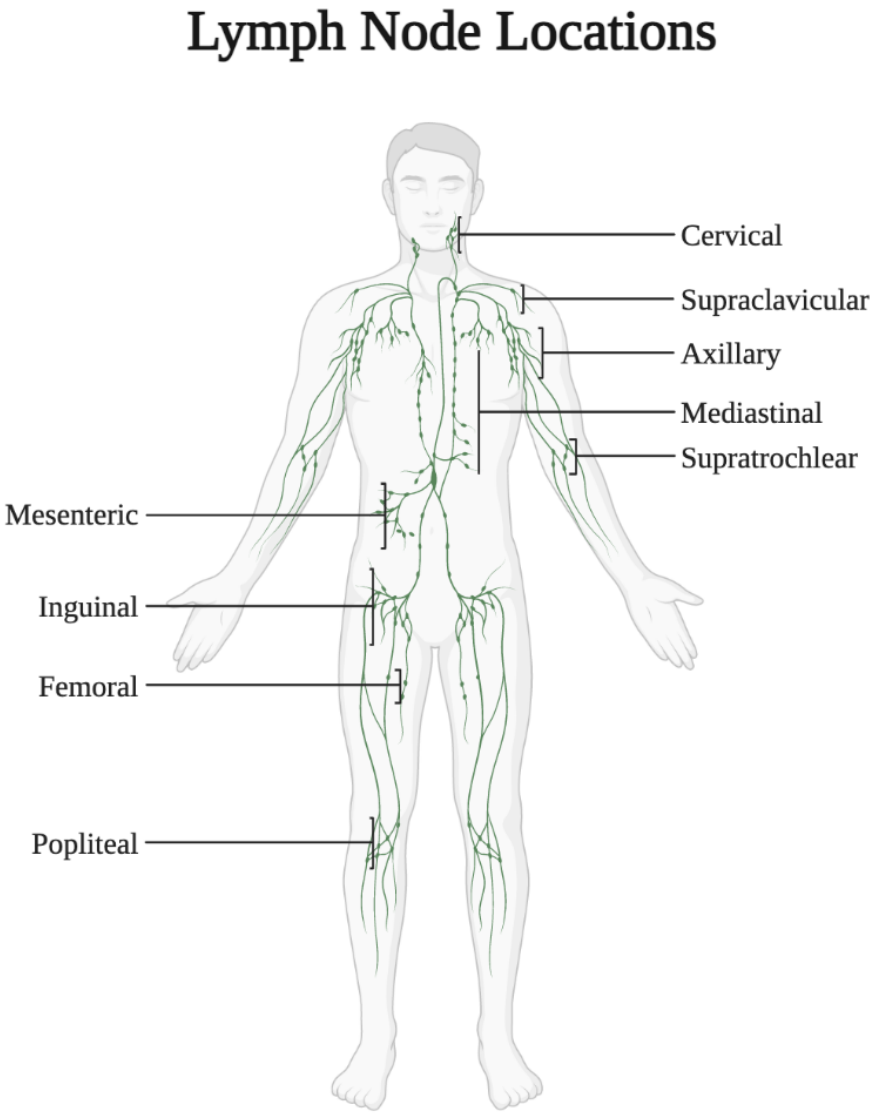


Figure 2.1: The figure depicts the locations of lymph nodes in the body with the corresponding name of the region where they can be found. Lymph node regions are used to determine the staging of lymphoma.

tion. It has been shown that PET/MRI can be as effective as PET/CT for both the disease staging and response assessment of NHL [4]. The benefits of using FDG PET/MRI as an alternative to PET/CT include the absence of ionising radiation and might therefore be advantageous [4], other advantages of PET/MRI, in addition to comparisons to PET/CT, can be found in Table 2.2

The clinical introduction of PET/CT imaging as a standard contributed significantly to the advancements of staging and response assessment systems in lymphoma such as the Ann Arbor staging system and the Deauville five point scale. The Ann Arbor system and the Deauville score were later modified once again after the Lugano Classifications were published in 2014, where the goal was to simplify and standardize the response assessment worldwide and address the new role of FDG PET/CT imaging for both staging and interim treatment response assessment [10].

2.1.2 Ann Arbor Staging System

For years, the Ann Arbor staging system was the most used classification system for cHL and NHL. The system itself is named after the town of Ann Arbor where the Committee on Hodgkin's Disease Staging Classification met in 1971 to agree upon it [13].

The Ann Arbor classification contains four different stages describing the basis of lymph node involvement. Stage 1 represent the involvement of a single lymph node region, or the involvement of a single extralymphatic organ or site. Stage 2 describes further progression of the disease with involvement of two or more lymph node regions on the same side of the diaphragm. However, this stage can also be a localized involvement of an extralymphatic organ or site. Moreover, stage 3 involves lymph node structures or regions on both sides of the diaphragm. And at last, stage 4 include diffuse or disseminated involvement of one or more extralymphatic organs. This stage can have isolated extralymphatic organ involvement without adjacent regional lymph node involvement, but with disease in distant sites involvement of the liver, bone marrow, pleura, or cerebrospinal fluid [13]. The Figure 2.2 provides a schematic overview of the original Ann Arbor staging system as described above.

There are additionally five substaging variables that can be included in the original Ann Arbor staging system in order to further classify and provide the correct staging for the specific lymphoma:

- A: The patient is asymptomatic
- B: The patient has presence of B symptoms (this includes fever, night sweats and/or weight loss of more than 10% of body weight over 6 months)
- E: There is involvement of a single, extranodal site, contiguous or proximal to a known nodal site (stages 1 to 3 only; additional extranodal involvement is stage 4)
- S: The patient has splenic involvement
- X: There is presence of bulky nodal disease: nodal mass larger than 1/3 of

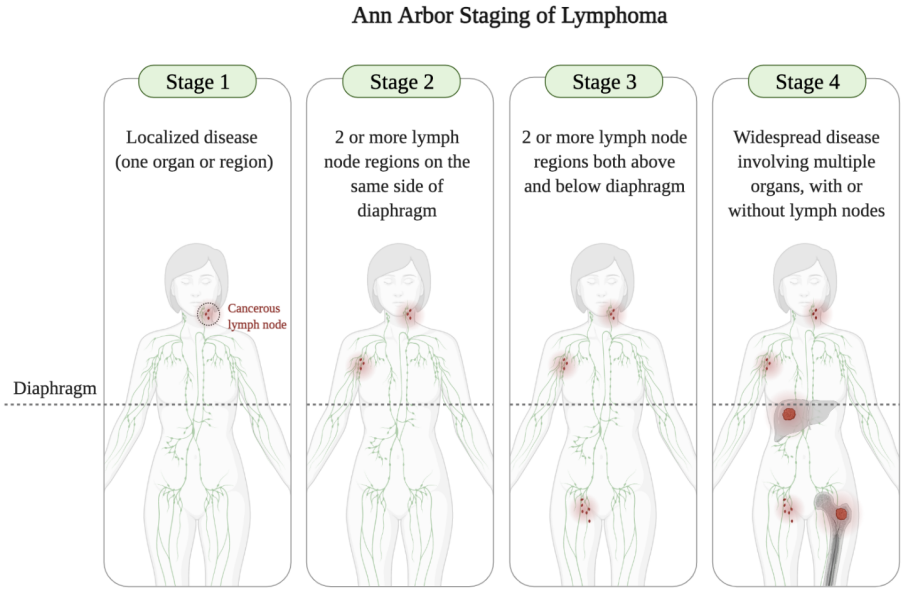


Figure 2.2: The figure shows a schematic table of the Ann Arbor staging system. A brief description of the areas of involvement for each stage is included.

intrathoracic diameter or 10 cm in dimension [13].

The Ann Arbor staging system has since been updated and thereafter modified after the Lugano Conference in 2014, and the new and improved Ann Arbor classification is now currently being used together with the Lymphoma Response to Immunomodulatory Therapy Criteria (LYRIC) classification [13].

2.1.3 Deauville Score

In addition to the Ann Arbor staging system, the Deauville five point scale (5-PS) was initially introduced for assessment on interim FDG PET/CT images due to the fact that the imaging modality is based on metabolic activity indicated by the FDG uptake in tumor cells. It is an internationally-recommended scale for routine clinical reporting and clinical trials using FDG PET-CT both in the initial staging and assessment of treatment response in cHL and for certain types of NHL [14].

The five point scale was established as the preferred reporting method in Deauville, France at the First International Workshop on PET in Lymphoma and has thereafter been included and used in several international lymphoma trials. The 5-PS Deauville score was intended as a simple scoring method with the flexibility to change the threshold between good and poor response according to the treatment strategy. The Deauville five point system has been validated for use in both interim and End-Of-Treatment imaging [4].

The Deauville point system scores the most intense uptake in a site of initial disease and includes the following categories [4]:

1. No FDG uptake
2. FDG uptake $\leq$ mediastinum
3. FDG uptake $>$ mediastinum but $\leq$ liver
4. FDG uptake moderately higher than liver
5. FDG uptake markedly higher than liver and/or new lesions

X. New areas of uptake unlikely to be related to lymphoma

Furthermore, as the 5-PS is now applied to both interim and EOT FDG PET/CT response assessments, four additional categories of response has been proposed as an outline for the metabolically activity of the tumors. The new categories has been defines as: (a) complete metabolic response of tumor, which is equivalent to a score of 1, 2, and 3 on the 5-PS. (b) partial metabolic response in tumor which is equal to a score of 4 or 5 with reduced FDG uptake. (c) no metabolic response which corresponds to a score of 4 or 5 without significant change in FDG uptake. Lastly, (d) progressive metabolic disease which is equivalent of a score of 4 or 5 with increased FDG uptake and/or findings of new lesions [10].

In order to achieve a correct interpretation of the 5-point Deauville scale a patient that scores a 1 or 2 on the scale will be interpreted as negative of lymphoma. In general, a patient with the score of 3 on the Deauville point scale will most likely represent complete metabolic response at interim imaging. This will often result in a good prognosis, and is for this reason also considered to be negative of lymphoma [10]. On the other hand, a patient with the score of 4 or 5 will be considered positive for lymphoma.

2.1.4 The Lugano Classification

The Lugano staging classifications is today the standard lymphoma staging system that is commonly used in clinical practice for both Classic Hodgkin Lymphoma and Non-Hodgkin Lymphoma [3]. The Lugano classifications serve as recommendations for initial evaluation, staging and response assessment of lymphoma. In 2014, the Lugano classification modernized staging of lymphomas by formally incorporating FDG PET/CT into standard staging processes [2]. A modification of the Ann Arbor staging system was introduced in order to most accurately diagnose the patients for the purpose of giving the best treatment. The Lugano classification recommends a modification of the Ann Arbor system's anatomical description of the lymphoma disease extent. The lymphoma patients are now recommended to be categorized as having *limited* or *advanced* disease. The limited disease was previously classified as stage 1 or 2 and the advanced disease was classified as stage 3 or 4 in the Ann Arbor system [10]. The Table 2.1 below shows the revised staging system for primary nodal lymphomas, and the renewed definitions describing the staging [11].

Furthermore, the classification recommends employing the Deauville 5 point system for both interim and End-Of-Treatment (EOT), response assessment with FDG PET/CT scans. Interim PET/CT imaging has become a valuable tool for the prognostication of the curable subtypes of lymphoma such as cHL and Diffuse

Table 2.1: Revised staging system for response assessment in lymphoma based on FDG-avid tumors for limited and advanced stages, response recommendations [11].

Stage	Involvement	Extranodal (E) Status
Limited		
1	One node or a group of adjacent nodes	Single extranodal lesions without nodal involvement
2	Two or more nodal groups (same side of diaphragm)	Stage 1 or 2 by nodal extent with limited contiguous extranodal involvement
2 bulky	2 as above, but with bulky disease	Not applicable
Advanced		
3	Nodes on both sides of the diaphragm or nodes above the diaphragm with spleen involvement	Not applicable
4	Additional non-contiguous extralymphatic involvement	Not applicable

Large B-Cell Lymphoma. A decrease in metabolic activity in the tumors, as seen by interpreting the patient's interim scan, indicates that the response is associated with improved outcome for the patient involved [12]. Although, EOT was introduced in order to establish a remission status for lymphoma patients, the Lugano classification do not recommend to change the undergoing patient treatment based solely on the PET/CT interim scan unless there is clear evidence of disease progression. The classification provides an interpretation of the FDG PET/CT evaluation [15]:

Score 1 and 2: The score is considered to represent Complete Metabolic Response (CMR) at interim and End-Of-Treatment scans.

Score 3: The score is dependent on the timing of the assessment, the clinical context and the treatment itself. The FDG uptake in the tumors declines during therapy in chemo-sensitive disease and residual FDG uptake higher than normal liver uptake is commonly encountered at interim scans in patients who achieve CMR at EOT scans.

Score 4 and 5 (interim): The scores suggests that chemo-sensitive disease provided uptake has reduced from the baseline scan and is considered to represent partial metabolic response in the patient.

Score 4 and 5 (EOT): The scores epitomizes residual metabolic disease even if the uptake has reduced from the baseline scan.

Moreover, the classification proposes suggestions for the timings of FDG PET/CT scans for EOT. It recommends to wait as long as possible after the last chemotherapy administration before doing an interim scan on a patient, and comments that it is ideally to wait $6-8$ weeks post chemotherapy for EOT scans but that a minimum of 3 weeks will also subdue. However, if the patient has undergone radiotherapy it is recommended to wait more than 3 months before performing the FDG PET/CT scanning [15].

Lastly, the Lugano classification provides a recommendation for response according to the Deauville five point scoring system:

Score 1 and 2: A low score will represent Complete Metabolic Response (CMR)

Score 3: This score also portrays CMR with standard treatment. However, in response-adapted trials, exploring deescalation, a score of 3 may be deemed as an inadequate response to avoid under-treatment of the patient. Therefore, the interpretation of score 3 depends on factors such as timing of assessment, treatment modality, and clinical context.

Score 4 and 5: At these scores, a patient with reduced uptake from baseline has Partial
Metabolic Response (PMR). However, at interim, this score indicates responding disease and for EOT scans, these scores suggests residual disease. On the other hand, if at these scores there are no indication of and/or no change in uptake from the baseline scan it will mean there is No Metabolic Response (NMR). If there is an increase in uptake compared to the baseline scan and/or new lesions detected, these scores are considered as Progressive

Metabolic Disease (PMD). In other words, at both interim and EOT scans, NMR and PMD indicates treatment failure [15].

The Lugano classifications were developed by clinical experts in lymphoma and the classification itself are meant to serve as unified guidelines for all physicians involved in diagnosis, management, and disease treating of lymphoma. The criteria developed at the Lugano conference represents consensus statements from several experts involved with lymphoma disease [10], which constitutes a renewed opportunity for nuclear medicine specialists and radiologists to clinically determine the best treatment regimen for their patients based on imaging findings.

2.2 Positron Emission Tomography

The nuclear medicine procedure Positron Emission Tomography (PET) measures the metabolic activity in different cells and tissues. PET is most commonly used for cancer detection, however, since it is an combination of nuclear medicine and biochemical analysis it is possible to visualize the biochemical changes happening in the body. The prominent reason FDG-PET differs form other nuclear medicine assessments is due to the fact that it detects the metabolic activity within different body tissues, whereas other types of nuclear medicine examinations, e.g. SPECT, discover where in the body the majority of the radioactive substance is collected.

Considering that PET is a nuclear medicine procedure, it acquires an injection of a radiopharmaceutical to assist in the examination of the patient's physiology (functionality) or the possible pathology (cancer uptake) of organs or tissues. As a result, it is possible to detect biochemical changes in the human body which can identify the onset of a disease process before anatomical changes can be viewed in other imaging modalities such as CT or MRI. Another practical feature with PET is that it can be used in conjunction with other diagnostic examinations to attain more absolute information about malignant tumors or benign lesions found in patients [16].

2.2.1 Radiopharmaceuticals

A radiotracer defines any radioactive substance used for tracing purposes, and in nuclear medicine it is routinely called a radiopharmaceutical. The typical radiopharmaceutical is designed to be identical or mimic a biochemical property of naturally occurring substances found in the human body. Today, the most commonly used radiopharmaceutical is the 18F-fluorodeoxyglucose (FDG). 18F-FDG has a glucose resembling structure where one of the hydroxy groups have been replaced with a fluorine atom. 18F-FDG is said to be a glucose analog, meaning that it chemically resembles the structure of normal sugar and will therefore enter the metabolism pathway for glucose. The molecular structure of both glucose and 18F-FDG can be seen in Figure 2.3. However, despite the resemblance between the two molecules, the 18F atom connected in the FDG tracer is indeed an unstable radioactive isotope [17].

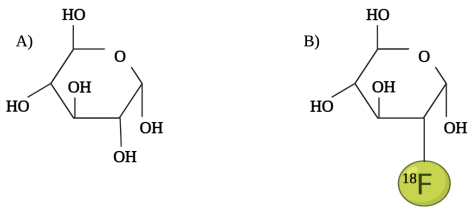


Figure 2.3: The figure shows the molecular structure for glucose in A) and 18F-FDG in B).

The fluorine-18 isotope decays by $\beta+$, in other words, a positron is emitted. Thereafter, the positron collides with an electron as it traverses through tissue, and resultantly, an annihilation process occurs and produces two anti-parallel 511 keV photons as can be seen in Figure 2.4. Therefore, when 18F-FDG is injected into the bloodstream of a patient, the tracer will selectively be absorbed by cells in the body that are highly metabolically active. It is due to this reason that FDG is quite effective in detecting cancer cells, owing to the fact that malignant cells have a much higher metabolic rate than normal healthy tissue and cells. It is for this simple reason that FDG-PET has become the standard in detection for certain types of cancer [17].

2.2.2 Standardized Uptake Value (SUV)

The distribution of the FDG radiopharmaceutical in a patient's body can be presented in a semi-quantitative manner by the Standardized Uptake Value (SUV). This value is defined by the equation:

$$SUV = \frac{C_{image}}{C_{injection}} \tag{2.1}$$

In the expression above the C_{image} is the concentration or activity in a specific region of interest in the PET image, while the $C_{injection}$ is the concentration or activity of the injected radiotracer. The injection concentration is usually normalized to the overall body mass of the patient and thereafter corrected for decay from the time of injection. This is done because of the long acquisition time of the PET scan. In other words, a SUV equal to 1 will therefore represent an uniformed distribution across the body. As a result, a high SUV in cells or tissues indicates a high concentration of the radiopharmaceutical used. Considering a lesion, a high SUV would determine the region where the lesion is localized, and the increased metabolism in these cells could possibly represent cancer [17].

2.2.3 Tomography and Image Reconstruction

The tomographic principle of PET exploit the detection of the annihilation processes caused by the 18F-FDG molecule to produce an image. The opposed photons from the result of the positron decay is localized on the PET scanner by using the concept of coincidence detection, and they can be detected by using pairs of collinearly aligned detectors in coincidence [18]. Since the two gamma rays are travelling in opposite directions, 180° apart, the detector pairs are designed in a ring-pattern which allows for radioactivity measurement along lines through organs of interest at a large number of angles and radial distances [18].

The ring is designed to report on instances where the photons are detected at separate locations within a short period of time. The angular information obtained is thereafter used in the reconstruction of the tomographic image of the regional radioactivity distribution [18]. Today's PET systems consist of multiple,

closely packed rings of detectors which enables simultaneous imaging across several imaging planes and allows for sampling in three dimensions [18]. The acquired data from a PET scan is the measurement of several coincidence events, and the raw data obtained are integrals along the line-of-coincidence over the activity distribution [18].

Considering two photons that are detected in a coincidence time window, often around 1-10 ns, the joint detection in the ring is called a true-coincidence event for the line that joins the two detectors [19]. This line is more commonly known as the Line-Of-Response (LOR). Figure 2.4 show a schematic coincidence event where 18F-FDG emits a positron and depicts how the anti-parallel photons are detected in the detector ring. More specifically, it is the total number of true-coincidence events that are detected by the two detector elements that are proportional to the total amount of radiotracer contained in the organ of interest, and it is this proportionality that is the key to PET imaging [19]. Due to this relationship, it is possible to process the coincidence events in order to accurately reconstruct the distribution of the injected radioisotope. In regards to PET imaging systems that uses Time-Of-Flight (TOF), it is the difference in arrival time of the two photons detected that is important. Knowing this differential timing, it is possible to localize the annihilation along the LOR [19][20].

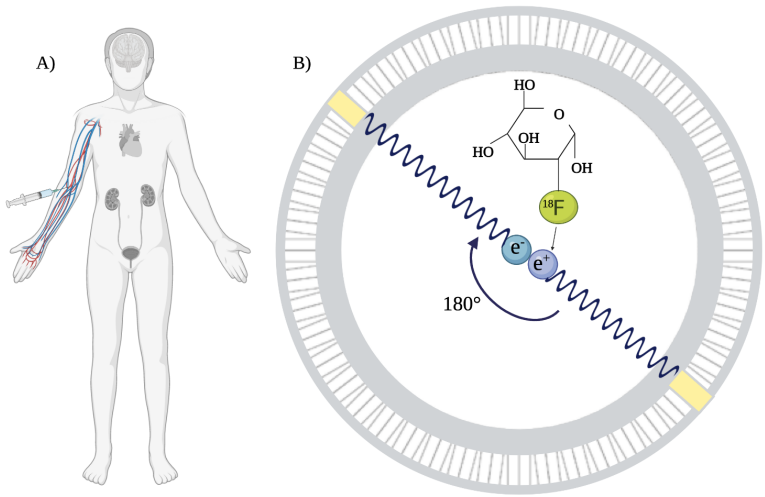


Figure 2.4: A) depicts a patient being injected with the radio-pharmaceutical 18F-FDG for a PET examination. The brain, heart, kidneys and urinary tract are organs that have physiological normal uptake of 18F-FDG. These organs will appear bright in the PET image which is caused by high glucose utilization, this is also the case for cancer tumors. B) shows how fluorine-18 fluorodeoxyglucose emits a positron, and how the positron annihilates locally with an electron. The result is the production of two anti-parallel 511 keV photons. The yellow regions show where the photons are detected and measured as near-simultaneous events on the detector ring [20].

Reconstruction Algorithm

In order to calculate the activity distribution of the isotope, the measurement of the coincidence events needs to be reconstructed. The approach to reconstruct the activity distribution is to displays the obtained data in an activity sinogram. The sinogram is first attenuation corrected and then filtered back-projection to reconstruct the distribution of the positron emitting isotope. There are different routes to acquire the back-projected image but in general the Fourier Slice Theorem is used to navigate between the Radon, object-, and Fourier-space [20]. However, more advanced methods such as iterative reconstruction algorithms are today the preferred choice over traditional filtered back-projection due to their superior image quality [18]. The basic principles of reconstruction algorithms are briefly as follows. The algorithm starts with an initial guess for the activity distribution and the data is forward projected according to the scanner geometry. The resulting projections are thereafter compared to the measured projection, and the error-projection is used for correcting the estimate. The new estimate found is then forward projected and the comparison between estimated and measured projections yields the next correction. The algorithm continues to iterate this loop until the measured and estimated projections agree within their statistics [18].

The spatial resolution in a PET image is limited by the positron range and the photon non-colinearity, where the PET spatial resolution in clinical systems is limited to 2.5 mm [21]. Moreover, the width of the detection elements in the tomograph will determine the width of the coincidence response function and consequently determine the image resolution [18]. In general, the angle between the two gamma rays is slightly noncollinear and varies in the range of 0.5° to 1°. This deviation from linearity causes a reduction in the PET image's spatial resolution. Similarly, the uncertainty as to where a given photon was absorbed in a thick detector element in the detector ring decreases the spatial resolution of the PET image [20]. Apart from this, scattering of the annihilation photons can also occur and limit the PET performance. Likewise can the photons that deviate between the random coincidences where separate photons from different annihilation events strikes the detector array simultaneously [20].

2.2.4 FDG-PET

The number of clinical applications for 18F-FDG and PET continues to increase, especially in the field of oncology. It has been known for many years that increased glycolysis is a distinctive feature of malignant tumors compared with normal tissues [22]. As previously mentioned, FDG is one of the more common radiotracers used for PET imaging since the FDG uptake reflects on the general metabolic process in most malignant tissues [23].

After injecting the FDG into the patient's bloodstream, the FDG is transported into viable cells by glucose transporter molecules, and thereafter it is the phosphorylated by hexokinase into FDG-6-phosphate. This process is identical to the phosphorylation of glucose into glucose-6-phosphate. However, unlike glucose-6-

phosphate, FDG-6-phosphate will not undergo further metabolism within the cell, making it a suitable tracer for imaging [23]. Besides, the dephosphorylation by glucose-6-phosphate is a relatively slow process. In combination with the fact that FDG-6-phosphate is not easily transported through the cell membrane, it results in the entrapment of FDG-6-phosphate within viable cells [23]. In summary, FDG will accumulate within malignant tumors cells and with the half-life of fluorine-18 being 110 minutes, it allows acquisition of FDG-PET images for 30-120 minutes [23].

There are several limitations of FDG-PET, and one such example is that the radiotracer may be taken up by physiological muscles and resultantly show increased activity in the PET images. However, such uptakes are easily identified when comparing it to either MRI or CT images. Similarly, the physiological uptake of FDG in the brain, heart, kidneys, liver, and bowel can be distinguished by the use of combined MRI or CT images [23]. Figure 2.5 shows such aforementioned physiological uptake. Even so, other false-positive findings in FDG-PET images may be more difficult to recognize. These findings include inflammatory changes that have been caused by inflammatory processes or infections such as bronchitis, viral infections, etc [23].

A clinically relevant question regards how FDG-PET images impacts the staging of cancer. For instance, in studies regarding lymphoma cancer, there has been variable results. The study conducted by Shöder et al. found that FDG-PET contributed to changes in clinical staging in 44% of patient cases, where 21% were upstaged and 23% were downstaged [24]. What is more is that there were made changes in treatment in more than 60% of cases in this study. Showing that FDG-PET can contribute to the significance of both positive and negative findings.

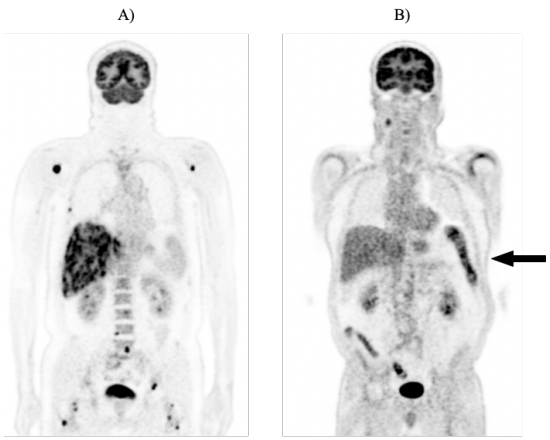


Figure 2.5: Two FDG-PET images of different DLBCL patients showing both pathological and physiological uptake of FDG. A) show the presence of several lesions in addition to brain, kidney, bladder, and severe liver uptake. In this patient case the liver is contaminated with disease. B) shows a patient with normal liver uptake in addition to normal FDG bowel uptake (arrow).

2.3 Magnetic Resonance Imaging

Magnetic resonance imaging (MRI) has since its discovery become one of the most important imaging modalities available to physicians today. Compared to other medical imaging devices such as CT or X-ray, MRI offers superior soft tissue contrast and is additionally an non-invasive imaging technology, to be more concise, it does not involve ionizing radiation. Moreover, the desired level of image contrast between different tissues can be achieved by simply adjusting the acquisition timing parameters of the setup. Therefore, MRI is an invaluable tool for both diagnostics and assessment of different diseases including several types of cancers [25].

2.3.1 Nuclear Magnetic Resonance and Magnetization

Magnetic Resonance Imaging (MRI) is emanated from Bloch and Purcell's discovery of how nuclei with a spin angular momentum (spin) interacts with a magnetic field. The nature of interaction between a spin and a magnetic field is known as Nuclear Magnetic Resonance (NMR) and is described by the equation:

$$\vec{\omega_0} = \gamma \vec{B_0} \tag{2.2}$$

Equation 2.2 describes how the static magnetic flux density, i.e., the magnetic field, $\vec{B_0}$ is experienced by a nuclei and how it results in the angular frequency of rotation ω_0 of the nuclear spin [26]. The γ is the gyromagnetic ratio, a unique constant for each nuclear isotope which is in possession of a spin and incorporates the mass and size of the particle in question. Besides, the angular frequency ω_0 is commonly referred to as the Larmor frequency. The Larmor frequency is equal to the electromagnetic radiation associated with the possible spin energy transitions induced by $\vec{B_0}$ [26].

All currently clinical use of MRI concerns the imaging of water molecules, specifically the protons of the hydrogen atom. To be more concise, it is based on proton NMR. Shortly, the NMR principle concerns the spin and the magnetic dipole moment which is locked to each other and needed to create precession. The hydrogen nuclei possesses an angular momentum $\vec{I}$ and due to the protons positive charge the corresponding spin additionally possess a magnetic momentum $\vec{\mu}$.

Despite the fact that NMR is exclusively a quantum mechanical process, its macroscopic manifestation is however, often well described by classical physics [26]. In the absence of an external magnetic field, the spins will be oriented randomly in the space and as a result there will be no net magnetization. On the other hand, if there is an external magnetic field, $\vec{B_0}$, the spins will either be aligned parallel or antiparallel to the direction of $\vec{B_0}$. In other words, the spins become polarized [25]. Even though the eigenvalues of the spin $1/2$ particle are spin-down or spin-up, the general quantum state for a spin can be any superposition of the two. Nevertheless, the ratio of the spin population is oriented parallel, or antiparallel, and given by the Boltzmann distribution:

$$\frac{N_+}{N_-} = e^{\Delta E/k_b T} = e^{\hbar\omega_0/k_b T} \tag{2.3}$$

here the N_+ and N_- are the population of parallel and anti-parallel spins, the ΔE is the energy difference between the two states, respectively k_b is the Boltzmann constant, T is the absolute temperature and ω_0 is the resonance frequency.

The magnetic field produces a torque, which is perpendicular to the orientation of the angular momentum. Subsequently, this torque causes the magnetization to precess in the direction of the field [25], and the equation for the motion of a spin in a magnetic field is expressed by the Bloch equation:

$$\frac{d\vec{\mu}}{dt} = \gamma\vec{\mu} \times \vec{B} \tag{2.4}$$

The Bloch equation describes the behaviour of the magnetization, where the γ again represents the gyromagnetic ratio , $\vec{\mu}$ is the magnetic momentum and $\vec{B}$ is the magnetic field [25].

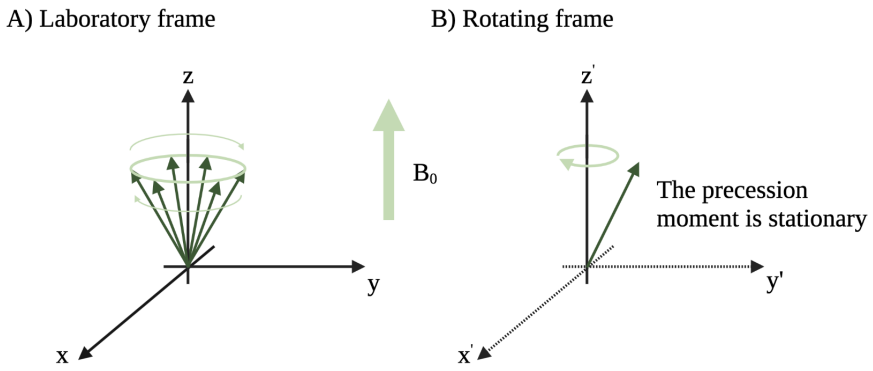


Figure 2.6: The figure shows A) the laboratory frame where the spins are precessing along the z-axis and B) the rotating frame where the spin rotates exactly at the Larmor frequency.

Initially, when looking at the spins in a stationary frame, i.e., the laboratory frame, the spins will precess at the Larmor frequency along the z-axis, the direction of the external field $\vec{B_0}$. On the other hand, if one considers a frame of reference, one which rotates at the $\vec{\omega_0}$, the spins that precess exactly at the Larmor frequency will appear stationary. However, the spins that precess at another frequency, ω, will appear to precess with a frequency ω_r in the rotating frame [25]. In this rotating frame $\omega_r = \omega - \omega_0$. The described frames are depicted in Figure 2.6.

The net magnetization, i.e., the total magnetization, is a vector sum of all the spins which are contained within a given voxel. At equilibrium, the net magnetization $\vec{M}$ will be aligned along the $+z$ direction, which is the direction of the external field $\vec{B_0}$. Henceforth, the behavior of the net magnetization vector $\vec{M}$ as a result of magnetic interactions is classically expressed by the Bloch equation:

$$\frac{d\vec{M}}{dt} = \gamma\vec{M} \times \vec{B_0} \tag{2.5}$$

Here, the $\vec{M}$ is stationary and does not precess about any axis as it is aligned along the z-axis. Therefore, it remains constant and oriented along the direction of the external magnetic field as can be seen in Figure 2.7. If one were to apply another magnetic field, the magnetization would deviate from its equilibrium position and could even begin to precess about an effective magnetic field [25].

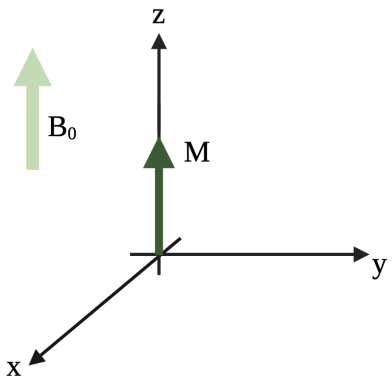


Figure 2.7: The figure depicts the net magnetization M in a voxel and how it is aligned along the direction of the applied magnetic field B_0. The magnetization M is stationary and does not precess around the z-axis [25].

Moreover, if one were to consider an external RF field, i.e., a time-varying B_1 field, which is resonating at ω_0 and is applied to the spins in a magnetic field B_0, then, according to Bloch equation, the magnetization will precess at an effective magnetic field [25]. This magnetization is the vector sum of both the external field B_1 and the static B_0 field. Observing the magnetization in the rotating frame will simplify the equation as the B_1 field will appear static due to the frequency of the external field being identical to that of the rotating frame. Besides, the magnetization is initially aligned along the z-axis in the rotating frame, however, it will begin to precess about the direction of the B_1 field. In other words, if B_1 is aligned along the x-axis, the magnetization will precess around the same axis. The magnetization will continue to precess about this axis for as long as the external RF field is applied. Naturally, to detect the MR signal, the B_1 signal needs to be applied long enough to cause for instance a 90° rotation in order for the magnetization to become aligned along the y-axis at the end of the RF pulse [25]. Additionally, it is possible to use a flip angle lower than 90° and still achieve a signal. Nonetheless, once the magnetization is rotated into the transverse plane and the RF pulse is removed, the spins will precess about the static B_0 at the Larmor frequency in the stationary frame in accordance with the Bloch equation [25] as depicted in Figure 2.8.

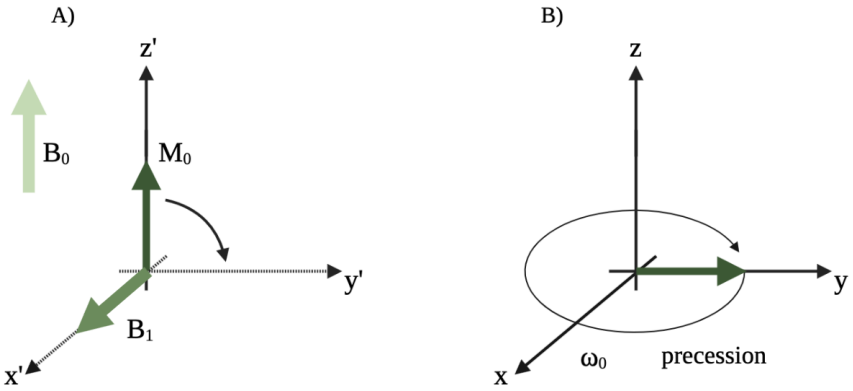


Figure 2.8: The figure shows A) the precession of the magnetization as a result of the RF field B_1 which resonates at the Larmor frequency in the rotating frame. B) depicts a situation where the B_1 has been removed once the magnetization has reached the transverse plane. The magnetization is now precessing around the z-axis in the stationary frame [25].

The precession of the spins caused by the magnetic field will not continue indefinitely, and will as mentioned shortly be followed by rotation of the magnetization into the transverse plane. Once the radio frequency pulse has perturbed the magnetization and the magnetization has been tilted away from the equilibrium position along the z-axis, it will be forced back to its equilibrium state [25]. The longitudinal T1 relaxation forces thermal equilibrium, whereas the transverse T2 (spin-spin) relaxation leads to signal loss due to dephasing.

2.3.2 Relaxation

T1 Relaxation

The longitudinal T1 relaxation is the process of where the net magnetization $\vec{M}$ returns to its initial maximum value, M_0, in an exponential fashion. As a result, $\vec{M}$ is once more parallel to the external field $\vec{B_0}$ [27]. The relaxation can be described by the following equation:

$$M_z(t) = M_z(0)e^{-t/T_1} + M_0\big(1 - e^{-t/T_1}\big) \tag{2.6}$$

here M_0 is the longitudinal magnetization immediately after the RF excitation. The energy of the spin system will decrease as the longitudinal component, M_z, returns toward M_0. This happens due to the fact that there are statistically more spins which favours the lower energy orientation compared to that of the higher energy orientation. The energy has to leave the spin system for the T1 relaxation to take place, and this energy loss is unrecoverable. The energy loss is transferred into nearby atoms, nuclei, or molecules through collisions, rotations, or electromagnetic interactions [27].

T2 Relaxation

T2 relaxation differs from T1 relaxation due to it being a process where the transverse component (M_{xy}) of the net magnetization $\vec{M}$ dephases. The spins involved do not only interact with the internal environment, but they also interact with one another and therefore results in a spin-spin relaxation. Each polarized molecule, the electronic origin of the magnetic field, will slightly alter the field in the spins' surroundings [25]. As a consequence, the spins in close proximity to one another will experience this additional field which will slightly alter its precessional frequency. Due to the spins constant motion, the precessional frequencies of each spins are in constant flux and will therefore result in an increase in phase coherence where different spins accumulate different amounts of phase over time [25]. It is the time constant T2 which describes the loss of coherence that results in an exponential decay of the signal in the transverse plane. The following equation describes the T2 relaxation:

$$M_{xy}(t) = M_{xy}(0)e^{-t/T_2} \tag{2.7}$$

Here $M_{xy}(0)$ is the initial transverse magnetization following the RF excitation.

The tipping of the net magnetization into the transverse plane does not result in all the spins being locked in phase. The same spins, which previously had the statistical preference of the low energy state prior to the RF pulse, have now been rotated into the transverse plane [28]. This rotation of spins is commonly referred to as phase coherence in the xy-plane [28]. Immediately after the RF pulse has been applied, the M_z has been converted to a net transverse magnetization, M_{xy}, by being tipped into the transverse plane.

Moreover, after the tipping of the magnetization and the removal of the RF pulse, the transverse spin components and the corresponding vector sum $\vec{M_{xy}}$ will begin to precess within the plane at the Larmor frequency [28]. It is due to the sweep of the M_{xy} that a current is induced in the receiver coils of the setup. This is solely responsible for generating the MR signal [28].

As previously mentioned, the T1 relaxation occurs due to the spins' energy exchange with its internal environment. Should for instance this energy exchange affect one of the spins contributing to M_{xy}, then both of the longitudinal and transversal components of the angular momentum would end up being randomly changed [28]. And as a consequence, the affected spins would lose phase relation with other spins. Therefore, any process causing T1 relaxation would also result in a T2 relaxation. As a result, both the T1 and T2 relaxation can be incorporated into the Bloch equation, and thus provide a more complete description of the MR signal [25].

2.3.3 Free Induction Decay (FID)

After the excitation of the magnetization with an RF pulse, the M_z returns to its equilibrium state M_0 through a T1 relaxation. At the same time, the signal in the

transverse plane will decay exponentially with a T2 time constant [25].

Thereupon, looking at the spins in the laboratory frame of reference will show that the spins are precessing at the Larmor frequency. At the same time the spins decay with the amplitude of the T2* relaxation [25]. Hence, it is this evolution of the signal in the transverse plane that give rise to signal named the Free Induction Decay (FID). This is why the spins in a homogeneous magnetic field will precess at a single Larmor frequency. Subsequently, the detected signal will be a perfect sinusoid which is modulated by the decaying exponential function with the T2* time constant. This resulting FID signal is a damped sine wave, as can be seen in Figure 2.9.

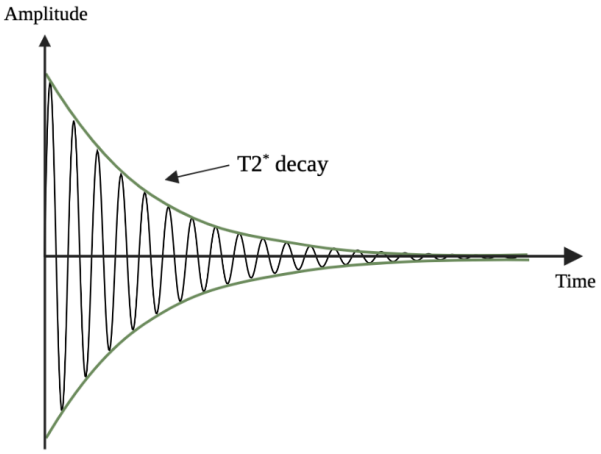


Figure 2.9: The figure depicts how the damped sine wave, the Free Induction Decay signal (FID), decays exponentially with T2*.

2.3.4 Pulse Sequences

A pulse sequence in MRI is the measurement technique that the actual MR image is obtained from. These pulse sequences are a programmed set of changing magnetic gradients and depends on different parameters including echo time (TE), repetition time (TR), inversion pulses, diffusion weighting etc. [29]. The sequences are often grouped accordingly to the type of sequence they represent, e.g. spin echo or inversion recovery, or it is possible to group the sequences by their general image weighting e.g. T1 or T2. Granted, one requires multiple sequences to fully evaluate a tissue [29]. In the following sections the HASTE and DWI sequences will be introduced. Additionally, a very brief explanation of k-space sampling is included to gain a better understanding of the HASTE sequence.

K-Space Sampling

In short, k-space is an array of numbers representing the spatial frequencies from the MR image, where each k-space point contains phase and spatial frequency information regarding every voxel in the image [30]. The k-space is filled with new data points by applying phase- and frequency-encoding gradients. As a matter of fact, the use of gradients is the fundamental concept of spatial encoding. Each voxel in the MR image maps to every point in k-space and by following the trajectories in k-space the data can be reconstructed into a MR image [30].

Currently, clinical MR imaging sequences often use Cartesian k-space sampling [31]. In a Cartesian trajectory, the sampling points are commonly placed on a square grid. After applying the necessary RF pulses and gradients to fill the k-space for the desired sequence, the data is reconstructed into the MR image by applying the Fourier transform [31]. There are several other k-space trajectories which can be used for sampling such as radial, zig-zag, and spiral [32].

Half-Fourier Acquisition Single-shot Turbo Spin Echo (HASTE)

Half-Fourier Single-shot Turbo Spin Echo (HASTE), is an echo-planar fast spin echo sequence which acronym clearly states what it entails. HASTE is a single-shot technique, meaning that all data from k-space is obtained after a single excitation of a 90° pulse [33]. As a consequence, the echo times in a HASTE image is relatively long and hence, is typically T2-weighted. The Figure 2.10 below depicts a typical HASTE sequence:

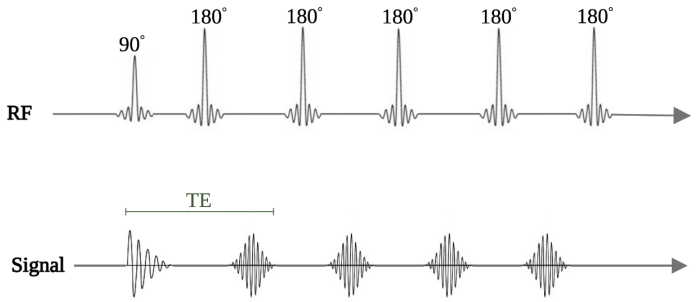


Figure 2.10: The figure shows a fast spin echo HASTE sequence with repetitive 180 degree RF pulses. The echo time (TE) is also included.

Diffusion Weighted Imaging (DWI)

Diffusion Weighted Imaging (DWI) detects motion of water molecules and can therefore possibly detect inherent alterations in underlying tissue cellularity and

organization due to the presence of tumor cells. There are different techniques for generating the diffusion maps, nevertheless, it is the spin-echo echo-planar sequence (SE-EPI) that is most commonly used today [34]. In addition, it is possible to measure the degree of diffusion weighting in the image. The degree of diffusion is denoted by b-value and given by the equation:

$$b = \gamma^2 G^2 \delta^2 (\Delta - \delta/3) \tag{2.8}$$

here, γ is the gyromagnetic ratio,G is the amplitude, δ is the time of the applied gradients and Δ is the duration between the gradients [35]. The Figure 2.11 below shows a typical DWI sequence and includes all the parameters necessary to calculate the b-value:

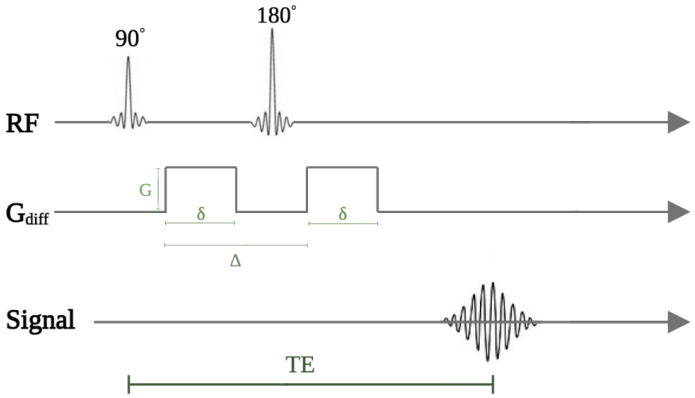


Figure 2.11: The figure shows a typical DWI imaging sequence. G is the amplitude, δ is the time of the applied gradients, TE is the echo time and Δ is the duration between the gradients.

2.3.5 Hybrid PET/MRI

Today, biomedical imaging has an important role in all stages of cancer management which includes biopsy guidance, screening, diagnosis, staging, prognostic assessment, selection of treatment plan, prediction, and monitoring of treatment response and lastly, assessment for recurrent disease [36]. Hybrid imaging modalities such as PET/MRI combines the molecular and quantifiable functional information from PET and the unique tissue characterization from MRI [36]. The result of such hybrid images provides clinical advantages not possible with other modalities [36].

The Siemens Biograph mMR hybrid PET/MRI has a fully-integrated system design in which the PET detector rings are placed inside the MR gantry [36][37]. The solid state PET detectors are not only compatible with the external magnetic fields of the MR scanner, but they are additionally smaller than the traditional PET

detectors, which facilitate the integrated design [36]. Due to the design, truly simultaneous PET and MR imaging is achieved. This has practical advantages such as improved lesion detection and co-registration [36]. In fact, potential advantages for simultaneous imaging of dynamic processes visualized on both PET and MRI is also achievable [36]. Other advantages of PET/MRI, in addition to comparisons to PET/CT, are shown in Table 2.2.

Table 2.2: Advantages of hybrid PET/MRI versus hybrid PET/CT [36].

Attribution	PET/MRI	PET/CT
Lesion detection	Improved lesion detection in organs such as the brain, breast, liver, kidneys and bone	No advantage
Lesion alignment	Better alignment of simultaneously acquired PET/MRI data compared with PET/CT	No advantage
Quantitative accuracy	Improved quantification by MRI-based motion correction without additional radiation	Industry standard (i.e., attenuation) is based on density from CT
Scanning time	No advantage	Currently, the PET/CT body scanning protocols are faster
Radiation exposure	Lack of CT reduces radiation exposure (up to 50% depending on CT protocol)	No advantage
Patient convenience	Single appointment for patients who require both PET and MRI; less scanner time overall	No advantage
Multi-parametric quantitative imaging	Expanded capabilities such as perfusion MRI, DWI and spectroscopy	No advantage
Availability	No advantage	More clinically available

2.4 Deep Learning

Deep Learning (DL) is a branch of machine learning that teaches the computer to do what comes naturally to humans, learning by example. In deep learning, a computer model is trained how to perform classification tasks directly from images, sound, or text. These DL models can achieve state-of-the-art accuracy, and sometimes they manage to exceed human level performance [38]. The DL models are trained by using artificial neural networks which contain several layers and utilize large sets of labeled data. The artificial neural networks tries to simulate the behaviour of the human brain in order to learn from the large amounts of data provided. A single layer neural network can make approximate predictions, however, the addition of several hidden layers can optimize and refine the network for better accuracy [38]. Today, DL operates multiple artificial intelligence applications that result in improved automation. Moreover, the performance of both analytical and physical tasks is done without human intervention [39], and deep learning applications are for instance used for the purpose of automated driving, found in medical devices, and it is suitable for image segmentation.

2.4.1 Artificial Neural Network

An Artificial Neural Network (ANN) is an adaptive system which learns by using interconnected nodes, commonly referred to as neurons, in a layered structure. This layered structure is made to resemble a human brain and an artificial neural network can therefore learn from using data. By exploiting the data provided, the neural network can be trained to recognize patterns, classify data, and forecast future events [40]. The neural network breaks the input into different layers of abstraction, and can therefore be used to recognize patterns in images in the same way the human brain does. The neural networks' behaviour is defined by the way its individual elements are connected and by the strength, also known as weights, of those connections [40]. During the training of the network, the weights are automatically adjusted according to specified rules and the they will continue to be adjusted until the ANN performs the requested task correctly [40].

As previously mentioned, neural networks are a type of deep learning approach that are heavily inspired by the human brain and adapts how neurons signal to each other. The neural networks are in particular suitable for modeling non-linear relationship due to the fact that they are typically used for performing pattern recognition and signal classification[40]. Deep neural networks have become acknowledged for their proficiency at complex identification applications and these networks can be found in technologies such as face recognition, text translation, and voice recognition.

The networks themselves are as aforementioned inspired by the biological nervous system. The neural network combines several processing layers where it uses simple elements which are operating in parallel. As can be seen in Figure 2.12, the network consists of an input layer, one or several more hidden layers

and an output layer. In each of these layers, there are several neurons (nodes) that have learnable weights and biases, and these neurons uses the outputs of all nodes in the previous layer as input to the next. When such an input enters the neuron, the input becomes multiplied by a weight value and results in an output. This output is either observed or passed to the next layer in the network. Consequently, all of the neurons in the network are interconnected with each other throughout different layers. Additionally, each neuron is often assigned a weight that is adjusted during the training process. This weight will decrease or increase with the strength changes of the specific neuron's signal [40].

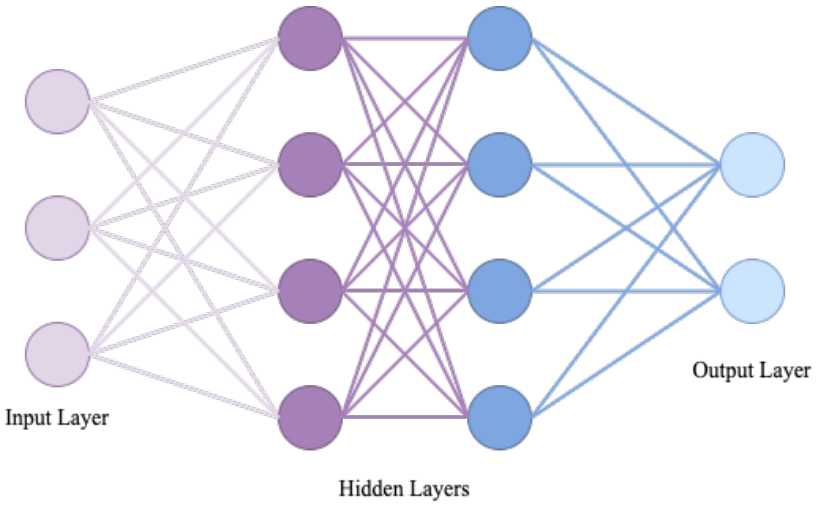


Figure 2.12: The figure depicts a model of a fully connected neural network. The model consists of an input layer, two hidden layers and an output layer, where each filled-in circle represents an artificial neuron (node).

Deep learning refers to artificial neural networks with several layers, whereas neural networks, which only consisting of two or three layers of connected neurons, are referred to as shallow neural networks [40]. One of the reasons why deep learning has become popular is due to the fact that it is eliminating the need to extract features from images and the need for automatic parameter tuning [40]. This was previously challenging in the application of machine learning to both image and signal processing [40]. Even though feature extraction can be excluded from image processing, some form of feature extraction is still often applied to signal processing tasks in order to improve the model's accuracy [40].

2.4.2 Convolutional Neural Network

A Convolutional Neural Network (CNN) is an artificial network which learns directly from data and eliminates the need for manual feature extraction [41]. CNNs are therefore particularly useful for finding patterns in images. By finding these

patterns, the network can, for instance, recognize objects or faces, and can in addition be quite effective in classifying non-image data [41]. Due to the convolutional neural networks' superior performance with images, the network are a key technology in applications when it comes to visually detecting the presence, or absence, of cancer, and is therefore an asset for segmentation of cancerous nodes in medical images [41].

Convolutional neural network consists of three different types of main layers: Convolutional layer, pooling layer, and fully-connected layer. The first layer encountered in a CNN is the convolutional layer. The convolutional layer may be followed by additional convolutional layers or pooling layers, nevertheless, the final layer encountered is the fully-connected layer. The convolutional neural network increases in its complexity with every layer and thus can identify larger portions of the input image. The simple features of the image, such as the colors and the edges, are handled by the earlier layers. As the image is progressing through the layers the organs, larger lesions, and anatomical structures present are recognized by the network. Therefore, as the CNN is able to recognizing these structures, it can identify abnormalities in the image [42].

Convolutional Layer

The majority of computations in the network occurs in the convolutional layer, and this layer is thus considered the core building blocks of a convolutional neural network. The convolutional layer requires input data (image), a filter, and a feature map. A colored input image will have three dimensions: height, width, and depth. These dimensions corresponds to the RGB in the image. The mentioned filter, the feature detector, will move across the respective fields of the image and check if the feature is present [42]. In other words, the layer will perform a convolution. The aforementioned feature detector is a 2D array of weights which is representing the image and the filter is typically a 3x3 matrix [42]. The filter is applied to an area of the image and the dot product is thereafter calculated. The calculated product is between the input pixels and the filter, and this product is further fed into an output array. Afterwards, the filter will shift by a stride and repeat the process until it has moved across the whole image. It is the final output from the series of dot products which is called the feature map which is a two-dimensional (2-D) array of weights [42].

Figure 2.13 shows a 3x3 filter operation in the convolution layer where each output value in the feature map is not necessarily connected to each pixel in the image. The output value only needs to be connected to the receptive field, i.e. where the filter is being applied in the image. For this reason, the convolutional layer is commonly referred to as a partially connected layer [42].

Ultimately, the purpose of the convolutional layer is to convert the input image into numerical values. This allows the neural network to interpret and extract relevant patterns from the image [42].

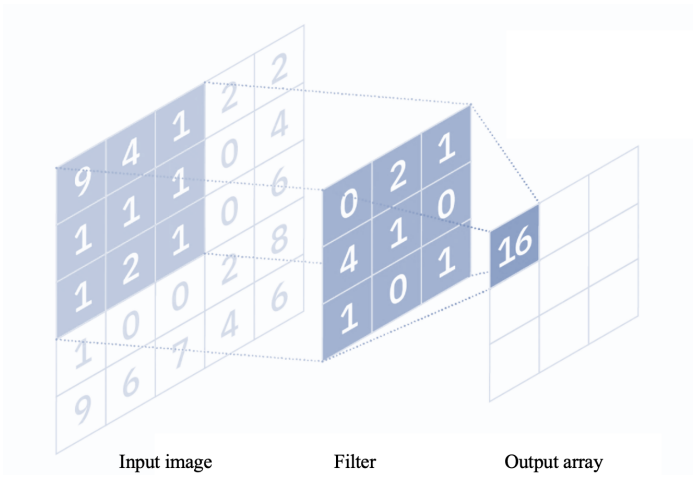


Figure 2.13: The figure depicts the 3x3 filter operation that takes place in the convolutional layer [42].

Pooling Layer

Another typical layer encountered in a convolutional neural network is the pooling layer. This layer causes a dimensionality reduction in the image, which reduces the numbers of parameters in the input and is for this reason often referred to as a downsampling layer [42]. The pooling layer operates by sweeping a filter across the entire input, but in contrast to the convolutional layer, the filter applied does not have any weights. Instead, the filter applies an aggregation function to the values within the receptive field which populates the output array [42] which can be achieved with maximum pooling. In order to perform maximum pooling, the filter moves across the input image, and selects the pixel with the maximum value and append it in the output array. Another possibility is to use average pooling, the only difference is that it is the average pixel value that is sent to the output array and not the maximum value as in max pooling. The Figure 2.12 below shows how the max pooling returns the maximum value of the filtered section of the image.

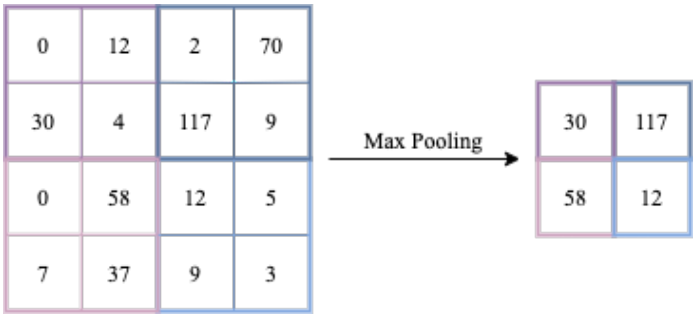


Figure 2.14: The figure shows how the maximum pooling returns the maximum value from the section of the image which is covered by the filter [43].

Even though a large part of the information is lost in the pooling layer, the layer has a number of benefits to the convolutional neural network. The pooling layer help reduce the complexity of the CNN, improves the efficiency, and additionally limit the risk of overfitting [43]. To encapsulate, the pooling layer simplifies the output by performing a nonlinear downsampling, which reduces the numbers of parameters that the network needs to learn in order to recognize objects and patterns [41]. Together, both the convolutional and pooling layer form the i-th layer of the CNN. In order to capture the low-level details even further in the input image, the number of the applied layers in the CNN needs to be increased. Consequently, by increasing the numbers of layers, it costs more computational power [43].

Fully-Connected Layer

The fully-connected layer in a convolutional neural network connects directly to a node in the previous layer, hence the name fully-connected. As previously stated, the pixel values of the input image are not directly connected to the output layer in any of the partially connected layers, i.e., the convolutional layers. However, it is in the fully-connected layer that each node in the output layers are connected directly to a node in a previous layer as is shown in Figure 2.12. The function of the fully-connected layer is that it performs the task of classification based on the features extracted from previous layers and their corresponding different filters [42].

To recapitulate, the operations of the convolutional-, pooling-, and fully- connected layers are repeated over tens or hundreds of layers. By doing this, the CNN trains the network in each layer and learns to identify different features in the input image. Like any other traditional artificial neural network, a convolutional neural network consists of neurons with weights and biases. The network model learns these values during the training process and thereafter continues to update the values with each new training example. However, in a CNN, both the weights and bias values are the same for every hidden neuron in a specific layer [41]. In other words, this means that all of the hidden neurons will detect the same feature, e.g. a pattern, in different regions of the input image. It is for this sole reason that the CNN network is tolerant to the translation of objects in an image [41].

Neural Network Training

The main goal of neural network training is to optimize and determine the best set of weights and biases that maximizes the networks' accuracy [44]. This is done by periodically updating the biases and weights in order to improve the output. For the purpose of automated segmentation, the goal is to minimize the errors between the ground truth and the predicted output. Thus, a loss function must be chosen to calculate the error of the model during the optimization process [45]. The loss function can be defined as $L = L(x, \theta)$ which depends on the input denoted by x and the parameters θ [46]. There exists two different forms of loss,

the first being training data loss which is often denoted by *loss*. The second loss represents the loss from the validation data which is more commonly referred to as *validation loss* [47]. Only the parameters θ can be updated during training since the input data is predetermined. For this reason, the loss function will only be dependent on θ, which reduces the equation to $L = L(\theta)$.

Minimizing $L(\theta)$ can be done by trial and error. However, a more robust way is to use an algorithm method called gradient descent [46]. The algorithm starts out with a random guess at the parameters. Thereafter, it determines which direction the loss function steeps downward the most, this with respect to changing the parameters. Thereafter, the algorithm will step slightly in that direction [44].

In order to find the direction the loss function steeps downward the most, the gradient i.e., the partial derivative of all parameters, is calculated:

$$\nabla L(w) = \left(\frac{\partial L(w)}{\partial w_1}, ..., \frac{\partial L(w)}{\partial w_n} \right) \tag{2.9}$$

Here the ∇ is the vectoral differential operator and $L(w)$ is the loss as a function of weights, w, in n-dimensions [44].

The way the gradient descent algorithm works can be summarized in four simple steps [48]:

1. Configuration of parameters which are either random or non-random initialization.
2. Calculate the gradient of the loss function, $L(\theta)$.
3. Move in the opposite direction of the calculated gradient.
4. Repeat step 2-4 until the loss function is minimized below a threshold decided beforehand.

How long the algorithm should move in the opposite direction of the gradient is determined by an update function, and this function updates the weights at a time (t-1):

$$w^t = w^{t-1} - \eta_l \nabla L(w) \tag{2.10}$$

Here η_l is denoted as the learning rate and it dictates the weighted gradient of the loss function [48].

Furthermore, the weights have to be updated with respect to several parameters in order to improve the neural network. This can be executed by implementing the back-propagation algorithm [48] which contains several iterative rules for the computation of the partial derivative of the loss function L with respect to the parameters (the weights and biases) in the network. The derivation of the updated expression is not shown here, but is derived nicely here [48]. Nevertheless, the expression gives information of how quickly the loss changes when the weights and biases are updated. In other words, the back-propagation algorithm is not only a fast algorithm for learning, but it additionally gives detailed insights into how the updated biases and weights influence the behaviour of the neural network [48].

In fact, for each forward propagation there will be a backward propagation updating and adjusting the parameters. Hence, after an epoch (iteration) has been completed, the training data used has been passed forward and backwards thorough the neural network. It is common that the datasets used for network training are quite large, and therefore the sets are often divided into batches. A batch size is defined as a fraction of training examples used in every epoch [49]. Generally, in order to execute a back propagation, the neural network needs to perceive the difference between the predicted value and the actual value, the loss. The binary cross-entropy is a loss function that is often used in binary classification tasks and for image segmentation [50]. Binary classification is defined as a problem with two class labels where one class depict the normal condition whereas the other class represents the aberrant state [51]. Additionally, there are classification tasks with more than two class labels, and this is called multi-class classification. Unlike binary classification, the multi-classification cannot distinguish between pathological and normal results [51]. The aforementioned binary cross-entropy loss function calculates the loss between the ground truth and the predicted values results in an output between 0 and 1 [50], deeming it a binary classification task.

As previously mentioned, the overall goal is to minimize the loss function in order to reduce the errors between the ground truth and the model prediction. As a rule, this is achieved by optimizing and updating the weights and biases. The optimiser "RMSprop" is commonly used for this task and it was therefore implemented. RMSprop divides the learning rate for a given weight by a running average of the magnitudes of the recent gradients of that specific weight [52].

Evaluating a Neural Network

Since the task of semantic segmentation is simply to predict the class of each pixel in an image, it is necessary to evaluate the performance of the model on an absolute performance metrics [53]. The aforementioned binary loss function will show a relative metric performance which indicates whether or not the model is moving in the right direction, but it will not tell how far away from the goal you actually are. Essential metrics are therefore needed to evaluate the segmentation model properly, and one such metric is pixel accuracy. This metric simply reports the percentages of pixels in the input image which were correctly classified and it is commonly reported for each class [53].

The accuracy is calculated as the ratio between the number of correct prediction to the total number of predictions, which can be computed by finding the True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN):

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.11}$$

Nonetheless, a high pixel accuracy does not always imply superior segmentation ability as this metric can provide misleading results [54]. If for instance the class

representation is small within the image, the measure of percentages of pixel accuracy will be biased in mainly reporting how well the model manages to identify the negative cases i.e., where the class is not present in the image [53].

By the same token, metrics such as precision and recall also make use of True Positive (TP), True Negative (TN), False Positive (FP), and False Negative (FN). The precision reflects how reliable the model's accuracy is at classifying samples as positive and it is often referred to as the Positive Predictive Value (PPV) [55]. It is calculated as the ratio between the number of positive samples from the image which is correctly classified to the total number of samples classified as positive [55]. The latter positives may either be correctly or incorrectly classified [55].

$$Precision = \frac{TP}{TP + FP} \quad | \quad PPV = \frac{TP}{TP + FP} \tag{2.12}$$

Recall, on the other hand, is calculated as the ratio between the number of positive samples which are correctly classified by the model as positive to the total number of positive samples [55]. To put it more simply, the recall metric evaluates the model's ability to detect positive samples. Sensitivity and True Positive Rate (TPR) are other words that are commonly used for recall.

$$Recall = \frac{TP}{TP + FN} \quad | \quad TPR = \frac{TP}{TP + FN} \tag{2.13}$$

Owing to the equations for precision and recall shown above, a model which results in a high recall but a low precision will classify the majority of the positive samples correctly, but it will consequently result in many false positives. To be more concise, the model often classifies negative samples as positive [55]. In the same way, when a model results in a high precision but a low recall it indicates that the model is accurate when it classifies a sample as positive, but it can only classify a few positive samples [55].

The False Negative Rate (FNR) is calculated as the ratio of values that are actually positive but were predicted negative of the total number of positives [56]. The equation below shows the connection between the FNR and the TPR:

$$FNR = \frac{FN}{TP + FN} = 1 - TPR \tag{2.14}$$

The False Discovery Rate (FDR) is defined as the expected proportion of false discoveries among all discoveries [57]. In other words, it is showing the ratio of false positives classifications that were predicted. The equation below shows the connection between the FDR and the PPV:

$$FDR = \frac{FP}{TP + FP} = 1 - PPV \tag{2.15}$$

The Negative Predictive Value (NPV) is another metric suitable to evaluate the model's performance. The NPV is defined as the proportion of predicted negatives which are real negatives [58]. In other words, the resulting value reflects the probability that a predicted negative is a true negative [58].

$$NPV = \frac{TN}{TN + FN} \tag{2.16}$$

An additional metric that can be used to evaluate the model's performance is specificity. Specificity is the model's ability to predict true negatives of each available category [59] and it is calculated accordingly:

$$Specificity = \frac{TN}{TN + FP} \tag{2.17}$$

Another commonly used metric is the Dice Score (DS). This metric evaluates how similar the automated segmentation, the model's prediction, is to the ground truth. Simply put, the DS is defined as:

$$DS = \frac{2 \cdot A \cap M}{A + M} \tag{2.18}$$

where A are the pixels which have been automatically segmented while M are the pixels from the manual segmentation. A DS close to 1.0 indicates a perfect overlap of the automated segmentation and the ground truth, whereas a score close to 0.0 indicates incorrect predictions.

2.4.3 Overfitting

One of the fundamental issues when it comes to machine learning is overfitting, a concept which prevents the trained model from performing. Overfit models arise due to the difficulty in coping with pieces of information in the testing data which differs from the training data set provided to the model. This is because an overfit model have memorized all the data provided in the training, including the unavoidable noise, instead of learning the discipline hidden behind the provided data [60]. In other words, because of overfitting, the model will perform exceptionally well on the training set provided, while predicting poorly and be inadequately fitted on the testing data.

In general, an overfit model has high variance and low bias [61]. This happens because the overfit model has learned the training data too well and as a consequence the model's performance varies widely with new unseen data in the testing dataset [61]. An overfit model is easy to spot when it is evaluated on both the training set and on a holdout validation dataset. Thus, by plotting the learning curves, lines that show the performance of the model during training, an overfit model will show a familiar and repetitive pattern [61].

There are two possible ways to approach an overfit model. The first option is to reduce the overfitting by training the artificial network on more data examples, i.e., increasing the training data. Whereas the second course of action is to reduce the overfitting by changing the complexity of the network itself. The advantage with deep neural networks is that the networks performance will improve for every dataset it is given. Since a model has sufficient capacity to overfit a training dataset, reducing the model's capacity will result in the likelihood of the model

actually overfitting the training dataset to the point where it no longer overfits the model. The complexity of the model, the capacity, is defined by the structure of neurons and layers with their corresponding weights. By reducing the complexity of the neural network, the overfitting may be reduced by either changing the network structure (the number of weights) or by changing the network parameters (the values of the weights) [61].

2.4.4 U-Net

A U-Net is an evolved version of a convolutional neural network that was first introduced by Ronneberger et al. in 2015 [62]. The network was first and foremost designed to process biomedical images, but it differs from a general convolutional neural network because it manages to distinguish whether the medical image contains disease and can additionally localize the area of abnormality. The reason the U-Net is able to localize and distinguish borders is because it does classification on every voxel in the medical image, and therefore the input and output share the same size [63].

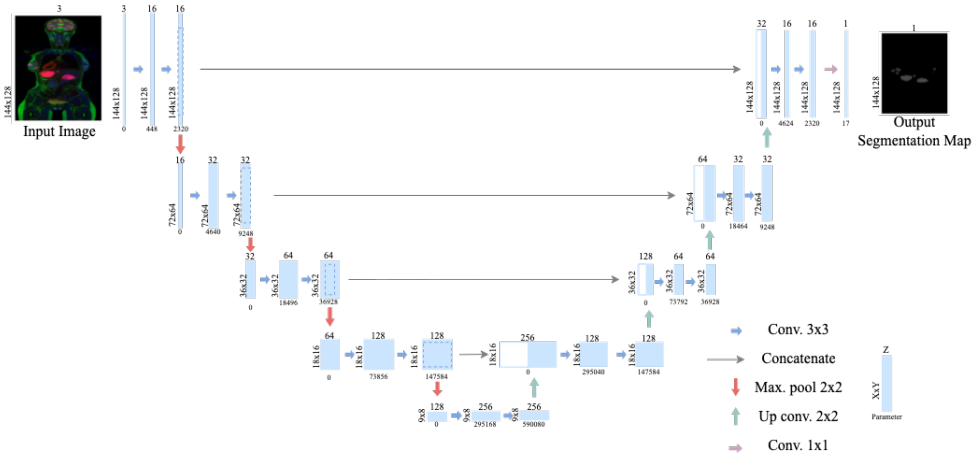


Figure 2.15: The figure depicts the architecture of the artificial neural network 2D U-Net [62]. The function of each arrow type is also included in the figure. At first sight, one notices the "U" shape of the set-up, hence the name U-Net.

Figure 2.15 depicts the architecture of a 2D U-Net, which at first sight has a symmetry resembling an "U" and thus the assigned name. The U-Net consists of a contracting path on left side and an expansive path on the right. The contracting path follows the typical architecture of a convolutional network where there are repeated applications of two 3x3 convolutions, depicted with blue arrows in the figure. These unpadded convolutions are thereafter followed by a rectified linear unit and a 2x2 max pooling operation with stride 2 for downsampling [62]. At every downsampling step, shown with down-pointing red arrows, the number of feature channels are doubled.

As can be seen in Figure 2.15 above, every step in the expansive path (right side) consists of an upsampling of the feature map. The upsampling is followed by a 2x2 convolution which is often referred to as an "up-convolution" (green arrows) that halves the number of feature channels. Moreover, a concatenation is preformed with the correspondingly cropped feature map from the contracting layer (grey arrows), which is thereafter followed by two 3x3 convolutions (blue arrows). This convolution is again followed by a rectified linear unit. The cropping is necessary due to the loss of border pixels in each convolution. At last, in the final layer, a 1x1 convolution (pink arrow) is used to map each component feature vector to the desired number of classes. Visibly in the figure, the U-Net consists of 23 convolutional layers, i.e., the total numbers of blue, grey, and pink arrows [62].

To summarize, the U-Net expands the feature channels by combining every contracting path to the expansive path. To be more precise, it means that the model can reserve more useful features from every layer, and it is due to this reason that U-Net is one of the most practical models to date [64].

Chapter 3

Materials and Methods

This chapter introduces the relevant materials and methods used in this thesis. The chapter is divided into six sections which contains useful information about the lymphoma dataset, a description of how the manual segmentations were performed and what types of software that were utilized. The chapter additionally includes information regarding how the image pre-processing, data augmentation, network architecture, and network training were built and executed. The majority of the methods used in this thesis is a continuation of what were presented in the specialization project excluding the subsections concerning image normalization and k-fold cross-validation.

3.1 The Lymphoma Dataset

The information regarding the image acquisition and the lymphoma dataset used in this thesis was attained from Live Eikenes, a professor at the Department of Circulation and Medical Imaging at NTNU. The lymphoma study was conducted at St. Olavs Hospital in Trondheim and was approved by the Regional Committee for Ethics in Medical Research (REK-Midt #2014/1289). All participants gave written informed consent before participating in the study.

3.1.1 Image Acquisition

The PET/MR images in the lymphoma study were acquired by the use of a single intravenously injection of 18F-FDG. The hybrid PET/MRI system Siemens Biograph mMR was used for the simultaneous PET and MRI acquisitions. Moreover, the PET/MR images were acquired for a median of 100 minutes where the range differed between $87 - 150$ min depending on the patient's size, and the images were obtained after the radiopharmaceutical had been injected. The lymphoma data set consisted of a variety of MR images such as Coronal Dixon-Vibe, transversal DWI with b-values: 50 s/mm^2 and 800 s/mm^2, transversal T2-HASTE, and coronal T2-TIRM.

The PET image reconstruction was achieved with an iterative reconstruction algorithm, more precisely with the 3D Ordered Subset Expectation Maximization (OSEM) algorithm consisting of 3 iterations, 21 subsets and 4 mm Gaussian filter with point spread function, decay and scatter-correction. However, the Time-Of-Flight was not available for the hybrid system. Lastly, the attenuation correction in the PET/MR images were compassed with the Dixon-Vibe sequence.

A noteworthy remark about the data-set is that the lymphoma study cohort includes PET/CT examinations in addition to the PET/MRI. However, since it is only the PET/MR images that are used in this thesis, the PET/CT is not described any further. For additional information concerning the PET/CT image acquisition and the whole data-set, please see Appendix A.

3.1.2 PET/MRI Data

The lymphoma study from St. Olavs Hospital consists of a total of 108 (61 baseline, 13 interim, and 34 end-of-treatment) PET/MRI examinations. All of the 61 patients participating in the study were scanned for a PET/MRI at baseline. Thereafter, the interim scanning was attained for cHL patients after 2 cycles of chemotherapy. The end-of-treatment images were obtained for both cHL and DLBCL patients after 3-6 weeks following the last cycle of chemotherapy. Therefore, the number of images pertained in the study for the 61 patients varies between 1, 2, and 3.

The clinical reading of the images were done by two pairs of radiologists and nuclear physicians using the same standardized protocols. The radiologist read the MR images, while the nuclear medicine physician interpreted the PET images. The reading of the images were done separately, and thereafter, the two teams provided a joint report for the PET/MRI examinations.

However, the segmentation ground truth (the ideal segmented image) was missing from the dataset and it is crucial for implementing the AI based model for automated segmentation. The required manual segmentation were performed by the author herself and were thereafter validated by Håkon Johansen, a specialist in nuclear medicine and chief attending physician at the Department of Nuclear Medicine and Medical Physics at St. Olavs Hospital.

3.1.3 HUNT Cloud

For this thesis, HUNT Cloud was used. Hunt Cloud is a Cloud computing service with Linux-based virtual machines. It is affiliated to the HUNT Research Centre, Department of Public Health and Nursing, Faculty of Medicine and Health Sciences, and Norwegian University of Science and Technology (NTNU) for data storage and data processing. The Lymphoma dataset has been merely uploaded on and been accessible through the Cloud. The dataset was stored in the XNAT database on HUNT Cloud [65] which provides a secure infrastructure for sensitive data. The anonymized lymphoma dataset was retrieved from the XNAT database on the virtual machine. What is more is that the used programs were run on a

HUNT Cloud GPU virtual machine which has a sufficient amount of memory storage and processing power for this large-scale analysis. This specifically concerns the NVIDIA Tesla P100 GPU computing machine with 16GB HBM2 (High Bandwidth Memory) [66]. The HUNT Cloud service was used for the implementations and computations of the automated segmentation method in this thesis.

3.2 Manual Segmentation

Segmentation of images refers to the procedure that delineate regions such as anatomical structures, lesions, or other various space objects found in the studied image. Segmentation, which is often also denoted as contouring or annotation, is a common procedure in medical image computing for further visualization of structures and quantifications in order to measure different volumes or surfaces. Additionally, segmentations are used for 3D printing and masking, which enables restricted processing and analysis of a specific region [67]. Depending on the software, the annotations can either be performed manually, semi-, or fully- automatic. When preparing manual segmentations it is possible to iterate through all slices in an image and draw contours around the boundaries of the desired target.

3.2.1 Software

Today, there are several open-source image analysis software which provides segmentation tools that can be used to manually segment cancer lesions. 3D Slicer is a free, open source, and multi-platform software package [68]. The software is commonly used for medical, biomedical, and related imaging research. The desktop software is designed to solve advanced image computing challenges and at the same time provide useful clinical and biomedical applications [69].

3D Slicer includes numerous modules and extension packages and additionally supports all types of datasets. To be more precise, data such as segmentations, surfaces, DICOM, NiFTI, transformations, etc. can be viewed in 2D, 3D, and 4D in this software. It is possible to visualize the images both on desktop and in virtual reality. A few analytical tools found in 3D Slicer includes segmentation, registration, and various quantifications.

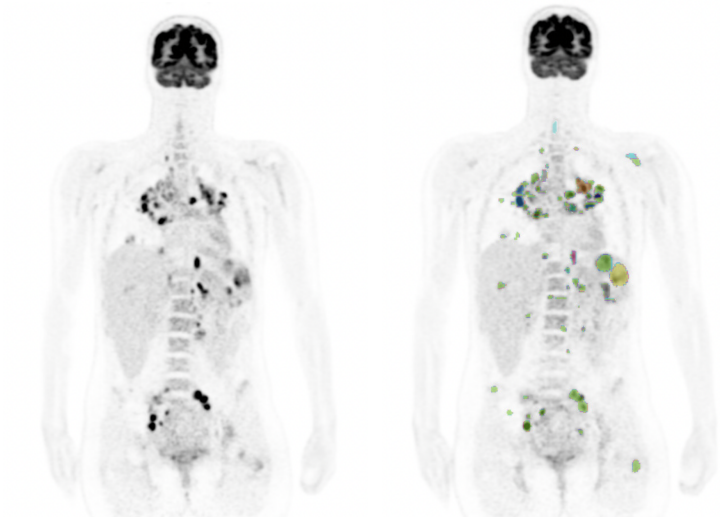
3.2.2 Lymph Node Segmentation

In total, 50 patients from the lymphoma study cohort were manually segmented by the author and thereafter validated by the nuclear physician at St. Olavs Hospital. Previously, in the specialization project, 30 patients were manually segmented, however, this quantity did not provide great results. It was therefore segmented 20 additional patients for the thesis. Since the main purpose is to develop a method to automatically detect cancer lesions, the model is not trained to recognize baseline, interim, and end-of-treatment images for one specific patient. To be more precise, the model sees the baseline, interim, and EOT images as three different

and individual patients. Therefore, the total number of segmented images in the ground truth consists of 64 PET/MRI examinations.

More specifically, 32 DLBCL and 18 cHL patients were manually segmented using the software 3D Slicer. The segmentations from the specialization project were performed on the baseline PET/MR images in addition too a few positive interim images from selected patients. However, for the new manually segmented patients, baseline, interim, and EOT were annotated and validated. Generally, the Regions Of Interest (ROIs) were delineated in areas of the body where cancerous lymph nodes were present and where it showed a clear pathological uptake indicated by the FDG on the PET-scan. The normal physiological uptake of FDG, like that in the brain, heart, kidneys, urinary tract, and bladder, were excluded. The Figure 3.1 below shows how the FDG uptake and the manual segmentations are visualized in 3D Slicer.

As a side note, the first five patients that were manually segmented in the specialization project were annotated by using the ITK-SNAP software, however, it was discovered that this software could not handle different spacing between the image slices in the data-set when overlaying the PET and MR images. It was therefore decided to change to 3D Slicer which was able to handle this inconvenience. From this point onwards it was the 3D Slicer software that was used to segment the patients. Apart from this, another benefit with 3D slicer is that the software provides a semi-automatic PET segmentation extension pack, PET indiC, which was of great assistance when performing the manual segmentations. For a detailed description of how to manually segment in ITK-SNAP and 3D Slicer, see Appendix A.



(a) FDG-uptake **(b)** Manually segmented lesions

Figure 3.1: A patient diagnosed with cHL showing the FDG uptake and the manually segmented cancer lesions. In (a) the figure shows both the pathological and physiological uptake of FDG on the PET image. In (b) the manually segmented FDG-avid cancer nodes (colored) of the same patient are shown.

3.3 Image Pre-Processing

3.3.1 Image Normalization

Quality assurance of the input dataset is necessary to improve the model's lesion prediction performance. The pre-processing step of image normalization is preformed due to the importance of achieving the same image intensity for the PET, T2-HASTE, and DWI. The standardization method chosen transforms the images non-linearly such that there is a significant gain in similarity of the resulting normalized images [70].

This method of standardization requires information from the foreground of the images. This was achieved by first identifying the slices where the brain of the patient were present. Thereafter, the air was removed from the images by creating a mask of the head. This mask was then used to find the histogram of the image. Based on examining 64 PET/MRI foreground regions, the author observed mainly one type of histogram among the images; a bimodal histogram. However, for seven patients there was detected a unimodal histogram. In the case of the bimodal histogram, the second mode, which corresponds to the main foreground object (patient) in the image, was chosen as a landmark. In relation to the unimodal histogram, the mode corresponds to the background of the image. Therefore, the shoulder of the hump of the background intensities were chosen as a landmark. Figure 3.2 schematically shows the locations of the landmarks for the bimodal and unimodal histograms where μ represents the second mode and ω the shoulder.

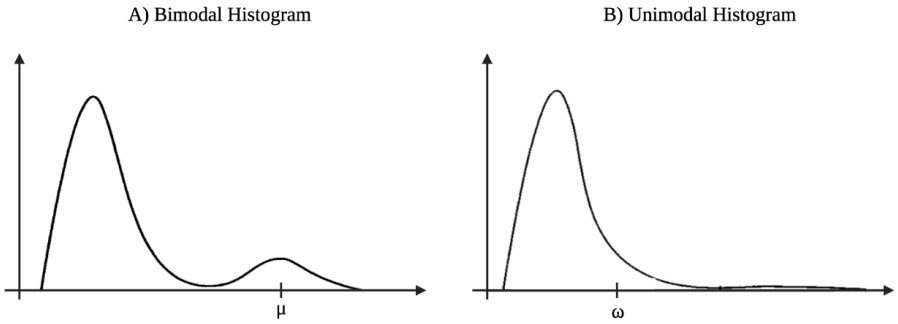


Figure 3.2: The figure shows A) an arbitrary bimodal histogram with landmark μ, where μ is the second mode of the histogram which represents the mode of the foreground of the image. B) shows an arbitrary unimodal histogram where the landmark ω represents the shoulder of the background hump.

After the landmarks were obtained form all the PET, T2-HASTE, and DWI images, the average value of the landmarks for each modality were found and incorporated in the expression:

$$I_{normalized} = \frac{I * 128}{landmark_{avg}}, \tag{3.1}$$

here I is the original image of either PET, T2-HASTE or the DWI, $landmark_{avg}$ is the second mode μ or the shoulder ω of all images in the ground truth for the specific modality chosen to normalize. By multiplying with 128 the average landmark is recurring at the same value for all the images being normalized, i.e., the main foreground for all images is localized at this value. The intensities values in the PET, T2-HASTE, and DWI images were thereafter standardized after one specific patient when creating the 3-channel multi-modal image. Thereafter, all the images were normalized accordingly.

A summarized approach of the normalization method is as follows:

1. Find the head of the patient (the body is removed)
2. Mask out the air. In other words, make a mask of the head to find the foreground of the image
3. Plot the histogram of the image to localize the second mode, i.e. foreground of the image, if bimodal histogram (shoulder if unimodal).
4. Find average of all landmarks
5. Normalize images

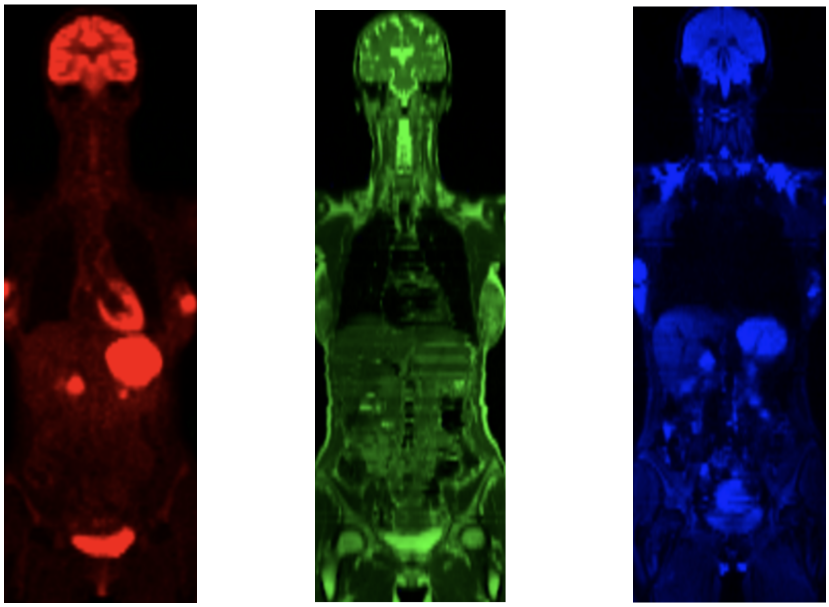
3.3.2 Creating a 3-Channel Multi-Modal Image

When training the model with the segmentation masks from the ground truth, it is necessary for the different imaging modalities to be resampled, i.e., have the same

dimensions. This was not the case for this study, and therefore the PET, T2-HASTE, and DWI with b = 800 s/mm^2 images were resampled into a multi-modal dataset with three channels: PET, T2-HASTE, and DWI. This corresponds to going from gray scale to an RGB image, and was achieved by using the software MRIcroGL and signal processing tools in MATLAB.

Another reason for why such a multi-modal image can be beneficial is due to the fact that the lesions will be more easily visualized in the PET image and more exactly located with the MRI. Therefore, by creating a 3-channel multi-modal image, it can give better functional information combined than separately. Additionally, the RGB image will make it easier for the model to distinguish between pathological and physiological FDG uptake. This is especially beneficial when it comes to the tracer uptake in the brain, heart, kidney, urinary tracts, and bladder as there is often a high concentration of FDG in these organs.

Figure 3.3 below shows which color the specific imaging modality was assigned. The PET image in red, T2-HASTE in green, and the DWI with b = 800 s/mm^2 in blue. In other words, R = PET, G = T2-HASTE, and B = DWI and these colored channels were thereafter combined into the aforementioned RGB image. One example of such an image can be seen in Figure 3.4.

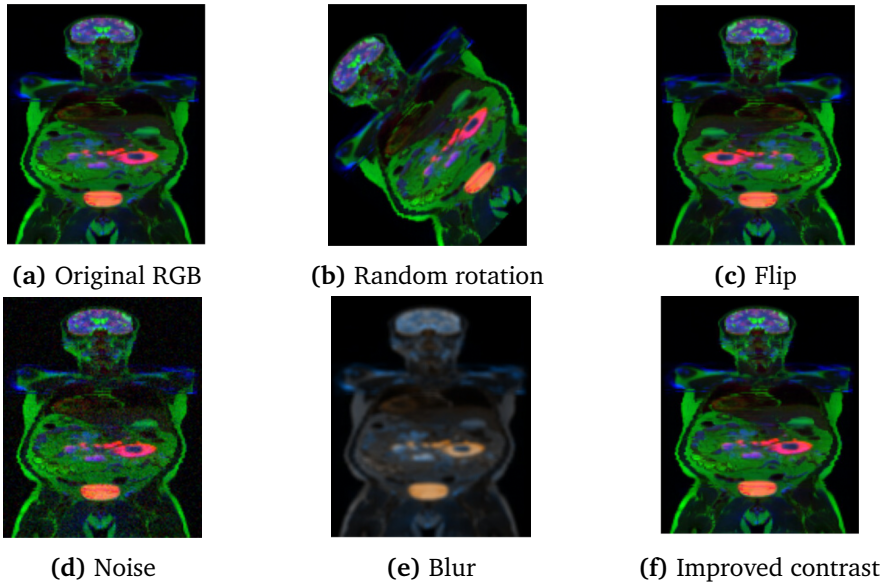


(a) PET **(b)** T2-HASTE **(c)** DWI

Figure 3.3: The figure shows how each imaging modality was assigned a specific colour. (a) Shows PET in red, (b) shows T2-HASTE in green, and (c) shows DWI in blue. These colored images were combined into a multi-modal image with three channels. In other words, an RGB image.

3.4 Data Augmentation

In order to prevent a CNN model from overfitting, it is important to have a large dataset for the network to train on. However, this is not the case for the lymphoma dataset where the ground truth only consists of 64 PET/MRI examinations. Therefore, it was decided to create more data from the existing dataset, i.e., preform data augmentation. This was done by introducing random rotation, flips, and adding noise, blurring, and contrast, and thus increasing the dataset considerably. Figure 3.4 below shows the aforementioned modifications:



(a) Original RGB **(b)** Random rotation **(c)** Flip

(d) Noise **(e)** Blur **(f)** Improved contrast

Figure 3.4: The figure shows the different augmentation methods used on the RGB images. (a) Shows the original RGB image, while (b),(c),(d), (e), and (f) respectively show the random rotation, flip, noise, blur, and improved contrast for the same RGB image.

3.5 Network Architecture

A 2D U-Net was used to train the model, which contained 23 convolutional layers. The network resembles a typical convolutional neural networks as it starts with a repeated application of two 3x3 convolutions which are followed by a rectified linear unit and a 2x2 max pooling operation with stride 2 for downsampling. At each downsampling step, the number of feature channels are doubled. Thereafter the upsampling of the feature map takes place and it is followed by a 2x2 convolution. A concatenation is done and thereafter follows two 3x3 convolutions. In the final layer of the network, a 1x1 convolution is performed. Fig 2.15 schematically shows the architecture of the 2D U-Net.

A ReLU activation functions has been used in every convolution layers except for the output layer. In the output layer, a Sigmoid function has been used to force the prediction output to take values between 0 and 1. The block diagram in Figure 3.5, shows a schematic overview of how each layer is build up in the code. The code and help functions implemented for the training of the network can be found in Appendix B.

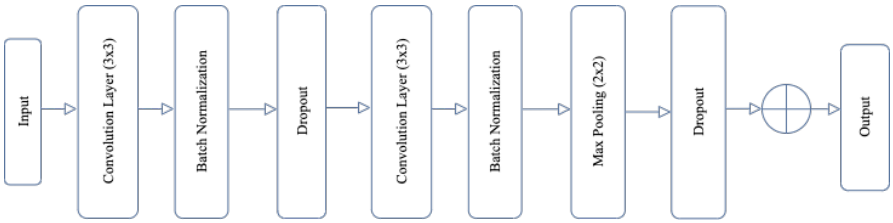


Figure 3.5: The figure depicts the architecture of one layer in the 2D U-Net model.

3.6 Training of the Model

Figure 3.6 shows the necessary steps to execute the training and testing of the 2D U-Net model. First, the lymphoma dataset has to be divided into a training data and testing data set. Secondly, the training data has yet again to be divided into a validation and training data set, which is separated in an approximately 85/15 ratio respectively. Usually, the training and validation is performed by introducing a k-fold cross-validation. As a rule, the training data is used to train the 2D U-Net model while the validation data is used to tune and evaluate the model's performance. After completing the training of the model, the testing data (i.e., unseen patient cases), is given to the model for a final performance validation.

3.6.1 k-Fold Cross-Validation

A cross-validation approach is a resampling procedure which is used to evaluate a deep learning model with a limited dataset. This method has a single parameter denoted with k which refers to the number of groups the data set is to be split into [71], hence the name k-fold cross-validation. More specifically, when the number of splitting folds (groups) k has been determined, it may be used in place of k when the method is referred to, e.g. a $k = 5$ will be denoted as a 5-fold cross-validation.

Primarily, the cross-validation is applied in machine learning in order to estimate the skill of the model when evaluating it on unseen data. That is to say, limited data samples are used to estimate how the model is expected to perform when making predictions on a testing set which has not been used during the training of the model [71]. Cross-validation has become a popular method due to its simplicity and due to the results being less biased, i.e., less optimistic estimates of the model's skill, as compared to other methods such as the simple train/test split [71].

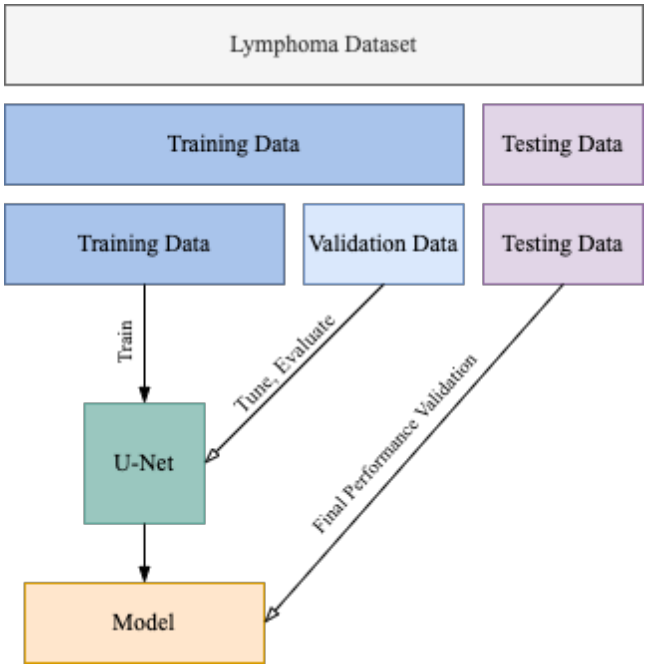


Figure 3.6: The figure depicts the road map of how the training, validation, and testing data are used for training, tuning, and performance evaluation of the 2D U-Net model.

A general procedure of a k-fold cross-validation is as follows [71]:

1. Random shuffle of the dataset
2. Split the dataset into k folds
3. For each unique fold:
    i. Take the fold as either test data or as validation
    ii. The remaining folds are used for training
    iii. Train the model on the training set and validate on the testing set
    iv. Retrain on the evaluation score and thereafter discard the model
4. Summarize the learned skills of the model using the sample of model evaluation scores

An important comment about this procedure is that each patient in the dataset is assigned to an individual fold. The patient stays in this fold during the whole procedure, i.e., each patient is to be used as validation once and used for training the model k−1 times [71][72].

A further introduction to the 4-fold and 13-fold cross-validation in addition to the Leave-One-Out Cross-Validation (LOOCV), which were the methods implemented and tested for this thesis, can be found below.

4-Fold Cross-Validation

In order to perform a 4-fold cross-validation, the lymphoma dataset was divided into a training and testing set in an approximately 85/15 ratio. The testing set consisted of 11 patients, whereas the remaining 53 patients of the ground truth were included in the training set. Nevertheless, a training set of 53 patients is still considered small. For this reason, data augmentation was introduced in order to increase the training data. The augmentation was only performed for the training data set, resultantly increasing the number of patients for training to 312. For each fold, six PET/MRI examinations were randomly excluded to equally divide the the number of patients into four subsets.

Figure 3.7 below shows that the 4-fold cross-validation approach involves randomly dividing the set of data from the lymphoma dataset into 4-folds, which are of approximately equal sizes. The first fold is treated as a validation set, whereas the model is trained (fit) on the remaining three training folds. The 4-fold cross-validation method divides the lymphoma dataset into 4 folds and for each fold again divides the dataset into one validation and three training sets. To be more precise, the training data was divided in a 75/25 ratio for the training and validation phase. Thus, leaving 234 patients for training and 78 PET/MRI examinations for validation in each fold.

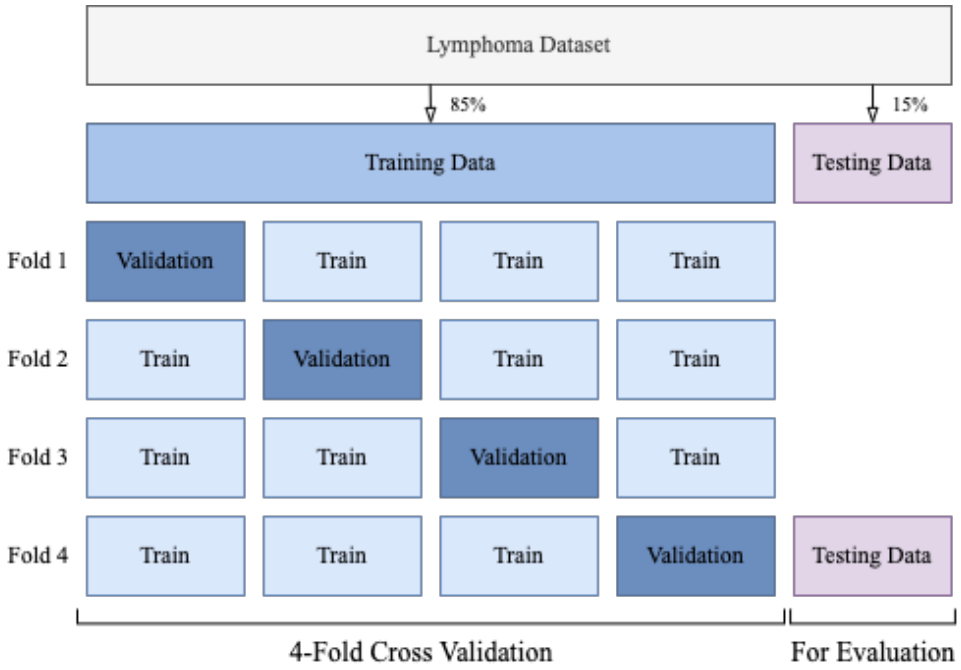


Figure 3.7: The figure depicts the 4-fold cross validation of how the training data is organized and trained accordingly within each fold.

13-Fold Cross-Validation

Initially, as the 4-fold cross-validation, the 13-fold cross-validation also needs the lymphoma dataset to be divided into a training and testing set in an approximately 85/15 ratio. Evidently, the testing set still consisted of 11 patients and the remaining 53 patients were used as training data. Naturally, data augmentation was once more introduced after the splitting in order to increased the patient count to 312. Once again, six PET/MRI examinations were randomly excluded to equally divide the the number of patients into thirteen subsets.

Figure 3.8 below shows a schematic overview of the 13-fold cross-validation where one subset (24 patients) is used for validation. The remaining twelve folds, consisting of a total number of 288 patients (24 in each group), were used for training. The 13-fold cross-validation is a more computationally expensive training method compared to the 4-fold cross-validation.

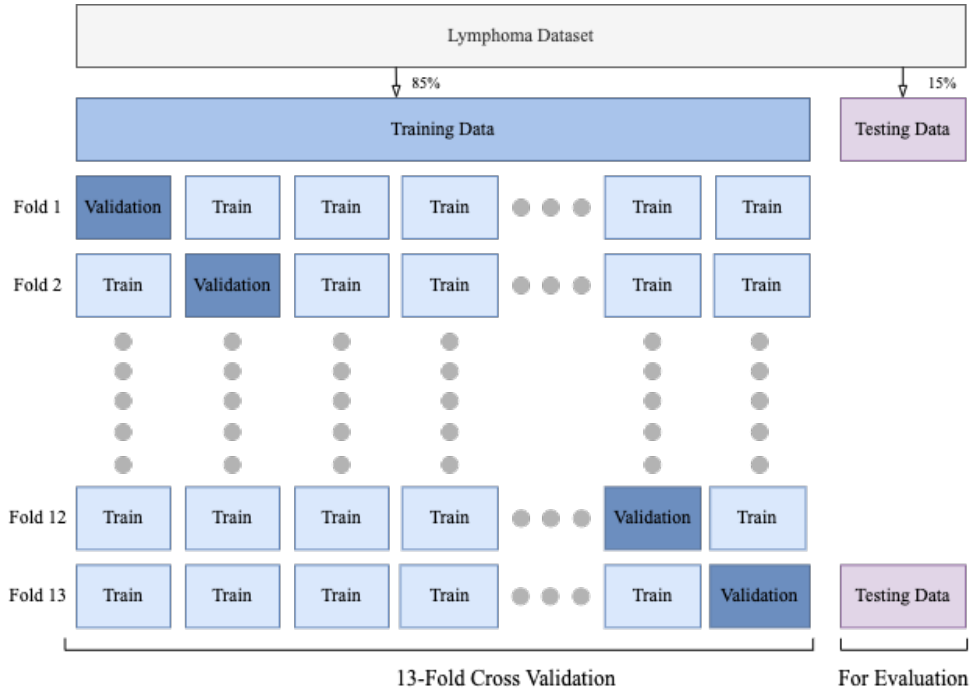


Figure 3.8: The figure depicts the 13-fold cross-validation of how the training data is organized and trained accordingly within each fold.

Leave One Out Cross Validation

Although the Leave-One-Out Cross-Validation (LOOCV) is considered a special case of a k-fold cross-validation with k = N, where N is the total number training data. This method may result in a reliable and unbiased estimate of model performance [73]. In comparison with the 4-fold and 13-fold, the LOOCV is a computationally expensive procedure to perform [73]. Additionally, it does not require to divide the lymphoma dataset in a 85/15 ratio. Figure 3.9 gives a schematic view of the leave-one-out cross-validation approach.

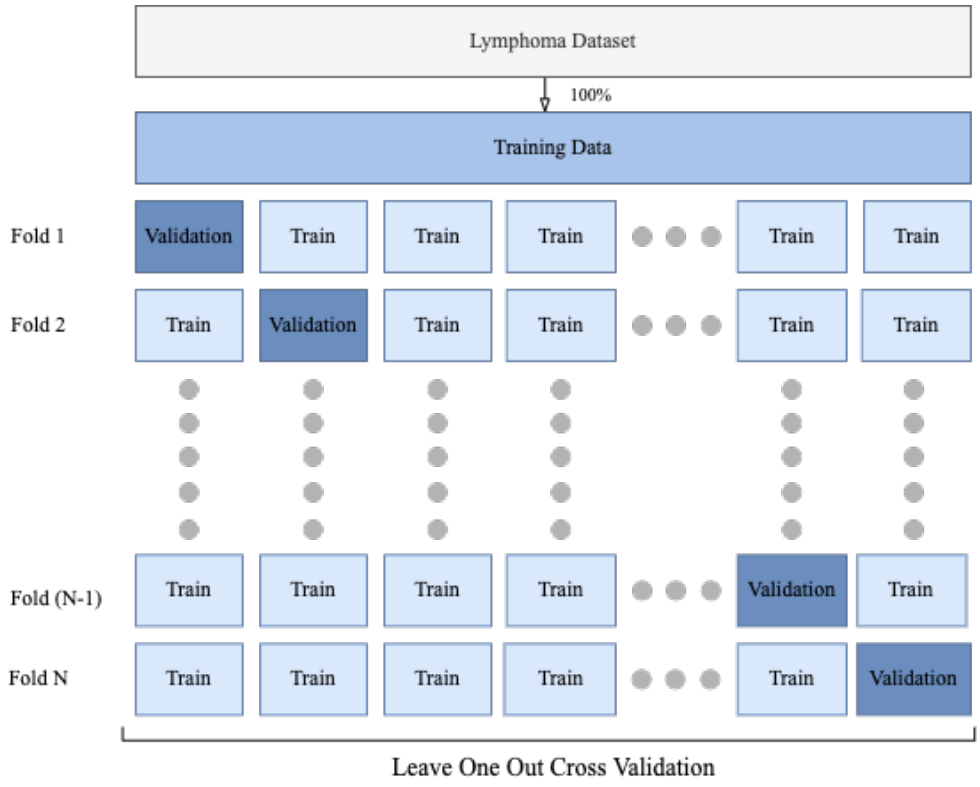


Figure 3.9: The figure depicts the leave-one-out cross-validation of how the training data is organized and trained accordingly within each fold where N represent the total number of data in the training set.

Due to LOOCV may result in improved estimates of the model's performance, the approach is appropriate when an accurate estimate of the model performance is critical [73]. However, for particularly small datasets, LOOCV can lead to model overfitting during training. Consequently, this can result in biased estimates of the model's performance [73].

Chapter 4

Results

In this chapter the relevant findings and results obtained will be presented. The chapter is divided into sections concerning the training and testing of the 2D U-Net model. In addition, a section regarding results of counting the number of lesions detected in the ground truth and the predicted segmentations is included. The sections in this chapter will present the accuracy of the model with both a visual and quantitative analysis. The predicted segmentations in both the validation and testing of the model will be presented and compared for the 4-fold and 13-fold cross-validation methods implemented.

4.1 Training of the 2D U-Net Model

This section will present the results obtained from the training of the 2D U-Net for both the 4-fold and 13-fold cross-validations. Specifically, the qualitative and quantitative results will be presented showing the segmentation accuracy of the 2D U-Net for training and validation. Additionally, the automated segmentation of cancer lesions from the multi-modal images of the same patients will be shown for both cross-validation methods where the same slice from the patients are presented.

Table 4.1 shows the average values for the binary loss and dice scores (DS) attained in the implemented k-fold cross-validations. Overall, the loss is low for all k-folds where a lower value is achieved in the training compared to the validation, which indicates that the model is training correctly. Evidently, the dice score is higher than 0.5, signifying similarity between the ground truth and the automated prediction. Moreover, the dice score is higher in the training set than in the validation, which once more affirms that the model is training correctly. The values from the table below show that the 13-fold achieved the best dice score and lowest loss.

Table 4.1: Average values for both loss and dice scores achieved from the training and validation of the different k-fold cross-validation methods implemented.

Cross-Validation	**Training**		**Validation**	
k-Fold	Loss	Dice Score	Loss	Dice Score
4	0.0061	0.6741	0.0107	0.6105
13	0.0056	0.6948	0.0065	0.6342

4.1.1 4-Fold Cross-Validation

This section will present the quantitative and qualitative results obtained from the training and validation of the 4-fold cross-validation. As stated earlier, 312 augmented patients were utilized for the training of the model.

Quantitative Results

Figure 4.1 shows four different learning curves achieved from the training and validation of the 4-fold cross-validation. The learning curves for binary loss, dice score, accuracy, and precision are presented as functions of epochs where the values are taken as averages of the four folds in the cross-validation. Besides, for a validation of 25% and a training of 75% of the dataset there is a good fit of the model. This is evident as the validation decreases (or increases) to a point of stability. There is also a small generalization gap with the training as shown for each of the learning curves. Additionally, the point in the training where the model performs best is also clearly marked in the plots for the different metrics. The best model varies for all the learning curves, but is often found between five to twenty iterations. Moreover, the curves shows that there is little to gain after the first $2-4$ epochs in terms of general performance. Only the training data values improve, whereas the validation data performance stays more or less constant.

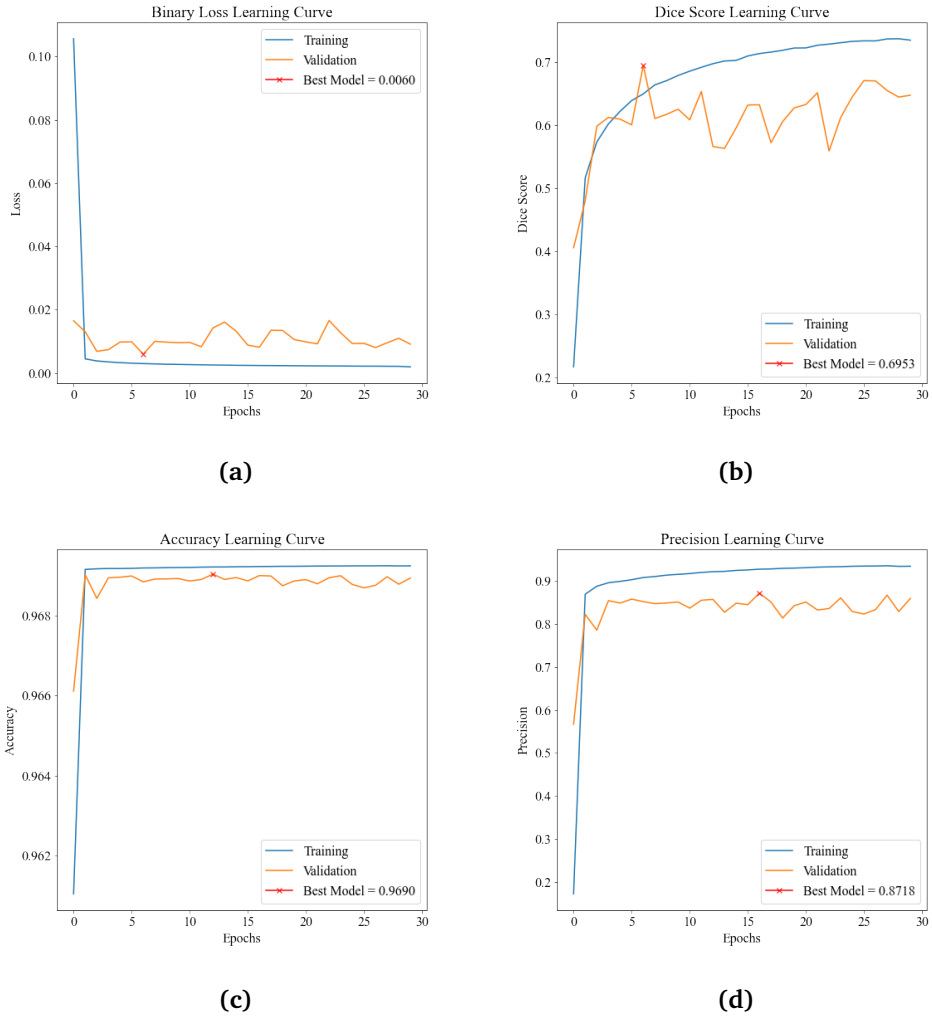
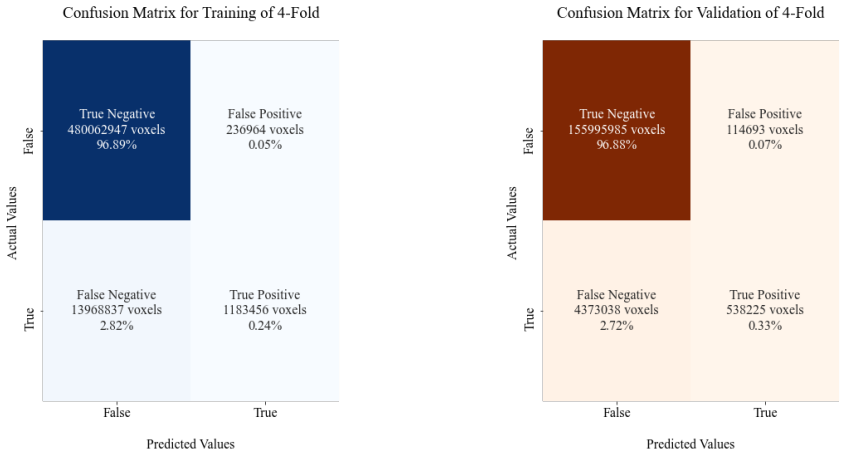


Figure 4.1: The figure shows different learning curves for a) loss, b) dice score, c) accuracy, and d) precision from the 4-fold cross validation during training and validation. In general the metrics are plotted as functions of epochs for both validation (orange) and training (blue) where the best model is depicted with a red x with its corresponding value.

Figure 4.2 gives an overview of the confusion matrices for the training and validation of the 4-fold cross-validation with the percentages and numbers of voxels classified as either True Negative (TN), False Positive (FP), False Negative (FN), and True Positive (TP). Noticeably, the majority of the voxels are classified as true negatives while there are few false positives and false negatives for both the training and validation.



(a) **(b)**

Figure 4.2: The figure shows the different confusion metrics for a) the training and b) the validation. The total number of voxels for all patients during training and validation which is classified as True Positive, True Negative, False Positive, and False Negative are depicted in addition to the percentages of each class.

For the confusion matrix plot in a), the number of True Positive (TP) shows the voxels which are predicted accurately out of the training data, i.e. the 312 PET/MRI examinations. Likewise, the confusion matrix plot in b) depicts the total number of voxels in the validations sets from all four folds that were classified as either TP, FP,TN, or FN.

Table 4.2 below shows the different average values for the evaluation metrics assessed for a validation split of 25% and a training split of 75% during the 4-fold cross validation. In general, the model scores well for the majority of the evaluation metrics. However, the recall (i.e. sensitivity) is low. Moreover, the dice score is adequate but it is a relative low score in comparison with the number of correctly classified voxels from the confusion metrics. This suggests that the DS might not give a great indication of how well the model is actually performing.

Table 4.2: Average values for the evaluation metrics used during training and validation of the 4-fold cross-validation.

Metrics	Training	Validation
Accuracy	0.9713	0.9721
Precision | PPV	0.8953	0.8335
Recall | Sensitivity	0.0781	0.1096
NPV	0.9717	0.9727
Specificity	0.9995	0.9993
DS	0.6741	0.6105

Qualitative Results

Figure 4.3 has been introduced as an example of how the automatic and manually segmentations appear for different patients used in the validation of the 4-fold. The 3-channel multi-modal images (RGB images), ground truths, and predictions in the example are shown in the coronal plane. The predicted segmentations for the different patients in the example appears to be precise when comparing them to the ground truth. Moreover, the predictions are overall placed correctly when looking at the placement of the ground truth and comparing it with the lesions in the patients. Evidently, it is easily noticed that the shape of the predicted lesions are not as defined as the ground truth. Nevertheless, the model's qualitative results shows that it manages to detect lesions with precision.

Additional evaluation results are presented in Table 4.3 for each individual patient from Figure 4.3. The general trend is that the patients score high with respect to specificity, accuracy, and NPV. Whereas the dice score varies for the individual patient, however patient #5 (the last patient in the patient column) has obtain the highest value for the dice score in the example below. Noticeably, the model manages to distinguish between the cancer lesions and physiological uptake in the bowel in this patient. This implies that the model has learned well during training, which again is reflected by the high recall value of 0.8505. Moreover, the model shows an inferior performance for patient #1 as demonstrated by the dice score in comparison to the other patients. The poor performance is also evident by the low scores obtained from the recall.

However, the results of the recall as shown in Tables 4.2 and 4.3 are counter-intuitive. Individually, the patients are scoring over 0.5 for recall, whereas in the training with augmentation it is scoring around 0.1. During training, the model is like a black box and it is impossible to pinpoint where errors appear. However, it is shown later in the result section, and in Appendix A, that the cancer free patients scores low for recall. Therefore, a possible explanation for the low score is the contribution from the augmented healthy patients during training and validation. Likewise, augmented cancer patients that scored low will also contribute to the low score.

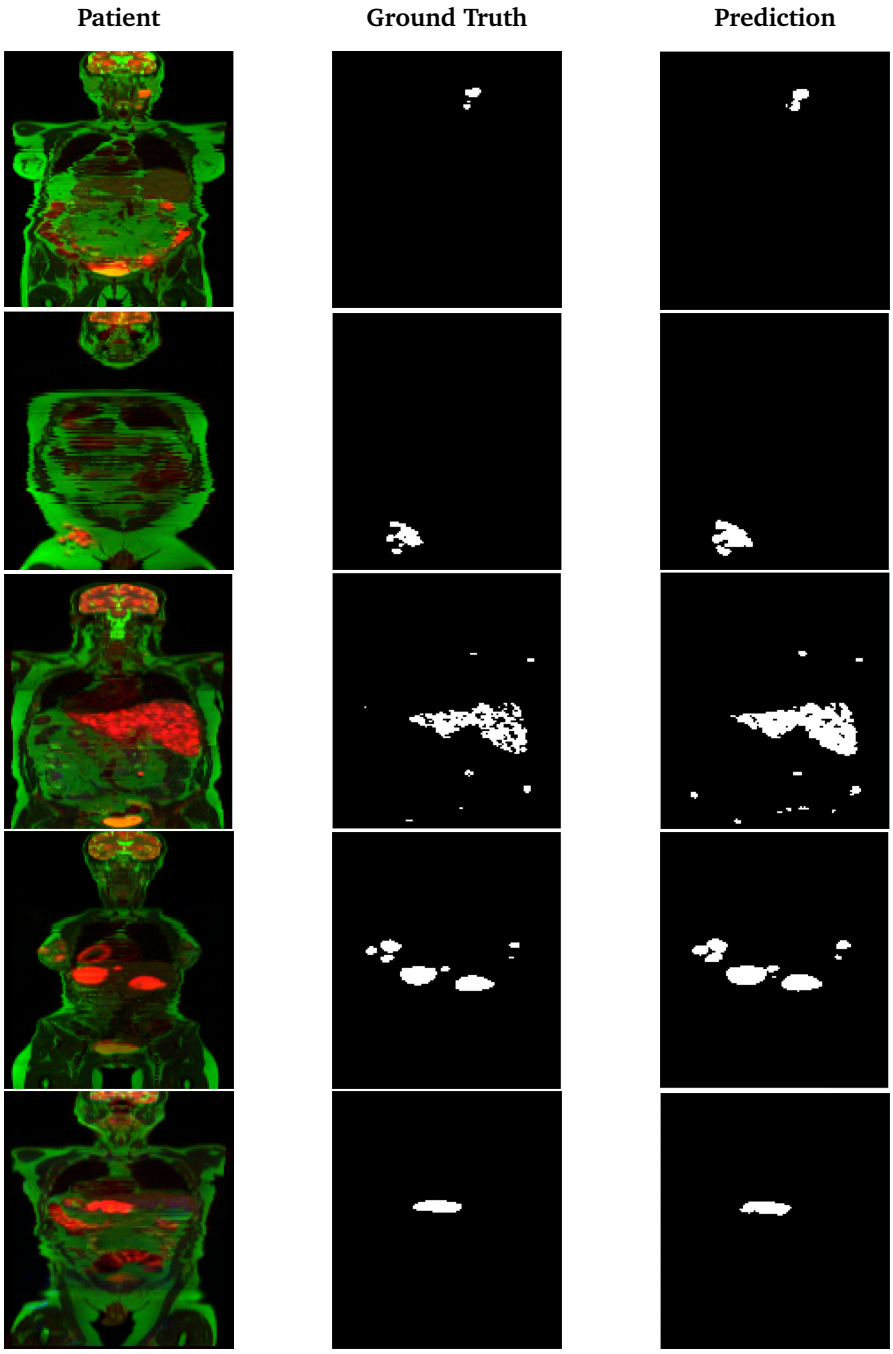


Figure 4.3: Five examples of different RGB images of patients (left column), the segmentation ground truth for these patients (center column), and the predicted segmentation performed by the model for the same patients (right column). The figure shows the results obtained from testing the model on the validation data, i.e., the known patients used in the 4-fold cross-validation.

Table 4.3: Average values for all the evaluation metrics for the patients in Figure 4.3.

Validation 4-Fold							
Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
#1	0.0010	0.4492	0.9995	0.3803	0.6199	0.9994	0.9999
#2	0.0009	0.6661	0.9996	0.5960	0.7549	0.9994	0.9998
#3	0.0143	0.7008	0.9955	0.6878	0.7179	0.9916	0.9961
#4	0.0035	0.7605	0.9990	0.7379	0.7873	0.9983	0.9993
#5	0.0007	0.7937	0.9997	0.7444	0.8508	0.9996	0.9999

4.1.2 13-Fold Cross-Validation

This section will present the quantitative and qualitative results obtained from the 13-fold cross-validation. As mentioned earlier, 312 patients where used for training.

Quantitative Results

Figure 4.4 presents the learning curves for binary loss, dice score, accuracy, and precision for the 13-fold cross-validation method. The 13-fold has an approximately 92% training and 8% validation split of the lymphoma dataset. For the most part there is a good fit of the model as once more the generalization gap between the training and validation is minimal in all the learning curves. Moreover, both the training and validation plot decreases to a point of stability. Noticeably, in the accuracy learning curve, the validation plot is above the training. This signifies that the model is more accurate after having seen the patient data from the training in previous folds and scores as high as 0.9711. Furthermore, the best model for the different learning curves varies between epochs zero to twenty-five and scores high for dice score, accuracy, and precision and low for loss, indicating that the model is training well. By comparing the results from the learning curves of the 4-fold and 13-fold, it is evident that the the 13-fold validation generalizes better than the 4-fold. This is implied as there is a trend in improved performance on the validation set throughout the training and because the generalization gap is smaller.

Figure 4.5 shows the confusion matrices for the training and validation of the 13-fold. The number of voxels and their percentages are represented for the True Negative (TN), False Positive (FP), False Negative (FN), and True Positive (TP). In general, there are fewer FP and FN for the validation data. This is plausible since the PET/MRI examinations in the validation matrix has been seen by the model during training. Nevertheless, as the FP and FN voxels have decreased it suggests that the model is learning well.

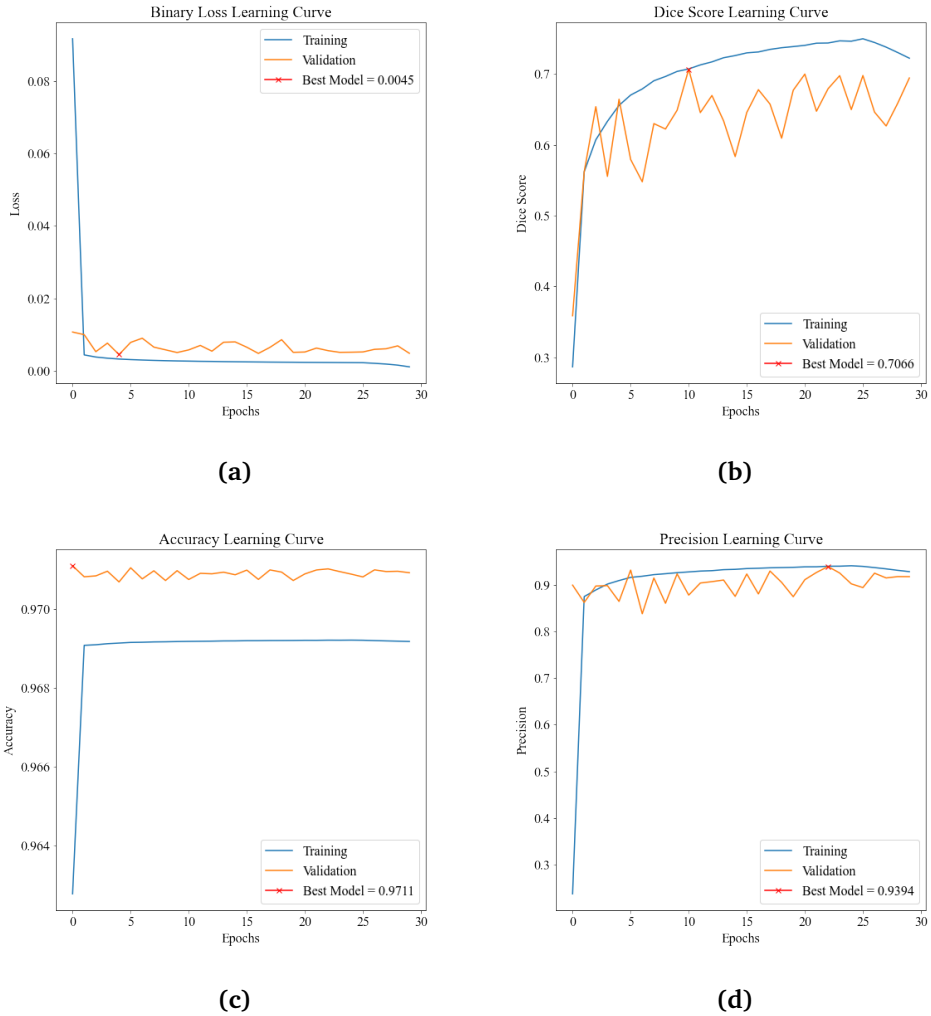
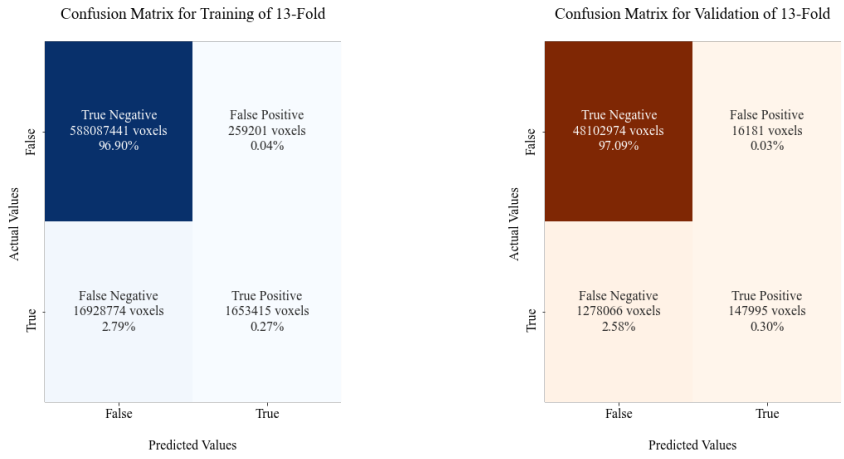


Figure 4.4: The figure shows different learning curves for a) loss, b) dice score, c) accuracy, and d) precision from the 13-fold cross validation during training and validation. In general the metrics are plotted as functions of epochs for both validation (orange) and training (blue) where the best model is depicted with a red x with its corresponding value.

The confusion matrix plots below arrange the number of voxels that are classified as TP, TN, FP, and FN in the training data and validation data. To be more concise, it considers the voxels of the 288 patients for training and 24 patients used in the validation. What can be noticed is that the number of TN voxels are high, over 95% for both training and validation, and this stems from the fact that the majority of the voxels are classified as cancer free. In other words, the cancer lesions constitutes a small part of the patient compared to the healthy tissue, organs and the background.



(a) **(b)**

Figure 4.5: The figure shows the different confusion metrics for a) the training and b) the validation of the 13-fold cross-validation. The total number of voxels for all patients during training and validation which is classified as True Positive, True Negative, False Positive, and False Negative are depicted in addition to the percentages of each class.

Table 4.4 displays the mean values from the different evaluation metrics used during validation and training of the model when implementing the 13-fold cross-validation. In general, the metrics scores high for both training and validation, but it is evident that the accuracy, precision, NPV, and specificity are the superior scores. The recall is considerably low when taking the entire training and validation datasets into consideration. However, the recall does increase in the validation indicating that it is performing better on seen patients during the validation folds.

Table 4.4: Average values for the evaluation metrics used during training and validation of the 13-fold cross-validation.

Metrics	Training	Validation
Accuracy	0.9717	0.9739
Precision | PPV	0.9042	0.9019
Recall | Sensitivity	0.0890	0.1038
NPV	0.9720	0.9741
Specificity	0.9996	0.9997
DS	0.6948	0.6342

Qualitative Results

Figure 4.6 has been included as an example of how the automatic and manual segmentations appear for different patients used in the validation. As previously mentioned, these are the exact same patients slices used for the 4-fold. The figure shows the multi-modal images, the ground truths and the predictions in the coronal plane. Once again, the predicted lesions appears to be placed correctly when comparing it to the ground truth and the patient image itself. Moreover, the lesions are similar to the ground truth, however, the shape of the predicted lesions are still not as thoroughly defined as the ground truth.

Consequently, the automatic segmentation has a tendency to overfit, i.e., making the cancer lesions appear larger. This results in more false positives voxels. The forth patient in the patient column in Figure 4.6 shows a clear over-labeling to the left. In this specific patient example, the ground truth shows three small lesion whereas the prediction segmented one large lesion. Therefore, the general trend shows that the larger the tumor lesion, the better the model manages to predict an accurate outcome. Additionally, patient #5 shows a few false positive lesions which clearly stem from the physiological bowel uptake. Nevertheless, the model manages to predict lesions despite the fact that the overall trend shows that the majority of the smaller lesions appear voxelwise larger than the ground truth.

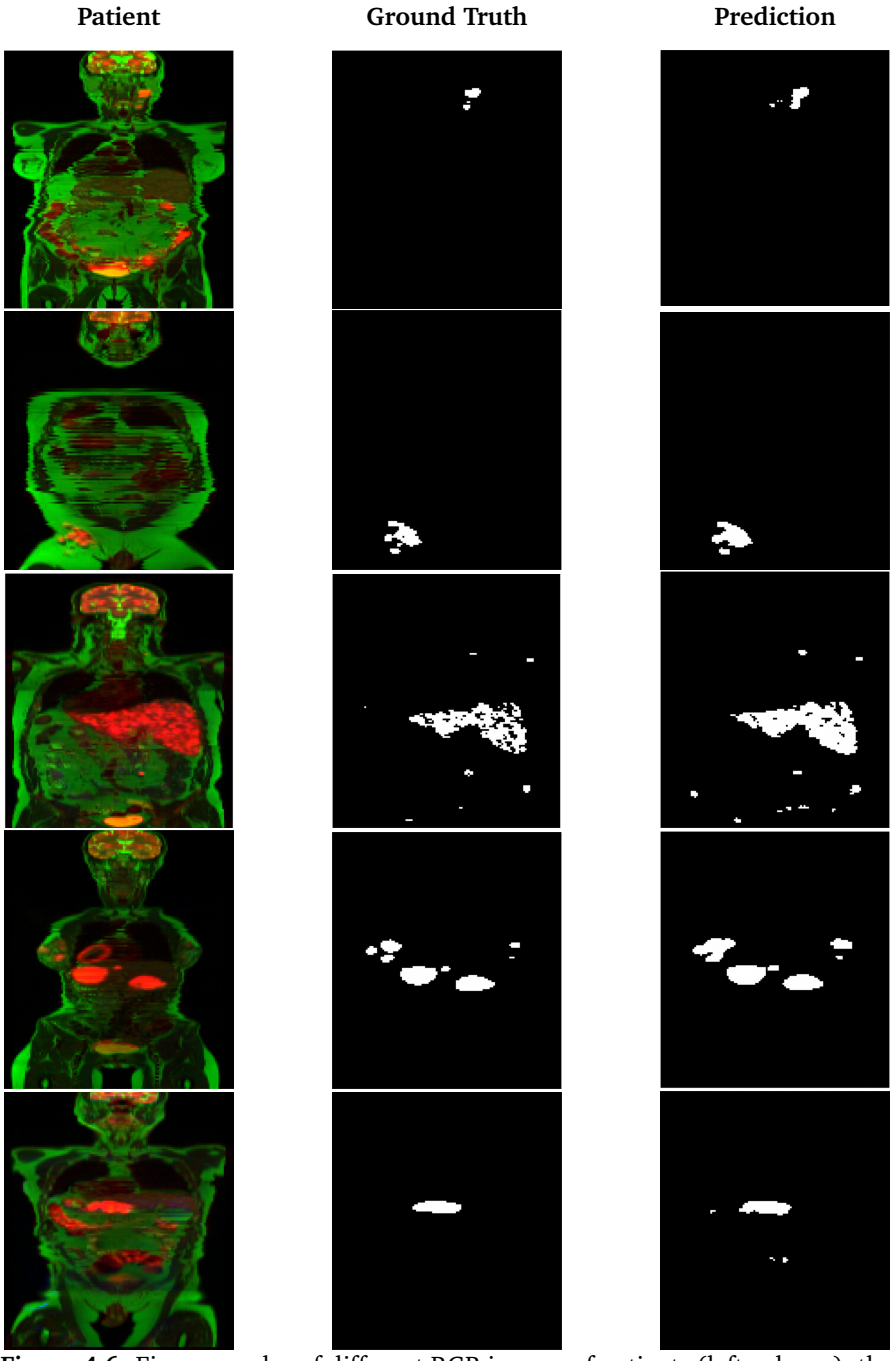


Figure 4.6: Five examples of different RGB images of patients (left column), the segmentation ground truth for these patients (center column), and the predicted segmentation performed by the model for the same patients (right column). The figure shows the results obtained from testing the model on the validation data, i.e. the known patients used in the 13-fold cross-validation.

Table 4.5 presents additional evaluation results for each individual patient from Figure 4.6. The general trend shows that the patients score high with respect to specificity, accuracy, and NPV. Noticeably, the dice score varies for the individual patient, however all patients have dice scores over 0.5. Patient #4 has obtained the highest DS in the example above despite not having clear lesion boundaries. Evidently, the model shows an inferior performance for patient #1 which is once again shown by the dice score in comparison to the other patients. The poor performance is also evident by the low scores obtained from both the precision and recall. Nonetheless, patient #1 has a higher dice score in the 13-fold cross-validation than compared to in the 4-fold, whereas the DS for the other patients varies between the different methods implemented.

Table 4.5: Average values for all the evaluation metrics for the patients in Figure 4.3.

Validation 13-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
#1	0.0007	0.5089	0.9996	0.4557	0.6277	0.9995	0.9999
#2	0.0009	0.6582	0.9997	0.6238	0.7091	0.9994	0.9998
#3	0.0126	0.7101	0.9960	0.7150	0.7084	0.9921	0.9959
#4	0.0031	0.7563	0.9991	0.7566	0.7604	0.9983	0.9992
#5	0.0008	0.7491	0.9996	0.7053	0.8101	0.9995	0.9998

However, once again, the model is showing inferior scores of recall for both training and validation in Table 4.4 compared to the individual patients in Table 4.6. As mentioned earlier, a possible explanation is the contribution from several low score obtained from the augmented healthy- and cancer-patients during training and validation.

4.2 Automated Lesion Segmentation

The following section of the result chapter will present the automated lesion segmentation with the focus on the performance of the trained 2D U-Net model on unseen data, i.e., the testing data. As previously mentioned, the testing set was excluded from the training data in order to serve as a final performance evaluation of the model. This section will present the values obtained from the evaluation metrics. Furthermore, a comparison of the ground truth and the automated segmentation for the testing in both the 4-fold and 13-fold cross-validation is also included. Once again, the automated segmentation of cancer lesions from the multi-modal images will be shown of the same patients for both cross-validation methods.

The box plot in Figure 4.7 shows the mean dice score for training, validation, and testing for both *k*-fold methods implemented. The results indicate that the model is performing inferior on the testing data as compared to both the validation and training. However, it was expected that the model would not perform as

well as in the validation considering the patients have been seen before, but not as low as is suggested from the plot. Apart from this, the average training and validation scores from the plot show that the 13-fold is slightly superior to the 4-fold. Taking all these factors into consideration, the 13-fold cross-validation has a slightly higher overall performance when only comparing the dice scores.

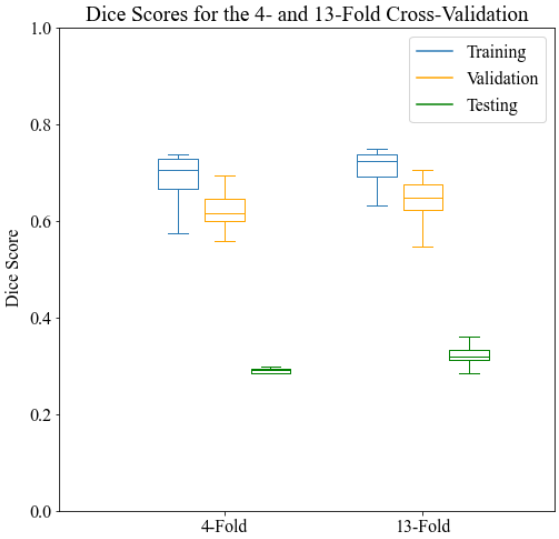


Figure 4.7: Box plot showing average values for the dice scores achieved from the training, validation and testing of the 2D U-Net model for both the 4-fold and 13-fold cross-validation.

4.2.1 Testing of 4-Fold Cross-Validation

This section will focus on the testing of the model for the 4-fold cross-validation where both the quantitative and qualitative results will be shown.

Quantitative Results

Table 4.6 shows the average values for the evaluation metrics which were obtained in the testing set consisting of 11 patients. Like previous results, the accuracy, specificity, and NPV are high. However, the average dice score is low in addition to the precision and recall. The low dice score indicates that the model is not performing as well on the unseen data as it was on the patients in the validation set.

Table 4.6: Average values for the evaluation metrics used during the testing of the 4-fold cross-validation model.

Metrics	**Testing**
Accuracy	0.9955
Precision | PPV	0.2078
Recall | Sensitivity	0.4689
NPV	0.9990
Specificity	0.9965
DS	0.2880

Figure 4.8 shows a selection of patients from the testing set and presents both the ground truth and predicted segmentation of the multi-modal images of these patients. As for previous qualitative results, the predicted segmentation appears to be precise when comparing it to the ground truth. Otherwise, the predicted lesions are voxelwise larger than the ground truth which has been the common factor in the former examples as well. What can be seen from the figure below is that the second patient is cancer free and both the prediction and ground truth is therefore blank. Moreover, in the prediction of patient number four it is evident that the model has missed a lesion all together. In contrast to this patient example, the model has segmented an additional lesion in patient five, giving rise to a false positive lesion.

As stated previously, the model performs better on larger lesions which is also the case for the testing set. This is evident when looking at the first patient in the example. Here the model manages to annotate the larger lesions well, whereas the smaller lesions are often missed or over-labelled. In most cases, the model manages to predict the lesions, however, once more, the overall trend shows that the majority of the smaller lesions are predicted but that they appear voxelwise larger than the ground truth.

Qualitative Results

Furthermore, Figure 4.9 shows the confusion matrices for the patient in the below example. As can be seen, the majority of the voxels has been classified as TN and there is only a small portion of the voxels labelled as TP, FP, and FN which indicates that the lesions only constitute to a small volume in the patient. Another noteworthy comment about these confusion matrices is the relatively high number of classified false negative voxels, which represents voxels that should have been segmented. On the other hand, the number of false positive lesions are small in comparison. Evidently, the cancer free patient from the example below (patient #2), shows zero true positive voxel whereas there are 142 voxels which have been classified as FP meaning that the model has segmented voxels as cancerous. Despite this fact, the main draw back of

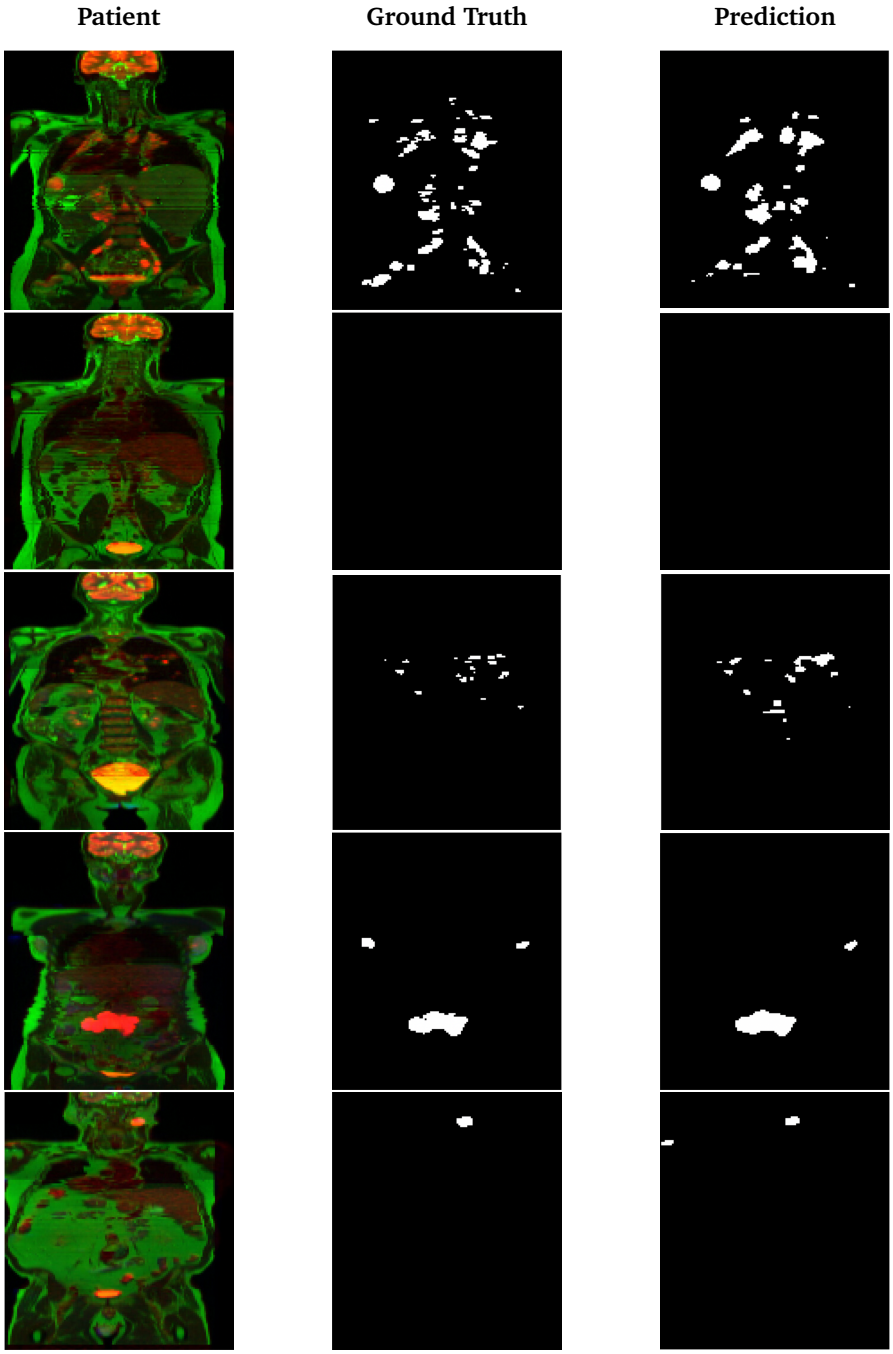


Figure 4.8: Five examples of different RGB images of patients (left column), the segmentation ground truth for these patients (center column), and the predicted segmentation performed by the model for the same patients (right column). The figure shows the results obtained from testing the model on the testing data, i.e., the unseen patients, with the 4-fold trained model.

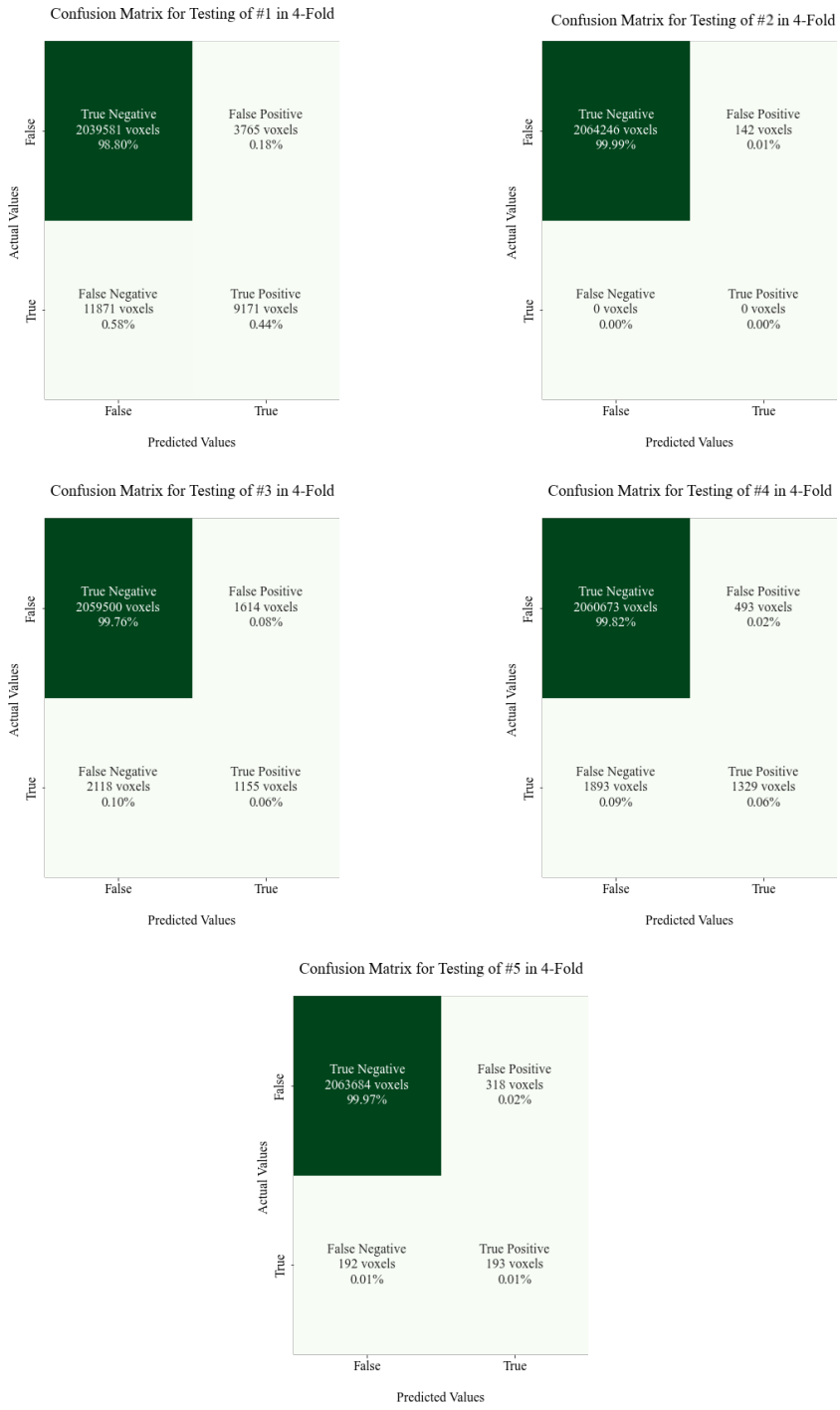


Figure 4.9: The figure shows the confusion matrices for the patients in Figure 4.8 where the TN, FN, FP, and TP voxels are depicted with their corresponding percentages.

this automated method is definitely the number of falsely predicted negative voxels as this contributes to the model's dice score and results in cancer lesions being completely missed, as is evident from patient #5 in Figure 4.8.

Table 4.7 shows the average values computed for the different evaluation metrics for all voxels in the patients in Figure 4.8. It is evident from the results that the accuracy, specificity, and NPV are once again the metrics which are scoring the highest. The dice score is lower than for the patients in the validation set, which was expected. In addition it can be seen that the DS varies for each patient. Noticeably, patient #2, the cancer free patient, has an inferior DS and it scores low. The score itself indicates that there is minimum overlap between the ground truth and the automated segmentation. However, as seen from the confusion matrix for this patient, only 142 voxels are classified as false positives which consequently should result in a high DS. Even though there is only a few voxels falsely predicted, the overlap should have resulted in a higher score, which can indicate that the dice score is not the best metric to evaluate the performance on cancer free patients.

Table 4.7: Average values for all the evaluation metrics used for the patients in Figure 4.6.

Testing 4-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
#1	0.0239	0.5390	0.9982	0.7092	0.4359	0.9924	0.9942
#2	0.0001	0.0086	0.9999	0.0000	0.0000	0.9999	1.0000
#3	0.0055	0.3897	0.9992	0.4515	0.3530	0.9982	0.9990
#4	0.0050	0.4952	0.9998	0.6797	0.4126	0.9988	0.9991
#5	0.0005	0.4246	0.9998	0.3963	0.5006	0.9998	0.9999

From the selected patients from the testing set, it clearly shows that the automated segmentation model performs best for patient #1. Even though this patient has the highest loss, both the dice score and precision is superior to that of the other patients in the example. Nevertheless, as seen from the qualitative results from Figure 4.8, the model manages to segment cancerous lesions from the unseen 3-channel multi-modal images in the testing set.

4.2.2 Testing of 13-Fold Cross-Validation

The following section will focus on the testing of the model for the 13-fold cross-validation, where both the quantitative and qualitative results will be presented.

Quantitative Results

Table 4.4 presents the average values from the obtained evaluation metrics used during the testing of the 13-fold cross-validation. Likewise as the 4-fold, the 13-fold scores high for the accuracy, NPV, and specificity. The mean DS is higher than

in the 4-fold testing, implying that there is a greater overlap between the ground truth and the predicted segmentation. However, the average DS for the entire testing set is still considered a low score. Nonetheless, the 13-fold performs better for all the evaluation metrics than for the 4-fold and suggests that this is the best model.

Table 4.8: Average values for the evaluation metrics used during the testing of the 13-fold cross-validation model.

Metrics	**Testing**
Accuracy	0.9959
Precision | PPV	0.2385
Recall | Sensitivity	0.4845
NPV	0.9990
Specificity	0.9969
DS	0.3183

Qualitative Results

Figure 4.10 shows the results from the testing of the 13-fold for the same patients used in the 4-fold. As can be seen from the figure, the predicted segmentation gives the impression of being placed correctly in accordance with the ground truth. Furthermore, the shapes of the predicted lesions are as mentioned earlier still not as defined as the ground truth where the smaller lesions are often larger then they appear in the ground truth. Evidently, for the second patient (the cancer free patient), the prediction has delineated two small lesions which are clearly not present in the multi-modal image. Moreover, the predicted masks for patient four clearly misses two lesions. This indicates that the 13-fold most likely will have more false negative voxels than the 4-fold. Additionally, the case for patient #5 fails to segment a very distinct cancer lesion from the patient's neck in this specific coronal image slice.

Figure 4.11 presents the confusion matrices for the patients in Figure 4.10 below. Persistently, the TN scores the highest which were to be expected. What is more is that the cancer free patient (number two) has fewer falsely predicted voxels, which imply that the 13-fold is slightly better at segmenting cancer free patients. As noted previously, the FN voxel were high in the 4-fold, and this is still the case for the 13-fold. This explicitly suggests that the model's main limitation is the wrongly classification of voxels which should be segmented. This consequently results in the model missing out on lesions altogether, as shown in the examples below. Notwithstanding patient #2, the 13-fold actually predicts more FP and FN voxels for the other patients compared to the 4-fold.

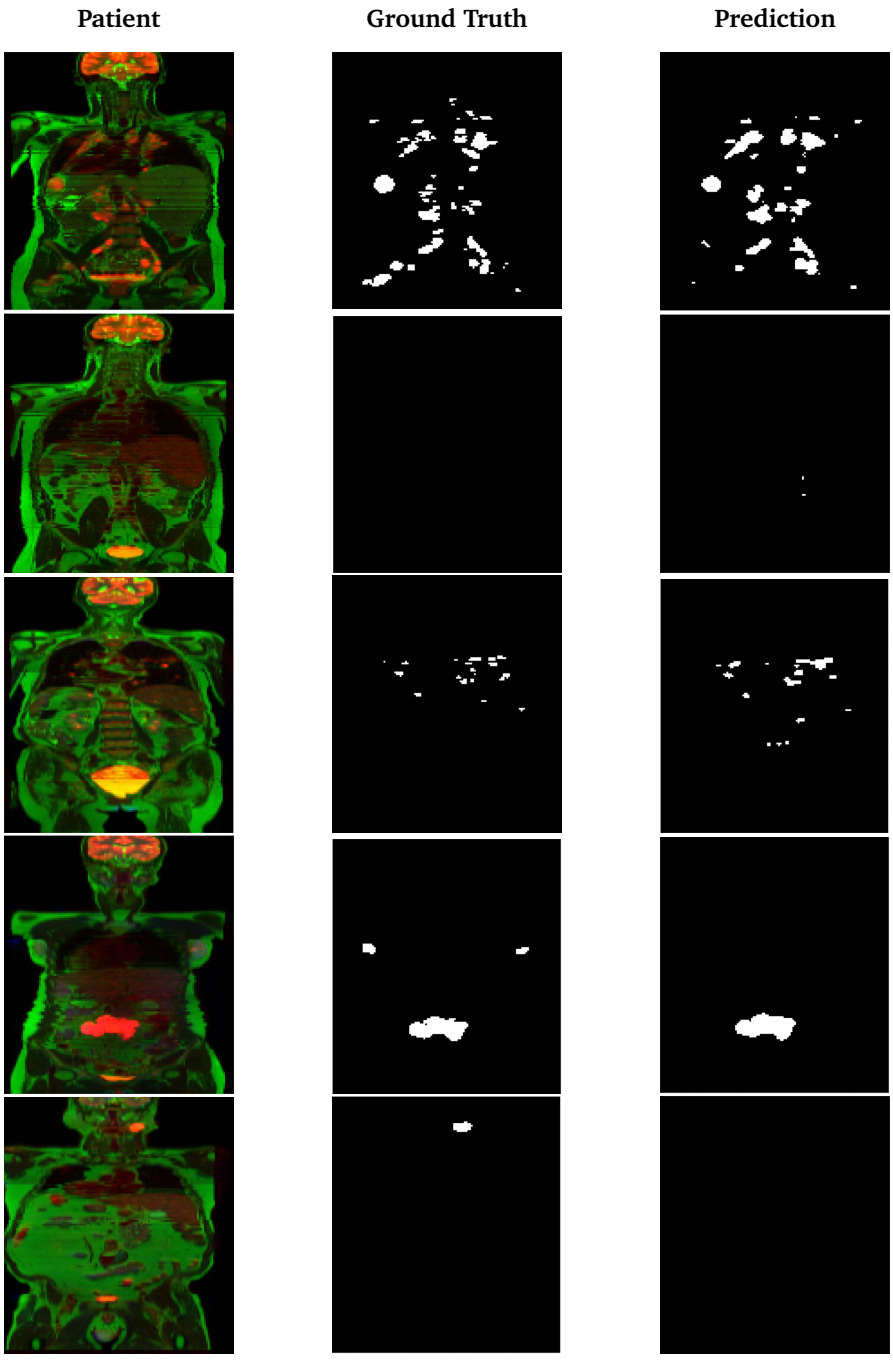


Figure 4.10: Five examples of different RGB images of patients (left column), the segmentation ground truth for these patients (center column), and the predicted segmentation performed by the model for the same patients (right column). The figure shows the results obtained from testing the model on the testing data, i.e., the unseen patients with the 13-fold trained model.

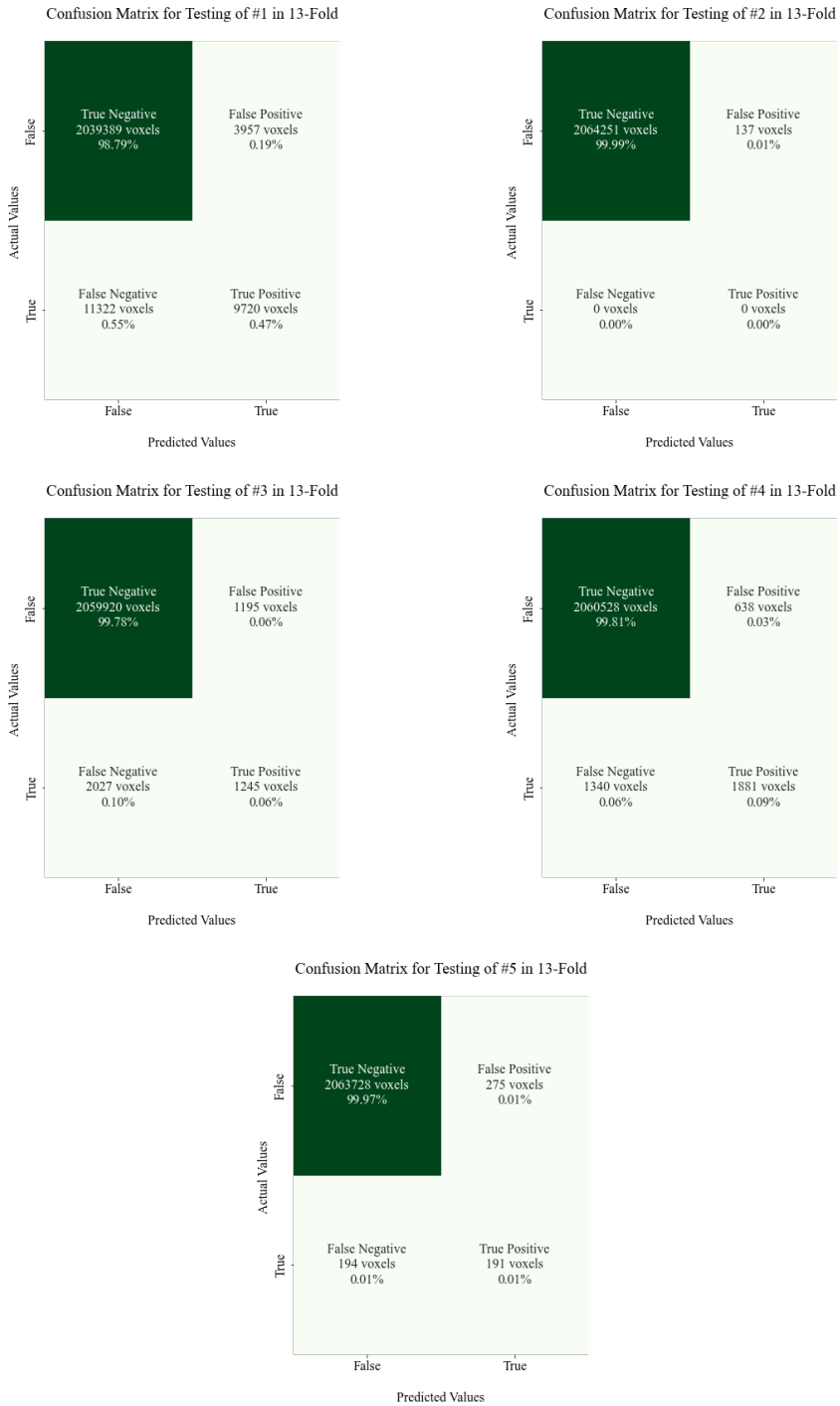


Figure 4.11: The figure shows the confusion matrices for the patients in Figure 4.10 where the TN, FN, FP, and TP voxels are depicted with their corresponding percentages.

Table 4.9 presents the average values for the evaluation metrics tested for the patients in Figure 4.10. As a rule, the model scores high values for accuracy, specificity, and NPV once again, which is expected due to the number of TN voxels from the confusion matrices. The overall DS for the patients are higher for the testing of the 13-fold than for the 4-fold. Patient #4 scores the highest values for DS, precision, and recall despite not segmenting the two lesions from the explicit slice used in the example. It is noteworthy to remember that only one slice of the 112 is shown for each patient, which suggests that the overlap is greater when taking all slices into consideration. The DS is still low for the healthy patient even though the number of falsely predicted voxel are lower for the 13-fold. This again indicates that the dice score might not be the best evaluation metric. Noticeably, the 13-fold performs better for the unseen patients compared to the 4-fold cross-validation.

Table 4.9: Average values for all the evaluation metrics for the patients in Figure 4.10.

Testing 13-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
#1	0.0206	0.5589	0.9981	0.7142	0.4619	0.9926	0.9945
#2	0.0001	0.0173	0.9999	0.0000	0.0000	0.9999	1.0000
#3	0.0046	0.4357	0.9994	0.5199	0.3805	0.9984	0.9990
#4	0.0034	0.6432	0.9997	0.7495	0.5839	0.9990	0.9994
#5	0.0005	0.4462	0.9999	0.4600	0.4956	0.9998	0.9999

4.3 Counting Cancer Lesions

This following section presents the number of lesions detected in the ground truth and the automated segmentation. In order to test the robustness of the 2D U-Net model's ability to segment cancerous lesions, it was decided to perform lesion counting. Lesion counting is typically executed by first binarizing both the predicted segmentation and the ground truth. This is then followed by counting the number of connected components in the binarized images.

The detected lesions were classified as TP if they had at least one voxel overlapping with the ground truth. If the segmentation did not contain any voxels overlapping it would be identified as a FP. Whereas if the automated segmentation did not overlap with any part of the ground truth, it was classified as a FN. Moreover, the TN are not accounted for due to the fact that a true negative finding is not defined per lesion. The metrics used for evaluation were the true detection ratio TPR, i.e., sensitivity, the false negative rate FNR, the PPV, and the false detection ration FDR. The metrics were computed in order to assess the lesion detection performance of both the 4-fold and 13-fold cross-validation methods implemented.

To demonstrate further, two sets of TP values were calculated, namely TP_{AI} and TP_{GT}. The value of TP_{GT} was calculated by iterating through all the connected-components found in the ground truth and comparing those with the AI predicted lesions. If there was one voxel in the lesion overlapping, it was counted as a TP_{GT}. If there was no overlap, the lesions was counted as a FN_{GT}. Likewise, the value of TP_{AI} was calculated by iterating through all the connected components from the model's prediction and comparing those with the lesions in GT. Where there no overlap, it was counted as a FP_{AI}. However, if there they were overlapping, it was counted as a TP_{AI}. It was not expected that the model would segment the exact same number of lesions as the ground truth. Therefore, by iterating through the two different masks of connected components, one assures that all segmented lesions are classified as either true positives, false negatives, or false positives. For the code implementation of counting cancer lesions, please see Appendix B.

4.3.1 4-Fold Cross-Validation

Table 4.10 shows the quantitative lesion analysis of the five patients in the validation found in Figure 4.3. The results show a high true positive rate, a low false negative rate, a high positive predictive value, and a low false discovery rate. These evaluation methods illustrate the usefulness of the 4-fold model for segmenting cancer lesions from the different 3-channel multi-modal images. The number of false positive and false negative lesions varies for each patient. Whereas the high number of false negative lesions for patient #3 imply that the model is more prone to missing lesion than segmenting false lesions for this specific example. The general trend show that the estimated number of predicted lesions (AI) are higher than the actual number of lesions (GT) found in the ground truth.

Table 4.10: The table shows the number of counted lesions from the validation patients in Figure 4.3 for both the ground truth and the automated lesion masks predicted by the model. The estimated number from the predicted lesions segmented by the model is denoted with AI, while the actual number of lesions from the ground truth is GT. The values for TP_{AI}, FP_{AI}, TP_{GT}, and FN_{GT} are shown. Additionally, the true discovery rate (TPR), i.e. sensitivity, the false negative rate (FNR), the positive prediction value (PPV) i.e., precision, and the false discovery rate (FDR), where FNR = FN_{GT}/GT and FDR = FP_{AI}/AI.

Counting Lesions in Patients from Validation of 4-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
#1	6	4	1	5	0	4	1.0	0.0	0.83	0.17
#2	4	3	1	3	0	3	1.0	0.0	0.75	0.25
#3	55	53	5	50	11	42	0.79	0.21	0.91	0.090
#4	9	13	0	9	3	10	0.77	0.23	1.0	0.0
#5	1	1	0	1	0	1	1.0	0.0	1.0	0.0

Table 4.11 shows the number of lesions counted in the ground truth (GT) and the prediction from the model (AI) with respect to the patients from the testing

of the 4-fold found in Figure 4.8. Again, there are several more false negative than false positive lesions with the exception of patient #2 which is cancer free[1]. For this patient, the model falsely predicts 61 lesions. Moreover, the values for TPR, FNR, PPV, and FDR varies for each patient. Nevertheless, there is no general trend with regard to the estimates as the number of lesions predicted by the model depends on the patient. This is evident from the table since the number of lesions is both higher and lower than what is found in the ground truth.

Table 4.11: The table shows the number of counted lesions from the testing patients in Figure 4.8 for both the ground truth and the automated lesion masks predicted by the model. The estimated number from the predicted lesions segmented by the model is denoted with AI, while the actual number of lesions from the ground truth is GT. The values for TP_{AI}, FP_{AI}, TP_{GT}, and FN_{GT} are shown. Additionally, the true discovery rate (TPR), i.e., sensitivity, the false negative rate (FNR), the positive prediction value (PPV) i.e., precision, and the false discovery rate (FDR), where FNR $= FN_{GT}/GT$ and FDR $= FP_{AI}/AI$.

Counting Lesions in Patients from Testing of 4-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
#1	53	63	6	47	28	35	0.56	0.44	0.89	0.11
#2	61	0	61	0	0	0	—	—	0.0	1.0
#3	70	69	27	43	27	42	0.61	0.39	0.61	0.39
#4	5	8	0	5	4	4	0.50	0.50	1.0	0.0
#5	7	2	6	1	1	1	0.50	0.50	0.14	0.86

4.3.2 13-Fold Cross-Validation

The number of counted lesions from both the ground truth and the predicted segmentation from the 13-fold cross-validation can be found in Table 4.12. The lesion are counted from the patients in Figure 4.6 and the results show an overall high TPR and PPV while the FNR and FDR are low. Generally, the estimated lesions detected have a small proportion of false positive lesions. It is shown that patient #3 has the highest number of undetected lesions. For the 13-fold, the number of estimated lesions and actual numbers of lesions varies. However, the general trend once more shows that the counted lesions from the model's prediction is often found to be higher than the lesions counted in the ground truth.

[1]From the definitions of TPR (2.13) and FNR (2.14), the cancer free patients will result in a $\frac{0}{0}$ expression. The evaluation metrics are therefore not relevant to consider when the ground truth is blank. TPR and FNR is consequently marked with — in the tables.

Table 4.12: The table shows the number of counted lesions from the validation patients in Figure 4.6 for both the ground truth and the automated lesion masks predicted by the model. The estimated number from the predicted lesions segmented by the model is denoted with AI, while the actual number of lesions from the ground truth is GT. The values for TP_{AI}, FP_{AI}, TP_{GT}, and FN_{GT} are shown. Additionally, the true discovery rate (TPR), i.e., sensitivity, the false negative rate (FNR), the positive prediction value (PPV) i.e., precision, and the false discovery rate (FDR), where FNR = FN_{GT}/GT and FDR = FP_{AI}/AI.

Counting Lesions in Patients from Validation of 13-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
#1	10	4	5	5	1	3	0.75	0.25	0.50	0.50
#2	3	3	0	3	0	3	1.0	0.0	1.0	0.0
#3	56	53	6	50	10	43	0.81	0.19	0.89	0.11
#4	15	13	2	13	1	12	0.92	0.080	0.87	0.13
#5	1	1	0	1	0	1	1.0	0.0	1.0	0.0

Table 4.13 presents the results obtained from counting the lesions for the patients from the testing example in Figure 4.10. There are varying TPR, FNR, PPV, and FDR for these patients. Moreover, there is a higher number of false negative lesions detected than compared to the false positive lesions with the exception of the cancer free patient[1] (#2). Additionally, this cancer free patient has a significantly lower number of segmented lesion than compared with the cancer free patient in the testing of the 4-fold. This implies that the 13-fold is better equipped at delineating cancer free patients.

Table 4.13: The table shows the number of counted lesions from the testing patients in Figure 4.10 for both the ground truth and the automated lesion masks predicted by the model. The estimated number from the predicted lesions segmented by the model is denoted with AI, while the actual number of lesions from the ground truth is GT. The values for TP_{AI}, FP_{AI}, TP_{GT}, and FN_{GT} are shown. Additionally, the true discovery rate (TPR), i.e., sensitivity, the false negative rate (FNR), the positive prediction value (PPV) i.e., precision, and the false discovery rate (FDR), where FNR = FN_{GT}/GT and FDR = FP_{AI}/AI.

Counting Lesions in Patients from Testing of 13-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
#1	73	63	12	61	20	43	0.68	0.32	0.84	0.16
#2	7	0	7	0	0	0	—	—	0.0	1.0
#3	59	69	14	45	30	39	0.57	0.43	0.76	0.24
#4	8	8	4	4	4	4	0.50	0.50	0.50	0.50
#5	2	2	1	1	1	1	0.50	0.50	0.50	0.50

Noticeably, the TPR is higher than the values achieved in the testing of the 4-fold. This suggests that the 13-fold model performs better in regard to segmenting the actual lesions. Regardless, the number of counted predicted lesions is still

varying with both higher and lower values than for the lesions found in GT. Nevertheless, the model has managed to predict the same number of lesions for two patients, but only half of the them are deemed as true positive lesions.

Additional results for the lesion counting for the other 53 patients in the validation and 11 patients from the testing can be found under Additional Results in Appendix A.

Chapter 5

Discussion

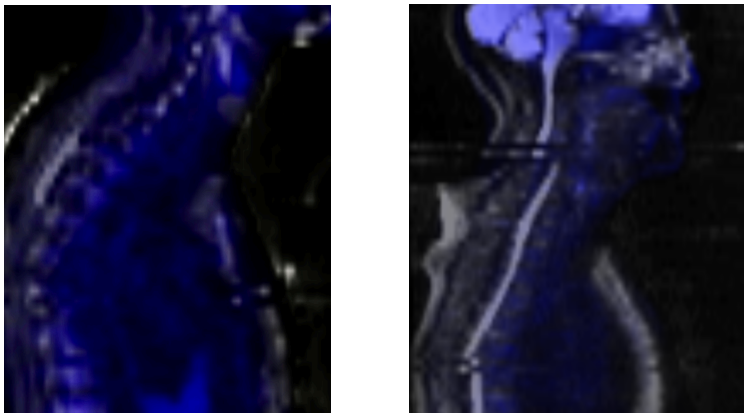
In this chapter the results and findings will be discussed in addition to some discoveries and comments regarding the image pre-processing step. The sections are divided into pre-processing of data, training of the 2D U-Net model, automated lesion segmentation, and counting cancer lesions. Lastly, further work and improvements will be proposed to surpass the results acquired in this thesis.

5.1 Pre-Processing of Data

As previously mentioned, the 3-channel multi-modal images consists of two MR images, namely the T2-HASTE and DWI with a b-value of 800 s/mm^2. One of the reasons why the diffusion weighted image with the highest b-value was chosen was to get information provided from the low signals in places of high diffusion. Since the diffusion of water molecules follows a pattern according to tissue structure and properties, the measured diffusion in an image that deviates from the expected diffusion values might be an indication of pathology [74]. This is especially common for cancer tumors. Thus, the DWI can contribute to detect lymphoma cancer lesions. Moreover, the DWI with the b-value of 50 s/mm^2 was not chosen because it was deemed that the T2-HASTE would contribute to better anatomical and morphological information. This is especially crucial when training the model to distinguish between pathological and physiological uptake of FDG.

Subsequently, the variation in co-registration of the different imaging modalities can affect the performance of the model. The alignment of PET, T2-HASTE, and DWI images did not always prove to be optimal when overlaying the data. Creating the RGB image in the pre-processing step required the dimensions of the images to be equal. However, since this was not the case for the original NiFTI images, the volumes were resampled to the same dimension as the DWI. The DWI was used due to it having the smallest dimension. This resulted in losing slice information from the the T2-HASTE and PET as they had to be compressed to a smaller size. As a consequence, the co-registration for some of the patients did not always prove to be optimal. This was prominent when looking at the spine, which was not always aligned properly. Figure 5.1 below shows an example of two

different patients where one spine is aligned correctly while the other one is not. Additionally, the figure serves as an example of how the diffusion-weighted stations were not stitched sufficiently. The head/neck and torso stations are clearly not aligned correctly as there are two visible gaps in image b). Hence, this stitching can be a contributing factor to the discrepancy between the co-registration of the PET, T2-HASTE, and DWI. Another possible explanation for the misregistration is that it could originate from patient motion during the PET/MRI examination.



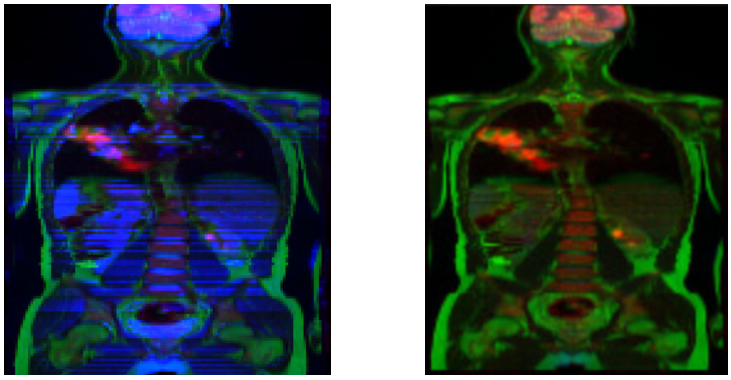
(a) **(b)**

Figure 5.1: The figures depicts how the co-registration is not optimal for every patient, especially when studying the superimposing of PET with the DWI with regard to the spine. Here the blue color represents the PET image overlaid on a gray DWI. Image a) shows a spine that is aligned correctly whereas image b) shows a patient where the spine is not aligned correctly.

Despite the deviation of co-registration between the different modalities, combining PET and MRI will result in better functional information. This is due to the cancer lesions being more easily visualized with PET and more exactly located with MRI. As a result, a better delineation of the tumor lesions can be achieved by the model. Additionally, the MR images will provide physiological information that can provide useful details when the model needs to distinguish between pathological and normal uptake of FDG. Hence, the model will have more information and be able to differentiate between organs that do have normal physiological uptake, like the brain and bladder, and organs and lesions that are affected by cancer. However, if the co-registration is not optimal it will affect the predicted segmentations. Consequently, this results in cases of false positive lesions as the visible organs from the PET, T2-HASTE, and DWI are not properly aligned.

Furthermore, the normalization of the 3-channel multi-modal images were improved in order to standardize the intensities in the different imaging modalities for each channel. The normalization of the input image is important for a deep learning model for the purpose of ensuring that each input parameter, the voxels in this case, have similar data distribution. Figure 5.2 shows the before and after results of the normalization implemented. It is evident that the patient in a)

had a superior contribution from the DWI image which has been rectified in b). Moreover, it was found for a few patients that the DWI image was composed of the two b-values, 50 and 800 s/mm^2. This was corrected for by splitting the original DICOM image into two new files which contained only one type of b-value each. After the splitting, the normalization was performed and resulted in a normalized 3-channel multi-modal image:



(a) **(b)**

Figure 5.2: The figure shows an example of the improved normalization of the 3-channel multi-modal image. a) shows the RGB image with the old normalization and b) shows the image after performing the new normalization method.

5.2 Training of the 2D U-Net Model

The following section will discuss the qualitative and quantitative results obtained from the training of the 2D U-Net model. An additional comparison of the 4-fold and 13-fold cross-validation methods implemented is also included in order to determine which method performed the best.

5.2.1 Validation of 4-Fold and 13-Fold Cross-Validation

From the learning curves in Figures 4.1 and 4.4 we can see that the 2D U-Net model learnt well, and this was evident both for the 4-fold and 13-fold cross-validation methods. It is by interpreting the training and validation loss one gets an indication whether or not the model has a good fit with the data. In order for a learning curve to represent a good fit model it has to have a moderately high training loss at the start of the iterations and gradually decrease when training data is added. As more data is added, the loss flattens and it will not improve the score. This indicates that the model's performance does not improve with more training. Likewise, the validation should behave similarly as the loss and result in a small generalization gap. As shown in the results, this is the case for the learning curves obtained for both cross-validation methods.

Moreover, the learning curves for dice scores show the greatest generalization gap between the training and validation in both folds. This suggests that the network is correctly generalizing and not simply memorizing all the patients from the training set. If the model was memorizing the data it would manifest by having a validation DS higher than the training which kept increasing. In other words, the generalization gap would be large and would indicate overfitting. This is obviously not the case, since the validation is below the training and has a small generalization gap as shown in the learning curves. This implies that our model is generalizing well. However, the dice score is not as high as initially wanted despite the increase in patient data and improvements of normalization. Nonetheless, the model manages to predict cancerous lesions as seen from the patient examples in Figures 4.3 and 4.6.

Based on the average values obtained in Tables 4.2 and 4.4 the weights of the 13-fold cross-validation scores highest for the evaluation metrics. This suggests that this model performs better when considering the voxels and the voxel-vise overlap of the ground truth and automated segmentation. The evaluation metrics are computed based on the segmented voxels being classified as either TP, FN, FP, or TN. To be more precise, the values are representative for a voxel-level analysis and not on a lesion-basis. Additionally, the confusion matrices in Figures 4.2 and 4.5 display how well the model classifies every voxel in the patients used for training and validation in the cross-validations. The number of true negative voxels is superior to those of false positives, false negatives, and true positives, which was expected. The cancerous tumors, classified as TP, are often small and few in numbers in relation to the background and the entire body volume of the patient. As a result, the majority of the quantified voxels are classified as TN. Furthermore, there is a greater proportion of true positives in the training than for the validation. This indicates that the model has learnt well after seeing the patient in the validation set once.

The obvious drawback is the high number of false negative voxels, the absence of segmented voxels which is present in the ground truth but not labelled by the model. For segmenting cancer lesions, it is important to have a model which makes output-sensitive predictions. That is, a non-cancerous lesion needs to be easily flagged as incorrect by a professional when examining the images. Likewise, with deep learning models, a false positive voxel can be removed by post-processing where lesions under a specific size are removed. On the other hand, an undetected lesion can result in a patient being staged incorrectly and receive the wrong treatment. This is of course why there is at least two physicians examining the PET/MRI images separately, naturally to catch the undetected tumors. Unfortunately, there is not an additional model overlooking the segmentation performance. Consequently, a missed cancerous lesion early in the validation can previse a huge impact on the models ability to segment unseen and new patients. This may result in an increased number of voxels being classified as FN. It is therefore crucial to have a model that manages to classify few false negative voxel as early in the training phase.

In general, the results shows that the model works well in terms of the number of correctly classified voxels, but that it still tends to incorrectly label an amount of voxels as FP and FN. Besides, it is noticeably that there is an increase in incorrectly segmented voxels in the validation for the 4-fold. This signals that this trained model is over-labelling. Consequently, this can lead to the creation of lesions that is otherwise not present in the ground truth. This is verified in some of the predictions in Figure 4.3. It is worth noting that the lesion-based analysis is of greater clinical value than the voxel-based study. Even though the superfluous lesions are small and few, they will effect the results negatively when performing a lesion-based analysis.

As noticed in both Tables 4.3 and 4.5 the accuracy, NPV, and sensitivity scores par excellence compared to the other values. However, by collating the equations 2.11, 2.16, and 2.17 it is shown that these metric considers the TN voxels. The occurrence of the high number of TN voxels was drawn attention to earlier, and is an obvious reason for why the metrics scores superior values. Likewise, as discussed earlier, a high voxel accuracy will not always imply that a model has a superior segmentation ability since the accuracy metric can provide misleading results. The number of lesion representations, TP voxels, is small within the image compared to the rest of the patient's body. The measured percentages of voxel accuracy will therefore be biased in mainly reporting how well the model manages to identify the negative cases i.e., where the tumor is not present in the image.

Furthermore, the results from these tables show that both precision and recall varies for each patient but that they are inferior to the aforementioned metrics of specificity, accuracy, and NPV. The fact that precision is lower than recall indicates that the models will return results where most of its predictions are classified incorrectly when comparing it to the ground truth. An ideal model would return both a high recall and a high precision. This would imply that the model segmented the majority of the voxels correctly. A model with a high recall indicates that it is accurate when classifying voxels as positives and will have few false negatives. The recall measures the model's ability to correctly identify the positive voxel in a patient, and this is important for the model in order to be able to make accurate predictions. Due to the main goal being automatic segmentation of cancerous lymph nodes, a higher recall is desired. The fact that the model is resulting in output-sensitive predictions, indicates that it is important to cover the false negative voxels. As mentioned earlier, it is plausible that a non-cancerous lesion is classified as cancerous, but a cancer lesion should not be segmented as non-cancerous.

Naturally, most machine learning algorithms work best when there is an equal amount of samples representing each class of the input data. This is due to deep learning algorithms being designed to reduce errors and maximize the accuracy of the model's predictions [75]. Therefore, if the data-set is imbalanced, the accuracy will be high as the majority of the class is predicted, i.e., the negative voxels. At the same time, the model can fail to capture the true positives, the minority class. This is a problem for our model because the minority class, in other words the

cancer lesions, is the most important finding. In turn, this will lead to a model being more sensitive to segmentation errors for the minority class as compared to the majority.

The quantitative examples of patients in Figures 4.3 and 4.6 clearly show that both the 4-fold and 13-fold cross-validation models manage to segment cancerous lesions from the multi-modal RGB images. Apart from the obvious over-labelling in relation to the shapes and boundaries of the segmented predicted lesions, the model has learned how to distinguish between physiological and pathological uptake of FDG. This is visible when observing patient #1 in the figures. Here there is a clear physiological FDG uptake in the bowel in addition to a cancer lesion in the neck which the model distinctly differentiate between. However, the lymphoma dataset did not contain any patients with bowel metastases, which would have provided a better indication of how well the model is performing. If the model could have been tested on a patient with metastases in the intestine, and not just physiological uptake of FDG, one could have observed and compared how well the model actually handles this as it is only trained on patients with normal bowel uptake. Nevertheless, the model manages to distinguish between normal and abnormal uptake of FDG for the patients in the training and validation.

5.3 Automated Lesion Segmentation

This section will discuss the qualitative and quantitative results obtained from the testing of the 2D U-Net model's segmentation ability of cancerous lymph nodes. Additionally, a comparison between the 4-fold and 13-fold cross-validation methods is included and discussed in order to determine which method performed the best on the unseen patients in the testingset.

5.3.1 Testing of 4-Fold and 13-Fold Cross-Validation

In deep learning, it is preferred to have a varied set of data for both training and testing of the model. The lymphoma dataset contains all the possible Ann Arbor stages of lymphoma, namely stage 1 to stage 4, in addition to cancer free patients with negative PET scans. In fact, since the patient data concerns metastatic lymphomas, the tumor lesions and their placement all differs. This results in an inhomogeneous set of patients suitable for training and testing a CNN model. This is again important for machine learning models as a varied dataset will improve the training yet decrease the chance of getting an overfit model. Furthermore, this leads to a model that generalizes more as the network is varied, which in turn makes it easier for the model to navigate and learn the difference between physiological and pathological uptake of FDG. Another prominent reason for including all stages of lymphoma, in addition to cancer free patients, was to reflect reality. This ensures that our model is suited to handle all possible scenarios for a patient.

Even though the ground truth consists of patients which are both cancer free and diagnosed with limited and advanced disease in the Ann Arbor Staging system, the lymphoma dataset is still unbalanced. There were approximately eleven patients with negative PET scans, deeming the data skewed and biased towards patients with cancerous tumors. An imbalanced dataset is a common problem when it comes to machine learning mainly because it can impact the models ability to predict poorly on the minority of unrepresentative cases. Naturally, it was expected that the dataset would be skewed as PET scans of cancer free or healthy patients are not intentionally performed as it is an invasive examination due to the radiation. Nonetheless, as it is the segmentation of cancer lesions that is the main goal, cancer free patients were included in order to test the model's ability to recognize whether or not a patient is disease free.

Therefore, the testing set consisted of 11 PET/MRI examinations with three cancer free patients whereas the remaining eight included patients with different cancer stages. Thus, by having such a diverse testing set with unseen patient, the performance of the model could be tested and evaluated thoroughly. Figures 4.8 and 4.10 show the quantitative results of five selected patients from the testing-set. Overall, it can be observed that the model manages to automatically segment cancerous lymph nodes from the unseen 3-channel multi-modal images. The over-labelling of lesions, which were apparent in the validation, also appears for the patients in the testing set. Moreover, it is clear that both the 4-fold and 13-fold models fail to segment all lesions from the input images. In other words, the model misses lesions completely and this affects the overall performance. Evidently, from the quantitative examples, the model is better at segmenting larger lesions than the smaller ones.

Subsequently, the dice scores obtained for each of the patients were low despite the promising training results. As mentioned earlier, the dataset was increased from 30 to 64 PET/MRI examinations with the expectation that the increased number of patients would result in a higher dice score for the unseen patients in the testing set. Nonetheless, the dice scores achieved were anticipated to be lower for the unseen patient as compared to the DS that were obtained in the validation. However, not as low as the DS in Tables 4.6 and 4.8 show. An explanation for why the dice scores are low for the unseen testing patients is that the lymphoma dataset consisted of a very inhomogeneous group of patients with respect to age, gender, body volume etc. This makes it harder for the neural network to recognize patterns which can lead to more accurate segmentations. However, the same argument will also ensure that the model is not prone to overfitting as the patient data is diverse, which is also of great importance. Another possibility, which might have affected the low dice score achieved, is the over-labeling of small lesions. In addition, there is an obvious drawback of lesions not being segmented at all. This yields an impact on the overlap between the ground truth and the predictions made by the model.

In the example of the cancer free patient from Figures 4.8 and 4.10, it is shown that the automated segmentation predicts tiny lesions of cancer where the ground

truth is blank. In other words, as stated previously, false positive tumor cases. Consequently, if the model is to be used in the staging process of lymphoma this can lead to a misinterpretation of which cancer stage the patient actually has. A possible explanation for the false positive cases can be a result of overfitting or of segmenting random voxels (noise) where there is a high intensity in the input image. If it is the latter, this can easily be accounted for in a post-processing step by removing segmented voxels below a certain size. Voxel-vise, the confusion matrices for these patients show that there are in total 142 and 137 FP voxels for the 4-fold and 13-fold respectively. This suggests that there is a relative small amount of voxels being segmented as cancerous compared to the total number of voxels in the patient. Another possible reason for why the model segments FP voxel can be explained by the fact that there is only a few cancer free patients in the lymphoma dataset. This implies that the model has not been sufficiently enough trained on such cases and therefore performs poorly.

Additionally, as was shown in Tables 4.6 and 4.8, the cancer free patient scores terrible for most evaluation metrics. In particular, the dice score is exceptionally low and close to zero, which suggests that there is no resemblance between the ground truth and the prediction. Based on the few false positive voxels segmented, it was expected that the DS would be high as the majority of both the ground truth and prediction mask were blank. However, this was not the case. A simple explanation for this is that the DS does not have anything to take the intersection of as there is nothing in the ground truth to be compared against. Consequently, this results in a low value when in reality the masks are similar. This suggests that the dice score metrics is not the best evaluation method for cancer free patients.

An overview of the different dice scores achieved for the patients in the validation and testing of the model is shown in the boxplot in Figure 5.3. From the plot it becomes clear that the median DS is higher for the 13-fold model in the testing of unseen patients, suggesting that this model performs better. A possible explanation for why the 13-fold performs better than the 4-fold is due to it having been trained on a larger training set, implying that the model was tested on a smaller validation set (288 patients for training and 24 for validation in each fold). This means that the model saw more of the available data and resulted in a lower prediction error. In turn, this resulted in a better performance on the unseen data. On the other hand, the highest mean dice score from the validation is varying in performance. The values for both methods implemented, indicates that the model's performance is affected by the different degrees of difficulty for the patients in the example. The complexity of a patient's cancer prevalence will of course have an effect on the models ability to segment cancerous lesions. Consequently, this can result in over- or under-labeling of lesions depending on how similar cases the model has been trained on. As both the training and testing set included all stages of lymphoma, the model should be equipped to handle such situations. In regard to the aforementioned and previously discussed reasons of which model performs the best, it is evident that the 13-fold cross-validation model achieves the highest quantitative and qualitative results when considering a voxel based analysis.

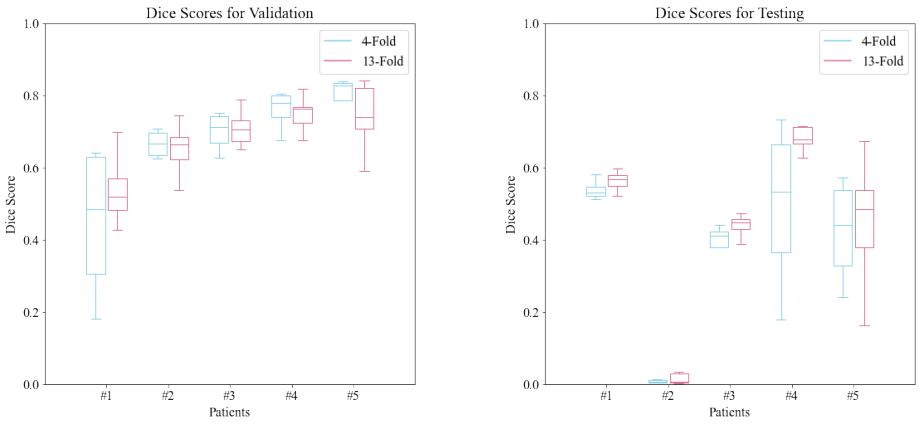


Figure 5.3: The figure shows the boxplots for the dice scores obtained in the validation and testing of the model. The boxplots compares the overall performance with respect to the dice score for the patient examples in Fig 4.3,4.6, 4.8 and 4.10 for both the 4-fold and the 13-fold cross-validation

Initially, it is desirable to achieve a DS close to one. However, an excessively high DS value can prove to be paradoxical. As noted in the results section and in the boxplot above, the dice score for the validation does not exceed over 0.8. All the cancerous lesions were manually segmented by the author, though in clinical practices it is common to have several physicians delineating the tumors where the annotations of ROIs will have small variations from physician to physician. To be more precise, the model is being trained on a ground truth which has only been delineated by one person and could consequently result in it being biased towards those annotations. Therefore, it is an important argument that the dice score should not exceed over 0.85, since a too high value of the DS could consequently lead to overfitting. Another explanation for why the dice score should not exceed 0.85 is based on the research conducted by Zhang et al. [76]. This research showed that when an expert segments the same volume twice, this person achieves a self DS between 0.82 and 0.86, which indicates that there are individual differences each time one segments a volume. Even for professionals, it is difficult to segment the same ROI identically each time. Therefore, it would have been beneficial to have several people segmenting the lesions in the ground truth, which would have lead to a better reflection of reality.

Before the ground truth was increased and the normalization issue resolved, a LOOCV approach was trained and tested. The result below serves as an example of one such patient from the testing set. From Figure 5.4, it is clear that the model is overfitting. This is especially demonstrated with the bladder, which has clearly been segmented as a lesion by the model. Although LOOCV is often a suitable approach when the dataset is modest, this was clearly not the case for the lymphoma dataset. The model could not distinguish between physiological FDG

of the bladder and pathological FDG avid lesions. This is most likely a result of the dataset consisting of too few patients and that the model has learnt the details of the training set to the extent that the performance on unseen data has a negative impact. Thus, consequently, it resulted in obvious errors.

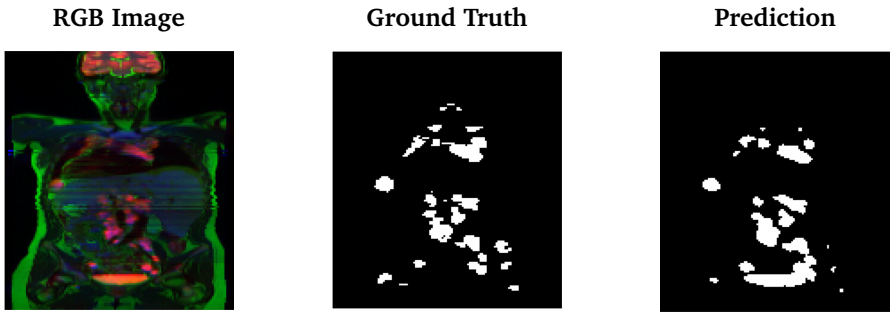


Figure 5.4: An example of model overfitting when the LOOCV method was implemented on a smaller dataset with unresolved normalization. Here the bladder has been segmented as a lesion which indicates overfitting.

5.4 Counting Cancer Lesions

As discussed earlier, it can be of more clinical value to perceive how many cancer lesions the AI has managed to predict correctly when comparing it to the ground truth. This lesion based analysis has a limitation as it only considers the overlay, meaning that a lesion will be considered a TP if only one segmented voxel is found in GT. In other words, this will not provide any further information regarding the size of the detected lesion in comparison to the GT lesion, it will only indicate whether or not that specific lesion is predicted. Furthermore, as was shown in the results, the number of lesions segmented by the model varies, where some patients have a higher number of lesions segmented and others fewer. This suggests that the complexity of a patient's cancer and the quantity of lesions in a patient can affect the performance of the model.

Nevertheless, the results in Tables 4.10, 4.11, 4.12, and 4.13 show that there is a large proportion of false negative lesions detected which suggest, beyond doubt, that the main limitation of the model is that it misses lesions completely. This was also verified with the qualitative testing examples where obvious lesions in patents were not segmented. It is, as stated previously, of greater importance that the false negative lesions are few in comparison to the false positive lesions. Since the model is trained to predict cancer, there is little risk if the model predicts a lesions in a healthy patient as quality assurance will indicate that this patient is cancer free. However, if a patient clearly has several cancer lesions and the model does not predict disease, it is clearly of greater importance as the model fails in its task. It is therefore important to reduce the number of false negative lesions in order to have a model that manages to segment cancerous lesions with accuracy.

Moreover, the TPR and FNR are the most interesting metrics in regard to the lesion based analysis. TPR and FNR shows the number of actual lesions from GT which the AI has managed to detect. It can be seen in the results that the TPR is high for the validation patients, equal or higher than 0.5 for the testing patients, whereas the FNR is high for the testing patients. Likewise the PPV is high, and often found to be higher than the TPR, which illustrates that some of the real lesions are divided into several clusters in AI. This is also perceptible when observing that the numbers for TP_{AI} are higher than the found TP_{GT}. The FDR metrics varies from patient to patient, the values achieved indicates that the model is segmenting false positive lesions. Moreover, this can also illustrates that the model finds it difficult to distinguish between physiological and pathological uptake of FDG for some patients.

Primarily, the most important function of the model is that it manages to predicted the cancerous lesions correctly, whether or not the lesions are accurately dimensioned is not deemed important at this stage. This is especially true since the goal of this thesis is to have a model that segments cancerous lymph nodes. However, if this model one day should be used as an aid to stage cancer patients correctly, then the importance of predicting tumors and the size of the lesions correctly is emphasized. A correct prediction and lesion size are needed in order for a patient to receive the optimal treatment.

Another noteworthy comment about the results is that it was expected that the model achieved higher values for the metrics in the validation relative to the testing in both the 4-fold and 13-fold. This comes from the fact that the patients in the validation have been seen at least once before during training whereas the testing patients are new and unseen because they are used as a final evaluation of the model. Nevertheless, there is a difference in how well these models handle the new and unseen patients. As an example, the 13-fold segments far fewer lesions in the cancer-free patient compared to the 4-fold. This suggests that the overall performance for cancer free patients is superior when using the 13-fold trained model.

5.5 Further Work

Even though the model manages to automatically segment cancerous lymph nodes in the 3-channel multi-modal PET/MR images, there are still a number of components that can be improved to yield better predicted segmentations and results. For instance, the co-registration of the multi-modal PET/MR images should be enhanced in order to achieve a better segmentation of the tumor lesions.

One of the most obvious drawbacks of this model's performance was the fact that it missed cancerous lesions completely. The false negative lesion predictions have to be reduced in order to obtain a more accurate segmentation of FDG avid tumors. This can be achieved by several approaches such as further optimizing the model, increasing the recall by changing the threshold in the training phase, increasing the training and testing data, and additionally include improved pa-

rameters for training. Consequently, by reducing the false negative predictions, the DS will increase as there will be a greater overlap between the ground truth and the predictions. In other words, it will result in masks that are indistinguishable. This again will culminate in a model that performs better on new and unseen cancer patient.

Combining the different imaging modalities into one 3-channel multi-modal image resulted in faster computational time in regard to the training of the model. However, additional information might have been lost from the PET/MR images given that it had to be resampled to the dimension of the DWI. Since the lymphoma dataset comprises several other MR images, it is possible to create channels consisting of additional images such as the DIXON and the DWI with the b-value of 50 s/mm^2. By including more images it could provide the model with further information, which could contribute to better training. Another possibility utilizes training on different channels comprising of PET and MRI. To demonstrate, one could further implement several different channels consisting of an optional number of images, meaning one would not only be limited to three channels like in this thesis. By doing this, one steps away from the 3-channel multi-modal image all together. This would lead to needing more computational power and time to perform the training, but more information would be provided to the model and consequently might provide even better results.

Subsequently, since a deep learning model will provide better results and become more robust as more data is given to the model for training, it is important to increase the dataset further. This is especially true due to the fact that each patient in the lymphoma dataset have different numbers of lesions, the complexities of the cancer varies, and there are additionally different tumor sizes. It is therefore of great importance to have a large enough training set in order to train the model properly, and this can only be achieved by increasing the dataset. Luckily, this is still possible since there are patients left in the dataset which have not been manually segmented yet. Generally, by increasing the data, a deep learning model will be able to learn more thoroughly to distinguish between the abnormalities in the input images as well as to predict more accurate segmentations. Hopefully, this will result in fewer false positives and false negatives cases, both on a voxel- and lesion-basis, which will improve the performance of the model sufficiently.

Furthermore, as both the dataset was increased and the normalization was resolved, it would be interesting to implement the Leave-One-Out Cross-Validation approach once more to test if the improvements would result in a more accurate estimate of the model's prediction. The LOOCV method can, as stated earlier, improve the model with regard to the avoidance of overfitting and additionally help obtain a better statistical foundation. Thus, one would expected that LOOCV would also yield a higher dice score which could benefit the automatic segmentations of new cancer patients. Nonetheless, this is a quite computationally expansive training to perform for a neural network. It would take a few weeks to run and may not even result in better results. Despite this, it would be fascinating to try since the lymphoma dataset only consists of 64 PET/MRI examinations. After

all, a dataset of this size is still considered small for a deep learning model to learn from.

An interesting extension to this project, which could provide an additional clinical value, would be to update the ground truth and label each cancer lesion in regard to the specific lymph node region it appears in. By doing this, one could train the model to both segment the cancerous lesions and label the lesion with its corresponding region. This would be an advantageous aspect if this model were to be used in accordance with the Ann Arbor staging of lymphoma. Only a few modifications of the code would be necessary to implement this approach, but the ground truth would have had to be labeled before training the 2D U-Net.

As previously discussed, the method could benefit from a post-processing step in relation to the size and boundaries of the lesions. First of all, a smoothing function could be implemented in order to get more distinct boundaries for the predicted lesions [77]. Moreover, one could also improve the counting of the lesions by only allowing cancer lesions of a certain size to be detected [78]. In other words, permit a minimum standard size for the lesions discovered such that residual segmentation (noise) consisting of a few voxels or smaller non-cancerous lesions would be excluded and not counted.

Although the model manages to segment cancerous lesions from PET/MR images, it could be favourable to train the 2D U-Net on PET/CT data. There are both advantages and disadvantages to using PET/MRI for lymphoma patients and there are different opinions among professionals if it should be the new standard. In terms of scanning time and the fact that it is a minor invasive procedure, the PET/MRI examination is still difficult to perform for a patient undergoing treatment. This was evident in the lymphoma study where several patients chose to drop out along the way. Therefore, it may also be an idea to use the same method for the PET/CT data since it is already conducted in the lymphoma study. Since PET/CT is the standard modality used for both staging and response assessment in lymphoma, it would be recommended to additionally train a network with respect to this data. Unfortunately, the ground truth is missing for the PET/CT and would have had to be manually segmented before training the neural network with PET/CT images.

At last, this method can be used on other cancer types as the 2D U-Net is designed to handle biomedical images. In this thesis, it was shown that the model managed to segment metastatic lymphoma which is a very variable type of cancer that can appear throughout the whole lymphatic system. The neural network might achieve even better results in for instance brain cancer or lung cancer, where the lesions are more restricted to a specific area of the body. Nonetheless, the network should be able to handle metastasis adequately as it already shows promising results with metastatic lymphoma data.

Chapter 6

Conclusion

The overall goal of this thesis was to develop an automated method for segmentation of cancer-affected lymph-nodes for patients with metastatic lymphoma in PET/MR images using the deep neural network 2D U-Net.

The results from training both a 4-fold and 13-fold model showed that the models obtained low loss scores and average dice scores for the validation and testing set. Moreover, the additional evaluation metrics such as accuracy, specificity, precision, recall, and NPV were used to evaluate the model's performance on a voxel-level. Whereas the results showed a superior performance for the accuracy, specificity, and NPV. However, as these result were dependent on the number of TN classified voxels, which were numerous in comparison to TP, FN, and FP in the patients, it was expected that these metrics scored high values. The number of lesion representations, TP voxels, were small in the 3-channel multi-modal image compared to the rest of the patient's body. Therefore, the measured percentages of voxel accuracy was biased in mainly reporting how well the model managed to identify the negative cases i.e., where the lesions were not present in the image.

Additionally, it was prominent that the predicted segmentations showed a tendency of over-labeling. This was especially true for smaller cancer lesions. In the final performance evaluation of the model, it was shown that the model predicted cancer lesions in the testing set adequately. However, the performance on cancer-free patients were not optimal as the model segmented several false positive voxels despite the fact that the training set had cancer free patients included. Nonetheless, both the 4-fold and 13-fold trained models managed to segment cancerous lesions from the 3-channel multi-modal images.

Furthermore, the lesions based analysis gave a better understanding of how well the 4-fold and 13-fold trained models managed to predict tumors. It was found that the main limitation of the automated segmentation model was the high number of false negative lesions. In other words, the undetected lesions which were clearly detected in the ground truth but not predicted by the model. Moreover, the lesion analysis also illustrated the fact that some of the real lesions from the ground truth were divided into several clusters, i.e., smaller lesions, by the model.

Moreover, for further improvements of the results it is sufficient to increase the training and testing set with several lymphoma patients. The model will become more robust with an even larger training set. Consequently, it will become better at automatically predicting the cancerous lesions in the 3-channel multi-modal images due to the larger variations in tumor size and complexity of patient cases. Simultaneously, the model will become better at distinguishing between physiological and pathological uptake of FDG when it learns from a larger dataset. Additionally, the model should be further optimized before being retrained on new data where for instance changes in the threshold parameters should be implemented. Owing to the improvements, the model is expected to attain a higher dice score, perform better on both the validation and the testing sets, and reduce the number of undetected tumors.

To encapsulate, it was shown that the 13-fold trained model performed better than the 4-fold for both the voxel- and lesion-based analyses. Not only did it achieve the highest DS values, but it additionally segmented the least number of false negative lesions. This was the case for both the cancer free and cancer affected lymphoma patients, indicating that this trained model should be used for further work. In conclusion, the 2D U-Net model automatically segments cancerous tumor lesions in the 3-channel multi-modal images yielding promising results for future research.

Bibliography

[1] World Health Organization. “Cancer.” (2022), [Online]. Available: https://www.who.int/news-room/fact-sheets/detail/cancer.

[2] S. Shanbhag and R. F. Ambinder, “Hodgkin lymphoma: A review and update on recent progress,” *CA Cancer Journal Clinicians*, vol. 68, no. 2, 2018. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC5842098/.

[3] J. Cheng and Y. Weerakkody. “Lugano staging classification.” (2018), [Online]. Available: https://radiopaedia.org/articles/63811.

[4] S. Zafar, R. Sharma, J. Cunningham, P. Mahalingam, A. Attygalle, N. Khan, D. Cunningham, D. El-Sharkawi, and S. Iyengar, “Current and future best practice in imaging, staging and response assessment for non-hodgkin´s lymphomas: The specialist integrated haematological malignancy imaging reporting (sihmir) paradigm shift,” *Clinical Radiology*, vol. 76, no. 5, 2021. DOI: 10.1016/j.crad.2020.12.022.

[5] E. Lysheim. “Analysis of quantitative susceptibility mapping in healthy volunteers at 3t and 7t.” (2021), [Online]. Available: https://ntnuopen.ntnu.no/ntnu-xmlui/handle/11250/2788832.

[6] 180N. “Clinical pet/mri work package 5:machine learning.” (2022), [Online]. Available: https://180n.no/wp/clinical-wp5/.

[7] The global cancer observatory. “Non-hodgkin lymphoma.” (2020), [Online]. Available: https://gco.iarc.fr/today/data/factsheets/cancers/34-Non-hodgkin-lymphoma-fact-sheet.pdf.

[8] Kreftlex, institutt for kreftgenetikk og informatikk. “Lymfom-hodgkin.” (2021), [Online]. Available: https://www.kreftlex.no/Lymfom-Hodgkin.

[9] S. Watson. “What is lymphoma?” (2020), [Online]. Available: https://www.webmd.com/cancer/lymphoma/lymphoma-cancer.

[10] S. Johnson, A. Kumar, M. Matasar, H. Schöder, and J. Rademaker, “Imaging for staging and response assessment in lymphoma,” *Radiology*, vol. 276, no. 2, 2015. DOI: https://doi.org/10.1148/radiol.2015142088.

[11] B. Cheson, R. Fisher, F. C. S.F. Barrington, L. Schwartz, E. Zucca, and T. Lister, "Recommendations for initial evaluation, staging, and response assessment of hodgkin and non-hodgkin lymphoma: The lugano classification," *Journal of Clincal Oncology*, vol. 32, no. 27, 2014. DOI: 10.1200/JCO.2013.54.8800.

[12] N. Kulkarni, D. Pinho, S. Narayanan, A. Kambadakone, J. Abramson, and D. Sahani, "Imaging for oncologic response assessment in lymphoma," *American Journal of Roentgenology*, vol. 208, no. 1, 2017. [Online]. Available: https://www.ajronline.org/doi/10.2214/AJR.16.16180.

[13] J. Cheng and R. Sharma. "Ann arbor staging system." (2018), [Online]. Available: https://radiopaedia.org/articles/63815.

[14] R.Pfleger and H. Knipe. "Deauville five-point scale." (2014), [Online]. Available: https://radiopaedia.org/articles/32555.

[15] M. Emanuele Zucca and M. Francesca Pavanello. "The lugano classification recommendations for hodgkin's and non-hodgkin's lymphoma:staging, response assessment and follow up." (2014), [Online]. Available: https://oncologypro.esmo.org/content/download/75355/1377102/file/ESMO-E-Learning-The-Lugano-Classification-Recommendations-For-Hodgkins-and-Non-Hodginks-Lymphoma-Staging-Response-Assessment-and-Follow-Up-Zucca-Pavanello.pdf.

[16] The Johns Hopkins University. "Positron emission tomography (pet)." (2021), [Online]. Available: https://www.hopkinsmedicine.org/health/treatment-tests-and-therapies/positron-emission-tomography-pet.

[17] "Radiopharmaceuticals." (2021), [Online]. Available: https://learn-eu-central-1-prod-fleet01-xythos.content.blackboardcdn.com/5def77a38a2f7/9058655?X-Blackboard-Expiration=1633791600000&X-Blackboard-Signature=HJybdxXF9qo0DXkJ5DNbUoDdio1vEsUiRwks.

[18] S. I. Ziegler, "Positron emission tomography: Principles, technology, and recent developments," *Nuclear Physics A*, vol. 752, 2005. [Online]. Available: https://doi.org/10.1016/j.nuclphysa.2005.02.067.

[19] J. J. Vaquero and P. Kinahan, "Positron emission tomography: Current challenges and opportunities for technological advances in clinical and preclinical imaging systems," *Annual review of biomedical engineering*, vol. 17, 2015. [Online]. Available: https://www.annualreviews.org/doi/10.1146/annurev-bioeng-071114-040723.

[20] E. J. Hall and A. J. Giaccia, *Radiobiology for the Radiologist*. Lippincott Williams and Wilkins, 2012.

[21] W. W. Moses, "Fundamental limits of spatial resolution in pet," *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment*, vol. 648, 2011. [Online]. Available: https://doi.org/10.1016/j.nima.2010.11.092.

[22] A. K. Buck and S. N. Reske, "Cellular origin and molecular mechanisms of 18f-fdg uptake: Is there a contribution of the endothelium?" *Journal of Nuclear Medicine*, vol. 45, no. 3, 2004. [Online]. Available: https://jnm.snmjournals.org/content/45/3/461.

[23] A. Rahmouni, A. Luciani, and E. Itti, "Mri and pet in monitoring response in lymphoma," *Cancer Imaging*, vol. 5 (Spec No A), no. S106–S112, 2005. [Online]. Available: 10.1102/1470-7330.2005.0038.

[24] H.Schöder, J.Meta, C.Yap, M.Ariannejad, J.Rao, M.E.Phelps, P.E.Valk, J.Sayre, and J.Czernin, "Effect of whole-body (18)f-fdg pet imaging on clinical staging and management of patients with malignant lymphoma," *Journal of Nuclear Medicine*, vol. 42, no. 8, 2001. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/11483671/.

[25] D. Dance, S. Christofides, A. Maidment, I. McLean, and K. Ng, *Diagnostic Radiology Physics: A Handbook for Teachers and Students*. IAEA, 2014.

[26] A. Bjørnerud. "The physics of magnetic resonance imaging." (2008), [Online]. Available: https://www.uio.no/studier/emner/matnat/fys/FYS-KJM4740/v14/kompendium/compendium-mri-feb-2009.pdf.

[27] mriQuestions. "T1 relaxation: Definition." (2021), [Online]. Available: http://mriquestions.com/what-is-t1.html.

[28] mriQuestions. "T2 relaxation: Definition." (2021), [Online]. Available: http://mriquestions.com/what-is-t2.html.

[29] J. Ballinger and A. Muphy. "Mri pulse sequence." (2013), [Online]. Available: https://doi.org/10.53347/rID-21957.

[30] mriQuestions. "What is k-space?" (2021), [Online]. Available: https://mriquestions.com/what-is-k-space.html.

[31] C. Malamateniou, S. Malik, S. Counsell, J. Allsop, A. McGuinness, T. Hayat, K. Broadhouse, R. Nunes, A. Ederies, J. Hajnal, and M. Rutherford, "Motion-compensation techniques in neonatal and fetal mr imaging," *American Journal of Neuroradiology*, vol. 34, no. 6, 2013. [Online]. Available: http://www.ajnr.org/content/34/6/1124.

[32] mriQuestions. "Methods for filling k-space." (2021), [Online]. Available: https://mriquestions.com/k-space-trajectories.html.

[33] mriQuestions. "Haste/ss-fse." (2021), [Online]. Available: https://mriquestions.com/hastess-fse.html.

[34] U. Bashir and A. Goel. "Diffusion-weighted imaging." (2012), [Online]. Available: https://doi.org/10.53347/rID-16718.

[35] A. Goel. "B values." (2014), [Online]. Available: https://doi.org/10.53347/rID-26733.

[36] A. B. Rosenkrantz, K. Friedman, H. Chandarana, A. Melsaether, L. Moy, Y.-S. Ding, K. Jhaveri, and L. B. andRajan Jain. "Current status of hybrid pet/mri in oncologic imaging." (2015), [Online]. Available: https://www.ajronline.org/doi/pdf/10.2214/AJR.15.14968.

[37] MR-tip. "Biograph mmr." (2011), [Online]. Available: https://www.mr-tip.com/serv1.php?type=db1&dbs=Biograph+mMR.

[38] MathWorks. "What is deep learning?" (2021), [Online]. Available: https://www.mathworks.com/discovery/deep-learning.html.

[39] IBM Cloud Education. "Deep learning." (2020), [Online]. Available: https://www.ibm.com/cloud/learn/deep-learning.

[40] MathWorks. "What is a neural network?" (2021), [Online]. Available: https://www.mathworks.com/discovery/neural-network.html.

[41] MathWorks. "What is a convolutional neural network?" (2021), [Online]. Available: https://www.mathworks.com/discovery/convolutional-neural-network-matlab.html.

[42] IBM Cloud Education. "Convolutional neural networks." (2020), [Online]. Available: https://www.ibm.com/cloud/learn/convolutional-neural-networks.

[43] S. Saha. "A comprehensive guide to convolutional neural networks — the eli5 way." (2018), [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[44] A. Refsgaard, F. Tseng, and G. Kogan. "How neural networks are trained." (2016), [Online]. Available: https://ml4a.github.io/ml4a/how_neural_networks_are_trained/.

[45] J. Brownlee. "Loss and loss functions for training deep learning neural networks." (2019), [Online]. Available: https://machinelearningmastery.com/loss-and-loss-functions-for-training-deep-learning-neural-networks/.

[46] L. Miller. "Machine learning week 1: Cost function, gradient descent and univariate linear regression." (2018), [Online]. Available: https://medium.com/@lachlanmiller_52885/machine-learning-week-1-cost-function-gradient-descent-and-univariate-linear-regression-8f5fe69815fd.

[47] Baeldung. "Training and validation loss in deep learning." (2022), [Online]. Available: https://www.baeldung.com/cs/training-validation-loss-deep-learning.

[48] M. Nielsen. "How neural networks are trained." (2019), [Online]. Available: http://neuralnetworksanddeeplearning.com/chap1.html.

[49] A. Murphy. "Batch size (machine learning)." (2019), [Online]. Available: https://radiopaedia.org/articles/batch-size-machine-learning.

[50] Peltarion. "Binary cross entropy." (2022), [Online]. Available: https://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/binary-crossentropy.

[51] Deepchecks. "Classification in machine learning." (2022), [Online]. Available: https://deepchecks.com/glossary/binary-classification/.

[52] G. Hinton, N. Srivastava, and K. Swersky. "Neural networks for machine learning." (2022), [Online]. Available: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[53] J. Jordan. "Evaluating image segmentation models." (2018), [Online]. Available: https://www.jeremyjordan.me/evaluating-image-segmentation-models/.

[54] E. Tiu. "Metrics to evaluate your semantic segmentation model." (2019), [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2.

[55] A. F. Gad. "Evaluating deep learning models: The confusion matrix, accuracy, precision, and recall." (2021), [Online]. Available: https://blog.paperspace.com/deep-learning-metrics-precision-recall-accuracy/.

[56] N. Shrivastav. "Confusion matric(tpr,fpr,fnr,tnr), precision, recall, f1-score." (2020), [Online]. Available: https://medium.datadriveninvestor.com/confusion-matric-tpr-fpr-fnr-tnr-precision-recall-f1-score-73efa162a25f.

[57] S. Rouam. "False discovery rate (fdr)." (2013), [Online]. Available: https://link.springer.com/referenceworkentry/10.1007/978-1-4419-9863-7_223.

[58] H. Wang and H. Zheng. "Negative predictive value." (2013), [Online]. Available: https://link.springer.com/referenceworkentry/10.1007/978-1-4419-9863-7_234.

[59] A. Mitrani. "Evaluating categorical models ii: Sensitivity and specificity." (2019), [Online]. Available: https://towardsdatascience.com/evaluating-categorical-models-ii-sensitivity-and-specificity-e181e573cff8.

[60] X. Ying, "An overview of overfitting and its solutions," *Journal of Physics: Conference Series*, vol. 1168, no. 2, 2019. DOI: https://iopscience.iop.org/article/10.1088/1742-6596/1168/2/022022/meta.

[61] J. Brownlee. "How to avoid overfitting in deep learning neural networks." (2018), [Online]. Available: https://machinelearningmastery.com/introduction-to-regularization-to-reduce-overfitting-and-improve-generalization-error/.

[62] O. Ronneberger, P. Fischer, and T. Brox. "U-net: Convolutional networks for biomedical image segmentation." (2015), [Online]. Available: https://arxiv.org/pdf/1505.04597.pdf.

[63] J. Zhang. "Unet — line by line explanation." (2019), [Online]. Available: https://towardsdatascience.com/unet-line-by-line-explanation-9b191c76baf5.

[64] W. Wang. "Image segmentation using deep learning regulated by shape context." (2018), [Online]. Available: https://www.diva-portal.org/smash/get/diva2:1203974/FULLTEXT01.pdf.

[65] NTNU. "Hunt cloud data." (2021), [Online]. Available: https://www.ntnu.edu/mh/huntcloud/.

[66] NTNU. "Machine types." (2021), [Online]. Available: https://docs.hdc.ntnu.no/services/machine-types/#compute-optimized-machine-types.

[67] Slicer Community. "Image segmentation." (2020), [Online]. Available: https://slicer.readthedocs.io/en/latest/user_guide/image_segmentation.html.

[68] A. Fedorov, R. Beichel, J. Kalpathy-Cramer, and et al., "3d slicer as an image computing platform for the quantitative imaging network," *Magn Reson Imaging*, vol. 30, no. 9, 2012. [Online]. Available: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC3466397/.

[69] Slicer. "3d slicer image computing platform." (2021), [Online]. Available: https://www.slicer.org/.

[70] L. G.Nyúl and J. K. Udupa. "On standardizing the mr image intensity scale." (1999), [Online]. Available: https://onlinelibrary.wiley.com/doi/epdf/10.1002/%28SICI%291522-2594%28199912%2942%3A6%3C1072%3A%3AAID-MRM11%3E3.0.CO%3B2-M.

[71] J. Brownlee. "A gentle introduction to k-fold cross-validation." (2018), [Online]. Available: https://machinelearningmastery.com/k-fold-cross-validation/.

[72] G. James, D. Witten, T. Hastie, and R. Tibshirani, *An Introduction to Statistical Learning: with Applications in R*. Springer, 2013.

[73] J. Brownlee. "Loocv for evaluating machine learning algorithms." (2020), [Online]. Available: https://machinelearningmastery.com/loocv-for-evaluating-machine-learning-algorithms/.

[74] Usman Bashir and Yahya Baba. "Diffusion-weighted imaging." (2021), [Online]. Available: https://doi.org/10.53347/rID-16718.

[75] J. Brownlee. "Failure of classification accuracy for imbalanced class distributions." (2020), [Online]. Available: https://machinelearningmastery.com/failure-of-accuracy-for-imbalanced-class-distributions/.

[76] Y. Zhang, H. Wei, M. J.Cronin, N. He, F. Yan, and C. Liu, "Longitudinal atlas for normative human brain development and aging over the lifespan using quantitative susceptibility mapping," *NeuroImage*, vol. 171, 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1053811918300089.

[77] S. Nazlibilek, D. Karacor, T. Ercan, M. H. Sazli, O. Kalender, and Y. Ege, "Automatic segmentation, counting, size determination and classification of white blood cells," *Measurement*, vol. 55, 2014. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0263224114001663?via%3Dihub.

[78] S. Kothari, Q. Chaudry, and M. D. Wang, "Automated cell counting and cluster segmentation using concavity detection and ellipse fitting techniques," in *2009 IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, 2009, pp. 795–798. DOI: 10.1109/ISBI.2009.5193169.

[79] itk-SNAP. "Itk-snap." (2017), [Online]. Available: http://www.itksnap.org/pmwiki/pmwiki.php.

Appendix A

Appendix A

This Appendix includes information for the PET/CT data acquisition, segmentation in ITK-Snap and 3D Slicer, and additional results. The information presented in sections A.1 and A.2 were previously described in the specialization project.

A.1 PET/CT Lymphoma Data Acquisition

As already mentioned, a total of 108 (61 baseline, 13 interim, and 34 end of treatment) PET/MRI were performed. There are additionally 108 PET/CT examinations with the same amount of baseline, interim and EOT images acquired in this study cohort. The 61 patients were scanned with a PET/CT directly followed by PET/MRI at baseline.The interim was attained only for the cHL after 2 cycles of chemotherapy. The end-of-treatment were obtained for both cHL and DLBCL after 3-6 weeks following the last cycle of chemotherapy. Both interim and EOT images were preformed on PET/CT and PET/MRI for a subgroup of the patients when PET/CT was clinically indicated.

All PET/CT and PET/MRI was acquired by using a single intravenously injection of 18F-FDG. The PET/CT images were obtained on a Siemens Biograph mCT. A hybrid PET/MR system (Siemens Biograph mMR) was used for simultaneous PET and MRI acquisitions. The study include coronal Dixon-Vibe, transversal diffusion weighted MRI with b-values: 50 and 800, transversal T2-HASTE and coronal T2-TIRM. The PET image reconstruction was performed with iterative reconstruction (3D OSEM algorithm, 3 iterations, 21 subsets, 4mm Gaussian filter) with point spread function, decay and scatter-correction. The time-of-flight was used on PET/CT, unfortunately it was not available on the hybrid PET/MRI system. The attenuation correction for the PET/CT was accomplished with the low dose CT images converted to the 511kev photons in PET. And as already mentioned, the attenuation correction for the PET/MRI used the Dixon-Vibe sequence.

A.2 Segmentation in ITK-SNAP and 3D-Slicer

ITK-SNAP Software Info

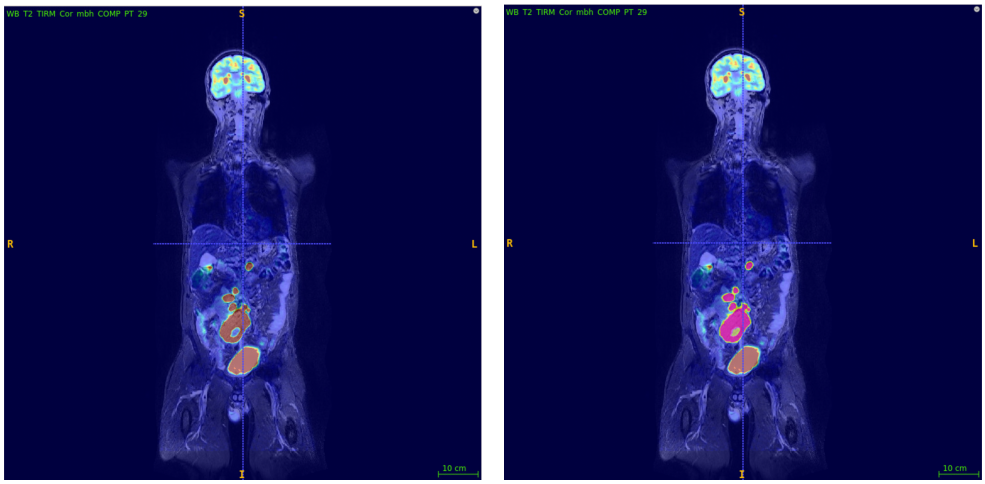
ITK-SNAP is free, open-source, and multi-platform software that is commonly used to segment structures in 3D medical images. The software is a product of collaboration between Paul Yushkevich, Ph.D., of the Penn Image Computing and Science Laboratory (PICSL) at the University of Pennsylvania, and Guido Gerig, Ph.D., of the Scientific Computing and Imaging Institute (SCI) at the University of Utah [79]. Their vision for ITK-SNAP was to create a software that would be dedicated to segmentation and would be easy to both use and learn.

The software provides a semi-automatic segmentation using active contour methods and the user has the opportunity to manually delineate and navigate in the image. Furthermore, the software offers several supporting utilities in addition to the core functions mentioned. The design of ITK-SNAP is focused specifically on the problem of image segmentation which differs it from other larger open-source image analysis tools and software. Moreover, the software design emphasizes the interaction and ease of use, where the bulk of the development effort is dedicated to the user interface [79].

ITK-SNAP

In order to visualize both the FDG-PET and MRI images in ITK-SNAP one has to overlay the images and choose different color maps to clearly distinguish between pathological and physiological uptake of FDG in the PET image. First start by uploading the chosen MRI image, either the T2 HASTE or T2 TIRM should be sufficient. Thereafter, under the File tab choose Add another image and upload the *MRAC PET HD image. It is important to select the display of the image as a semi-transparent overlay. This feature enables the PET image to be shown on top of the MRI. The image contrast can be changed manually or ITK Snap can do an automatic linear contrast adjustment. The recommended color maps for the PET image is either the Jet or the Hot map since the typically used inverted gray scale is not available in this software. The Figure A.1 shows the PET imaged as a semi-transparent overlay over the T2 TIRM. From the image one can clearly see the physiological uptake of FDG in both the brain and the bladder, however, there are several cancer lesions in the abdomen.

Before starting the segmentation process, one needs to name and choose colors for the segmentation labels. This can be done by clicking the Segmentation tab and selecting the Label Editor. The author decided to use pink for the gross tumor volume and yellow for the tumor boundaries. With this in mind, it is time to start the segmentation of each cancer node. ITK-SNAP provides several tools for segmentations and additionally an auto segmentation for 3D segment. The author decided against using the auto segmentation as it labelled lesions and parts of organs that were not affected by cancer. Instead the polygon inspector were used. The tool enables the user to draw freehand shapes around the cancer le-



(a) **(b)**

Figure A.1: The figure shows the FDG-PET scan on top of the T2 TIRM MRI image in ITK-SNAP of a patient diagnosed with DLBCL in a coronal view. The MRI image has a Grayscale color map while the PET has the jet color map. a) shows several cancer lesions, the red nodes, in the abdomen are clearly visible, whereas b) shows the same patient where the lesions have been manually segmented and are depicted in pink for the gross tumor volume and in yellow for the tumor boundary.

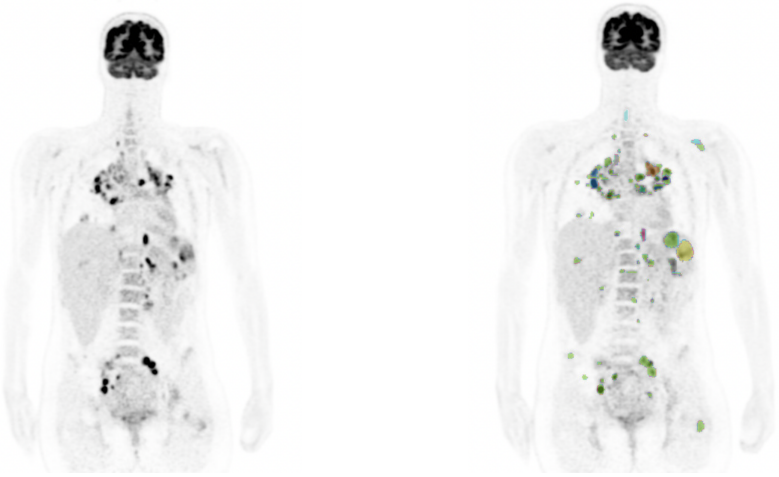
sions in one slice. In other words, the user has to draw segmentations around cancer lesions in every image slice they appear in. The paint brush inspector tool is a great asset if one manages to segment too few or too many pixels. It is possible to change the brush size to fit a single pixel, and thus, the user can easily remove or add segmentation labels in a specific lesion. The Figure A.1 show the result of the segmented cancer lesions in pink and yellow in one image slice of the previous patient shown in Figure A.1.

In the end, after manually segmenting every cancer lesions in each image slice the segmentations where saved as a NifTI file. As already mentioned, the first five patients were manually segmented using ITK-SNAP. After the segmentations were validated by the nuclear physician, it was decided that the remaining patients would be segmented using 3D Slicer. In addition to handling the spacing issue between the image slices, 3D Slicer has the option to display the PET image in an inverted gray scale which is commonly used by the nuclear physicians at the hospital. Moreover, the software has extension packages including different PET image segmentation additions that are advantageous for further segmentations

3D-Slicer

Even though the extension packages are quite beneficial for the manual segmentation, the navigation in 3D Slicer is a bit trickier and not so straight forward as compared with ITK-SNAP. Nevertheless, in order to upload the DICOM images to

the software, hit the import DICOM files and download the T2-TIRM, T2-Haste and *MRAC PET images. After loading the images, find the Volumes tab in the modules menu and navigate to the display section. Under the display the user has the option to change the contrast in the image and the color map. It is recommended to select the Auto W/L and the Lookup Table (color map) Inverted Grey for the PET and Grey for the MRI images. Moreover, to be able to overlay the PET and MRI in 3D Slicer one first have to find the Data tab in the modules menu. Under the node section, one has the opportunity to choose with patient and corresponding images to display. First, press the eye symbol on the right to select the MRI image, thereafter right-click the eye on the PET image and chose "Show in slice view as foreground" and as a result, the images are shown on top of each other.



(a) **(b)**

Figure A.2: a) shows how the FDG-PET appears in 3D Slicer of a patient diagnosed with cHL in coronal view. Several cancer lesions, the black nodes in the image, are clearly visible. b) shows how the manually segmentation of cancer nodes appear in 3D Slicer for the same cHL patient in coronal view. The segmentations of the cancer lesions are depicted in different colors.

However, this is not necessarily to do every time as it is possible to download extension packages as mentioned earlier. This is easily done by going to the extension manager in the software and downloading the PET extensions called: PET-IndiC, PETDICOMExtension and PETTumorSegmentation. Afterwards, close 3D Slicer and start the software once more, and the packages should be downloaded correctly. After this, the user can once more go to the modules menu and find the Quantification and then chooses the PET indiC. The user will then be directed to useful segmentation tools. The Figure A.2 shows how the FDG-PET appears in coronal view in the PET indiC.

Once directed to the PET indiC, the user must chose the *MRAC PET as the Input image to the left, and thereafter rename the label image beneath. The label

image will be where the segmentations are saved, therefore it is recommended to name it sufficiently. Once this step is finished, it is time to start the actual segmentation of the cancer nodes. To begin with thee user should select the tool called PetTumorSegmentationEffect. This tool is an semi-automatic segmentation that manages to segment cancer lesions in several slices at the same time. After hitting the tool, click on one of the defined black cancer nodes in the PET image to the right in 3D Slicer. Slicer will then give a window explaining that it is doing the calculation and thereafter the node in the PET image will be segmented with the color of the label. By scrolling through the different slices in the PET image, the user will clearly see the lesion marked in all slices where it is present. However, an important comment about this tool is that the user has to remember to change segmentation label for each time the PetTumorSegmentationEffect tool is used to segment different cancer nodes. This is easily done by hitting the upwards arrow in the label under Edit Selected Label Map (the label name will change). If for instance, the patient the user is working on has several cancer nodes and the same label color appear again, it is not an issue. Even though the color appear once more, the label itself has another number and the cancer node will be segmented.

Although the PetTumorSegmentationEffect tool is useful and saves the user much time, it sometimes has difficulty in covering the boundaries of the tumor nodes. If this happens, the user may either draw the missing pixel by hand or use the LevelTracingEffect tool which marks the whole shape of the node. However, if one selects the LevelTracingEffect tool for a lesion, then the user must remember to manually mark the lesion in every slice where it is present. It is not necessarily to change the label for marking the same lesion in different slices when using the LevelTracingEffect tool. The Figure A.2 shows how a fully segmented PET image in 3D Slicer looks like after using the PET indiC package. The segmented cancer nodes are shown in the coronal view. That being said, it is important and beneficial to work in all image views while manually segmenting the patients. The different cancer lesions can appear differently in the axial, coronal and sagittal views, and by using all three the user will be able to makes sure that every node is found and segmented correctly.

Finally, when the segmentation of the cancer nodes is finished one must remember to save it. This is done by clicking on the save tab in the left corner. Before saving the segmentation, it is important to change the folder where the file is located and select to save is as a NiFTI. After having discussed the segmentations with the nuclear physician, one can easily open the saved NiFTI files in either ITK-SNAP or 3D Slicer to make essential changes in the patient segmentations.

A.3 Additional Results

This section presents additional results from the thesis work.

A.3.1 Training of 2D U-Net

4-Fold Cross-Validation

Table A.1 shows the evaluation metrics for all 53 patients used in the validation of the 4-fold cross-validation.

Table A.1: Average values for all the evaluation metrics for all patients used in the validation of the 4-fold.

Evaluation Metrics for All Patients Used in the Validation of the 4-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
1	0.0010	0.4492	0.9995	0.3803	0.6199	0.9994	0.9999
2	0.0009	0.6661	0.9996	0.5960	0.7549	0.9994	0.9998
3	0.0143	0.7008	0.9955	0.6878	0.7179	0.9916	0.9961
4	0.0035	0.7605	0.9990	0.7379	0.7873	0.9983	0.9993
5	0.0007	0.7937	0.9997	0.7444	0.8508	0.9996	0.9999
6	0.0014	0.7810	0.9995	0.7384	0.8299	0.9992	0.9997
7	0.0010	0.6116	0.9996	0.5584	0.6833	0.9994	0.9998
8	0.0050	0.7503	0.9981	0.6886	0.8250	0.9972	0.9991
9	0.0007	0.6965	0.9998	0.6874	0.7078	0.9996	0.9998
10	0.0045	0.6138	0.9988	0.6110	0.6218	0.9977	0.9989
11	0.0003	0.4686	0.9998	0.4037	0.6305	0.9998	1.000
12	0.0014	0.6385	0.9995	0.5992	0.6844	0.9992	0.9997
13	0.0006	0.7591	0.9997	0.6944	0.8375	0.9996	0.9999
14	0.0044	0.7219	0.9986	0.6855	0.7625	0.9977	0.9991
15	0.0076	0.7406	0.9977	0.7267	0.7673	0.9960	0.9982
16	0.0120	0.7042	0.9969	0.7060	0.7078	0.9940	0.9970
17	0.0018	0.5111	0.9996	0.5054	0.5583	0.9992	0.9997
18	0.0059	0.7382	0.9982	0.7253	0.7527	0.9967	0.9985
19	0.0019	0.8052	0.9994	0.7825	0.8301	0.9990	0.9996
20	0.0017	0.6458	0.9994	0.5986	0.7017	0.9991	0.9996
21	0.0002	0.2314	0.9999	0.1912	0.2938	0.9999	1.000
22	0.0052	0.6020	0.9991	0.6317	0.5933	0.9980	0.9989
23	0.0001	0.0107	0.9999	0.0000	-	0.9999	1.000
24	0.0023	0.6196	0.9991	0.5690	0.6905	0.9987	0.9995
25	0.0018	0.6744	0.9993	0.6163	0.7480	0.9989	0.9996
26	0.0118	0.5132	0.9974	0.5294	0.5161	0.9945	0.9971
27	0.0024	0.5304	0.9994	0.5174	0.5458	0.9988	0.9994
28	0.0060	0.6076	0.9984	0.6043	0.6280	0.9970	0.9986

29	0.0001	0.0137	0.9999	0.0000	-	0.9999	1.000
30	0.0001	0.4841	0.9999	0.3894	0.6422	0.9999	1.000
31	0.0000	0.0177	1.0000	0.0000	-	1.0000	1.000
32	0.0090	0.6491	0.9973	0.6109	0.6978	0.9953	0.9980
33	0.0001	0.0126	0.9999	0.0000	-	0.9999	1.000
34	0.0002	0.0062	0.9998	0.0000	-	0.9998	1.000
35	0.0073	0.7526	0.9980	0.7481	0.7648	0.9961	0.9982
36	0.0002	0.0106	0.9999	0.0000	-	0.9999	1.000
37	0.0049	0.4997	0.9991	0.5095	0.5442	0.9981	0.9990
38	0.0008	0.5604	0.9998	0.5083	0.6254	0.9997	0.9999
39	0.0001	0.0106	0.9999	0.0000	-	0.9999	1.000
40	0.0048	0.7623	0.9987	0.7645	0.7688	0.9974	0.9987
41	0.0151	0.7210	0.9942	0.6500	0.8127	0.9918	0.9975
42	0.0001	0.0098	0.9999	0.0000	-	0.9999	1.000
43	0.0049	0.7757	0.9986	0.7710	0.7883	0.9974	0.9988
44	0.0005	0.3892	0.9999	0.3448	0.4664	0.9998	0.9999
45	0.0001	0.1217	0.9999	0.0714	0.3491	0.9999	1.0000
46	0.0022	0.7495	0.9994	0.7325	0.7690	0.9989	0.9995
47	0.0046	0.7717	0.9984	0.7349	0.8146	0.9974	0.9990
48	0.0001	0.0076	0.9999	0.0000	-	0.9999	1.000
49	0.0090	0.8137	0.9975	0.7892	0.8406	0.9957	0.9982
50	0.0007	0.6163	0.9997	0.5596	0.6911	0.9996	0.9998
51	0.0002	0.0082	0.9999	0.0000	-	0.9999	1.000
52	0.0091	0.7759	0.9976	0.7650	0.7950	0.9956	0.9980
53	0.0042	0.7815	0.9988	0.7621	0.8040	0.9979	0.9991

Figure A.3 shows the confusion matrices for the five patients used in Figure 4.3 from the validation of the 4-fold.

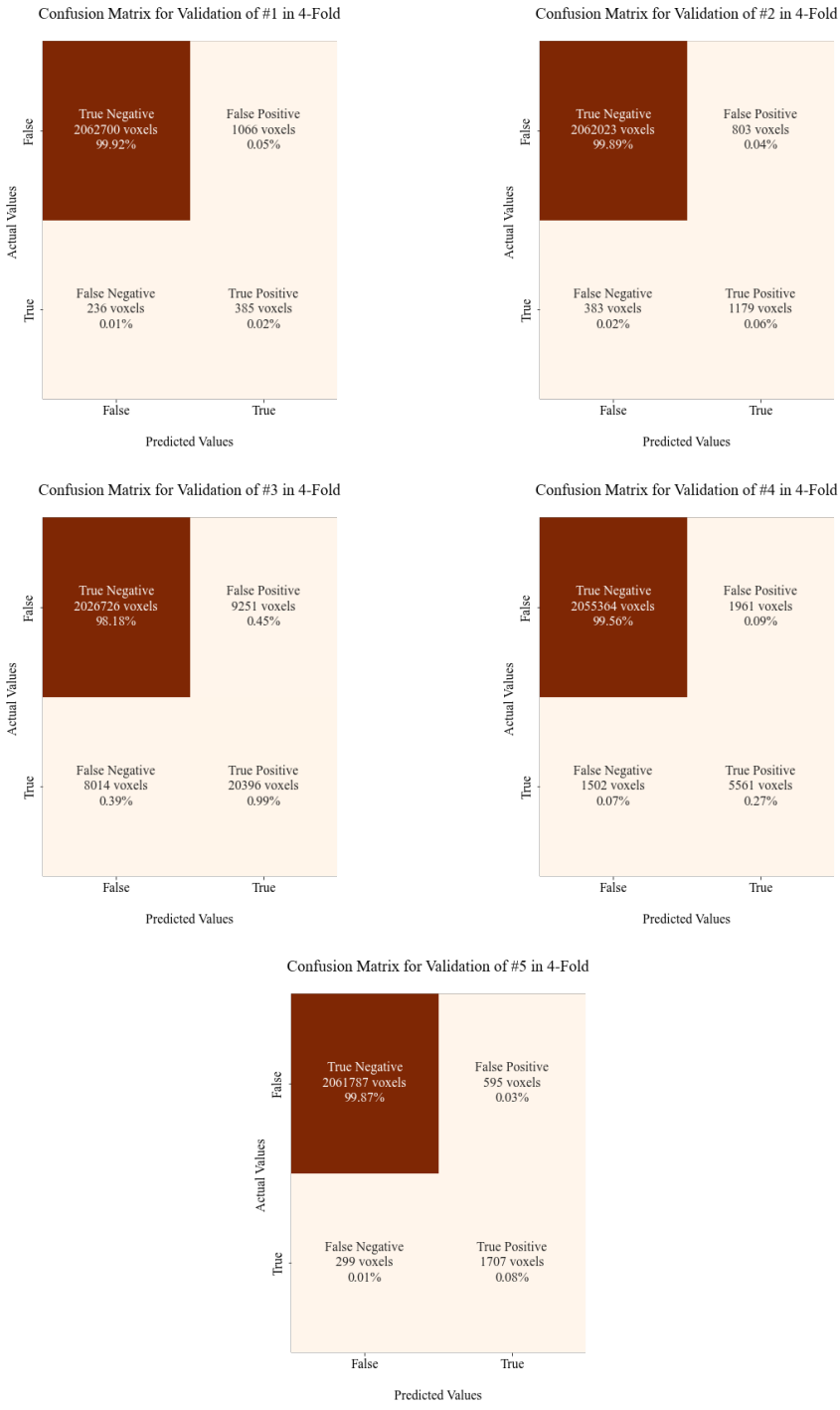


Figure A.3: The figure shows the confusion matrices for the patients in Figure 4.3

13-Fold Cross Validation

Table A.2 shows the evaluation metrics for all 53 patients used in the validation of the 13-fold cross-validation.

Table A.2: Average values for all the evaluation metrics for all the patients used in the validation of the 13-fold

Evaluation Metrics for All Patients Used in the Validation of the 13-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
1	0.0007	0.5089	0.9996	0.4557	0.6277	0.9995	0.9999
2	0.0009	0.6582	0.9997	0.6238	0.7091	0.9994	0.9998
3	0.0126	0.7101	0.9960	0.7150	0.7084	0.9921	0.9959
4	0.0031	0.7563	0.9991	0.7566	0.7604	0.9983	0.9992
5	0.0008	0.7491	0.9996	0.7053	0.8101	0.9995	0.9998
6	0.0013	0.7666	0.9995	0.7349	0.8049	0.9992	0.9997
7	0.0009	0.6237	0.9997	0.6034	0.6520	0.9994	0.9998
8	0.0045	0.7482	0.9983	0.7068	0.7995	0.9972	0.9990
9	0.0006	0.7116	0.9998	0.6981	0.7289	0.9996	0.9998
10	0.0040	0.6345	0.9990	0.6637	0.6239	0.9979	0.9989
11	0.0002	0.5513	0.9999	0.4892	0.6782	0.9999	1.000
12	0.0014	0.6239	0.9994	0.5719	0.6981	0.9991	0.9997
13	0.0006	0.7463	0.9998	0.7004	0.8039	0.9996	0.9999
14	0.0036	0.7349	0.9988	0.7177	0.7567	0.9979	0.9991
15	0.0067	0.7349	0.9980	0.7465	0.7340	0.9960	0.9980
16	0.0106	0.7056	0.9969	0.7043	0.7117	0.9940	0.9970
17	0.0011	0.5996	0.9997	0.6145	0.5993	0.9994	0.9997
18	0.0054	0.7373	0.9983	0.7326	0.7449	0.9967	0.9984
19	0.0018	0.7942	0.9994	0.7821	0.8087	0.9989	0.9995
20	0.0017	0.6359	0.9995	0.6090	0.6694	0.9991	0.9996
21	0.0002	0.2174	0.9999	0.2034	0.2782	0.9999	1.000
22	0.0036	0.6510	0.9990	0.6493	0.6605	0.9981	0.9991
23	0.0001	0.0216	0.9999	0.0000	-	0.9999	1.000
24	0.0018	0.6433	0.9994	0.6226	0.6752	0.9989	0.9995
25	0.0016	0.6823	0.9994	0.6501	0.7272	0.9990	0.9996
26	0.0089	0.5654	0.9973	0.5648	0.5732	0.9948	0.9975
27	0.0020	0.5612	0.9993	0.5474	0.5917	0.9988	0.9995
28	0.0060	0.5899	0.9985	0.6110	0.5843	0.9969	0.9984
29	0.0001	0.0206	0.9999	0.0000	-	0.9999	1.000
30	0.0002	0.4417	0.9999	0.3968	0.5222	0.9999	1.000
31	0.0000	0.0335	1.000	0.0000	-	1.000	1.000
32	0.0074	0.6765	0.9978	0.6691	0.6965	0.9958	0.9980
33	0.0001	0.0381	0.9999	0.0000	-	0.9999	1.000

34	0.0001	0.0248	0.9999	0.0000	-	0.9999	1.000
35	0.0065	0.7550	0.9980	0.7514	0.7611	0.9962	0.9981
36	0.0001	0.0306	0.9999	0.0000	-	0.9999	1.000
37	0.0035	0.5740	0.9991	0.5718	0.5895	0.9982	0.9991
38	0.0005	0.6447	0.9998	0.5835	0.7522	0.9998	0.9999
39	0.0001	0.0238	0.9999	0.0000	-	0.9999	1.000
40	0.0046	0.7580	0.9987	0.7705	0.7514	0.9974	0.9986
41	0.0128	0.7204	0.9953	0.6877	0.7618	0.9923	0.9969
42	0.0001	0.0276	0.9999	0.0000	-	0.9999	1.000
43	0.0043	0.7814	0.9987	0.7751	0.7900	0.9975	0.9988
44	0.0004	0.4499	0.9998	0.4236	0.5057	0.9998	0.9999
45	0.0001	0.1988	0.9999	0.1465	0.4423	0.9999	1.000
46	0.0020	0.7396	0.9994	0.7305	0.7527	0.9988	0.9994
47	0.0044	0.7730	0.9985	0.7459	0.8035	0.9975	0.9989
48	0.0001	0.0126	0.9999	0.0000	-	0.9999	1.000
49	0.0083	0.8065	0.9977	0.8014	0.8147	0.9957	0.9979
50	0.0007	0.6181	0.9997	0.5721	0.6927	0.9995	0.9998
51	0.0002	0.0222	0.9998	0	-	0.9998	1.000
52	0.0066	0.8055	0.9979	0.7928	0.8201	0.9961	0.9982
53	0.0037	0.7783	0.9988	0.7642	0.7954	0.9978	0.9990

Figure A.4 shows the confusion matrices for the patients in Figure 4.6 used in the validation of the 13-fold cross-validation.

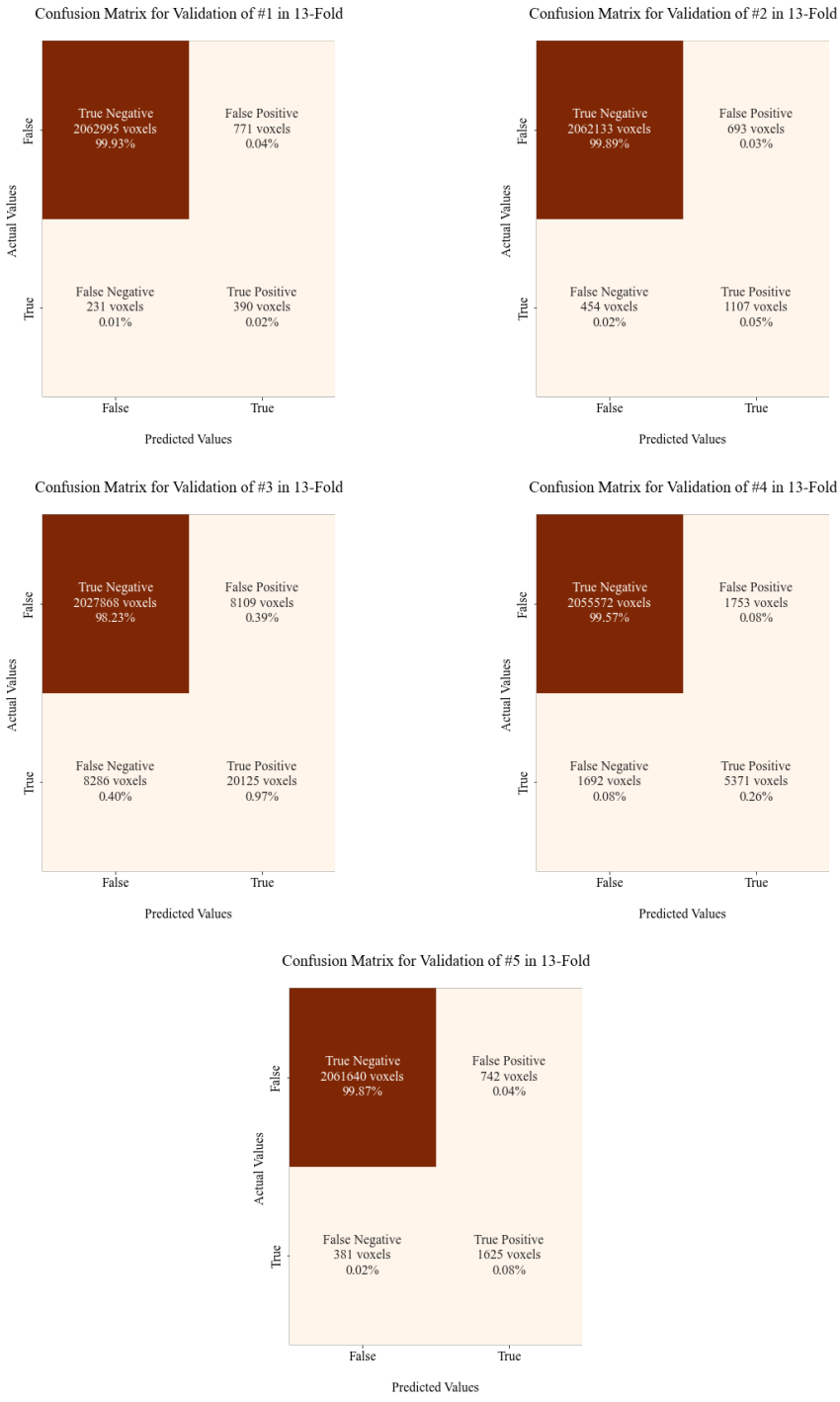


Figure A.4: The figure shows the confusion matrices for the patients in Figure 4.6

A.3.2 Testing of Model

Table A.3 shows the average values for the loss and dice score achieved from the testing of the 4-fold and 13-fold trained models on the testing data.

Table A.3: Average values for both loss and dice scores achieved from the testing of the different k-fold cross-validation methods implemented.

Cross-Validation	**Testing**	
k-Fold	Loss	Dice Score
4	0.0118	0.2880
13	0.0102	0.3183

Figure A.5 shows the boxplot of the dice scores from the five patients selected from the validation and testing-sets both for the 4- and 13-fold.

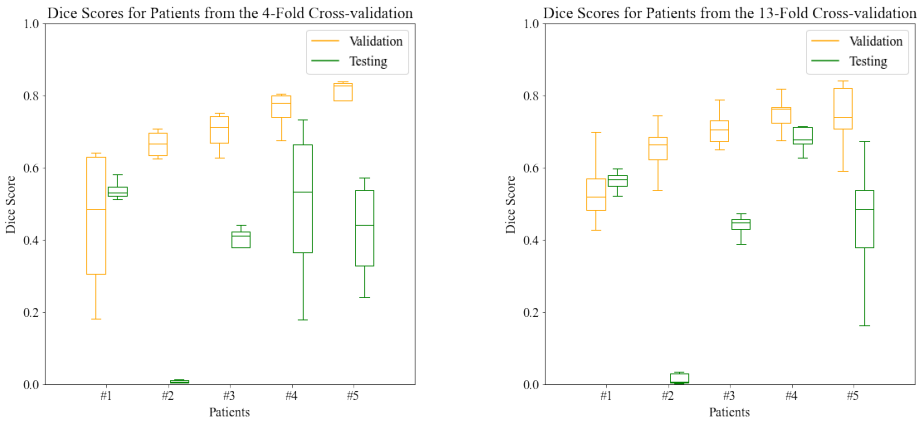
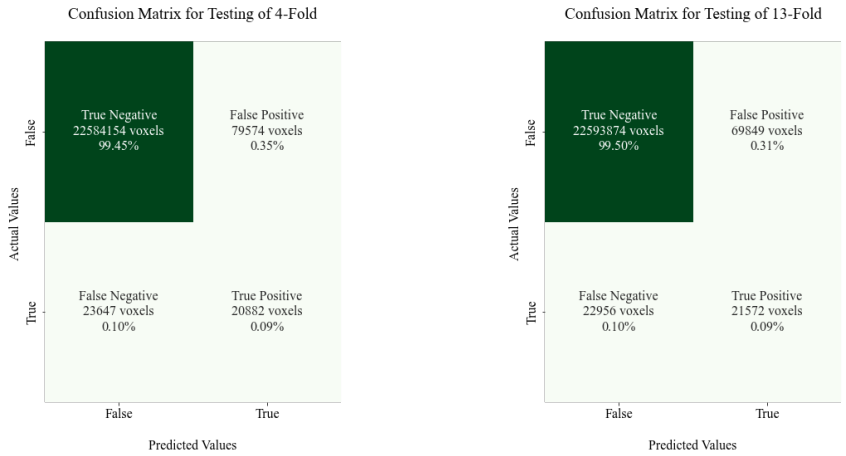


Figure A.5: The figure show the boxplot of the dice scores for the patients used as examples in the thesis for both the 4-fold and 13-fold.

The Figure A.6 shows the confusion matrices for the whole testing set both for the 4-fold and 13-fold trained models.



(a) **(b)**

Figure A.6: Confusion matrix for the testing data for the (a) 4-fold and (b) 13-fold cross-validation where the number of voxels and their respective percentages are classified as either TP, TN, FP, or FN.

4-Fold Cross-Validation

Table A.4 shows the average values for the evaluation metrics for the 11 patients in the testing-set for the 4-fold trained model.

Table A.4: Average values for all the evaluation metrics for all patients in the testing-set.

Evaluation Metrics for All Patients Used in the Testing of the 4-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
1	0.0239	0.5390	0.9982	0.7092	0.4359	0.9924	0.9942
2	0.0001	0.0086	0.9999	0.0000	0.0000	0.9999	1.000
3	0.0055	0.3897	0.9992	0.4515	0.3530	0.9982	0.9990
4	0.0050	0.4952	0.9998	0.6797	0.4126	0.9988	0.9991
5	0.0005	0.4246	0.9998	0.3963	0.5006	0.9998	0.9999
6	0.0001	0.0080	0.9999	0.0000	-	0.9999	1.000
7	0.0009	0.0026	0.9995	0.0000	-	0.9995	1.000
8	0.0011	0.3972	0.9999	0.4797	0.3602	0.9997	0.9998
9	0.0012	0.6884	0.9995	0.6121	0.7873	0.9993	0.9998
10	0.0753	0.1611	0.9658	0.0892	0.8265	0.9652	0.9993
11	0.0169	0.1135	0.9998	0.5215	0.0647	0.9973	0.9974

13-Fold Cross-Validation

Table A.5 shows the average values for the evaluation metrics for the 11 patients in the testing-set for the 13-fold trained model.

Table A.5: Average values for all the evaluation metrics for all patients in the testing-set for the 13-fold.

Evaluation Metrics for All Patients Used in the Testing of the 13-Fold

Patient	Loss	DS	Specificity	Precision	Recall	Acc	NPV
1	0.0206	0.5589	0.9981	0.7142	0.4619	0.9926	0.9945
2	0.0001	0.0173	0.9999	0.0000	0.0000	0.9999	1.000
3	0.0046	0.4357	0.9994	0.5199	0.3805	0.9984	0.9990
4	0.0034	0.6432	0.9997	0.7495	0.5839	0.9990	0.9994
5	0.0005	0.4462	0.9999	0.4600	0.4956	0.9998	0.9999
6	0.0001	0.0075	0.9999	0.0000	-	0.9999	1.000
7	0.0009	0.0058	0.9995	0.0000	-	0.9995	1.000
8	0.0007	0.5339	0.9999	0.5568	0.5462	0.9997	0.9999
9	0.0011	0.6754	0.9995	0.6062	0.7682	0.9993	0.9998
10	0.0601	0.1702	0.9705	0.0961	0.7632	0.9696	0.9990
11	0.0197	0.0959	0.9998	0.5245	0.0549	0.9972	0.9974

A.3.3 Counting Cancer Lesions

4-Fold Validation

Table A.6 shows all the counted lesions and the evaluation metrics for the 53 patients used in the validation of the 4-fold trained model.

Table A.6: The table shows the number of counted lesions for all patients used for validation in the 4-fold trained model.

Counting All Lesions in Patients Used in the Validation of the 4-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
1	6	4	1	5	0	4	1.0	0	0.83	0.17
2	4	3	1	3	0	3	1.0	0	0.75	0.25
3	55	53	5	50	11	42	0.79	0.21	0.91	0.090
4	9	13	0	9	3	10	0.77	0.23	1.0	0
5	1	1	0	1	0	1	1.0	0	1.0	0
6	2	2	0	2	0	2	1.0	0	1.0	0
7	13	8	5	8	0	8	1.0	0	0.62	0.38
8	9	9	0	9	0	9	1.0	0	1.0	0
9	3	4	1	2	2	2	0.50	0.50	0.67	0.33
10	46	36	15	31	4	32	0.89	0.11	0.67	0.33

11	3	3	0	3	0	3	1.0	0	1.0	0
12	16	17	2	14	5	12	0.71	0.29	0.88	0.12
13	3	2	1	2	0	2	1.0	0	0.67	0.33
14	24	21	4	20	1	20	0.95	0.050	0.83	0.17
15	15	15	3	12	6	9	0.60	0.40	0.80	0.20
16	85	81	15	70	11	70	0.86	0.14	0.82	0.18
17	8	9	1	7	2	7	0.78	0.22	0.88	0.12
18	31	28	5	26	2	26	0.93	0.070	0.84	0.16
19	5	3	0	5	1	2	0.67	0.33	1.0	0
20	12	9	3	9	0	9	1.0	0	0.75	0.25
34	14	0	14	0	0	0	-	-	0	1.0
35	45	36	9	36	2	34	0.94	0.060	0.80	0.20
36	3	0	3	0	0	0	-	-	0	1.0
37	21	19	5	16	5	14	0.74	0.26	0.76	0.24
38	1	1	0	1	0	1	1.0	0	1.0	0
39	3	0	3	0	0	0	-	-	0	1.0
40	19	18	3	16	1	17	0.94	0.060	0.84	0.16
41	44	37	9	35	1	36	0.97	0.030	0.80	0.20
42	14	0	14	0	0	0	-	-	0	1.0
43	7	7	1	6	2	5	0.71	0.29	0.86	0.14
44	25	2	24	1	1	1	0.50	0.50	0.040	0.96
45	3	1	2	1	0	1	1.0	0	0.33	0.67
46	3	3	0	3	0	3	1.0	0	1.0	0
47	13	5	7	6	2	3	0.60	0.40	0.46	0.54
48	8	0	8	0	0	0	-	-	0	1.0
49	14	9	5	9	0	9	1.0	0	0.64	0.36
50	21	4	18	3	1	3	0.75	0.25	0.14	0.86
51	17	0	17	0	0	0	-	-	0	1.0
52	22	9	10	12	0	9	1.0	0	0.55	0.45
53	28	11	17	11	0	11	1.0	0	0.39	0.61

4-Fold Testing

Table A.7 shows all the counted lesions and the evaluation metrics for the 11 patients used in the testing of the 4-fold trained model.

Table A.7: The table shows the number of counted lesions for all patients used for the testing of the 4-fold trained model.

Counting All Lesions in Patients Used in the Testing of the 4-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
1	53	63	6	47	28	35	0.56	0.44	0.89	0.11
2	61	0	61	0	0	0	-	-	0	1.0
3	70	69	27	43	27	42	0.61	0.39	0.61	0.39
4	5	8	0	5	4	4	0.50	0.50	1.0	0
5	7	2	6	1	1	1	0.50	0.50	0.14	0.86
6	10	0	10	0	0	0	-	-	0	1.0
7	12	0	12	0	0	0	-	-	0	1.0
8	1	2	0	1	1	1	0.50	0.50	1.0	0
9	8	6	2	6	0	6	1.0	0	0.75	0.25
10	25	5	20	5	0	5	1.0	0	0.20	0.80
11	28	20	12	16	11	9	0.45	0.55	0.57	0.43

13-Fold Validation

Table A.9 shows all the counted lesions and the evaluation metrics for the 53 patients used in the validation of the 13-fold trained model.

Table A.9: The table shows the number of counted lesions for all patients used for validation in the 13-fold trained model.

Counting All Lesions in Patients Used in the Validation of the 13-Fold

Patient	**AI**	**GT**	**FP_{AI}**	**TP_{AI}**	**FN_{GT}**	**TP_{GT}**	**TPR**	**FNR**	**PPV**	**FDR**
1	10	4	5	5	1	3	0.75	0.25	0.50	0.50
2	3	3	0	3	0	3	1.0	0.0	1.0	0.0
3	56	53	6	50	10	43	0.81	0.19	0.89	0.11
4	15	13	2	13	1	12	0.92	0.08	0.87	0.13
5	1	1	0	1	0	1	1.0	0.0	1.0	0.0
6	7	2	5	2	0	2	1.0	0	0.29	0.71
7	9	8	2	7	0	8	1.0	0	0.78	0.22
8	13	9	2	11	0	9	1.0	0	0.85	0.15
9	9	4	5	4	0	4	1.0	0	0.44	0.56
10	47	36	15	32	5	31	0.86	0.14	0.68	0.32

11	4	3	1	3	0	3	1.0	0	0.75	0.25
12	29	17	11	18	0	17	1.0	0	0.62	0.38
13	3	2	1	2	0	2	1.0	0	0.67	0.33
14	24	21	4	20	1	20	0.95	0.050	0.83	0.17
15	15	15	4	11	8	7	0.47	0.53	0.73	0.27
16	89	81	17	72	15	66	0.81	0.19	0.81	0.19
17	6	9	0	6	3	6	0.67	0.33	1.0	0
18	34	28	5	29	2	26	0.93	0.070	0.85	0.15
19	5	3	0	5	0	3	1.0	0	1.0	0
20	11	9	3	8	1	8	0.89	0.11	0.73	0.27
21	5	4	2	3	1	3	0.75	0.25	0.60	0.40
22	39	28	15	24	5	23	0.82	0.18	0.62	0.38
23	1	0	1	0	0	0	-	-	0	1.0
24	10	6	3	7	0	6	1.0	0	0.70	0.30
25	11	8	4	7	1	7	0.88	0.12	0.64	0.36
26	66	52	16	50	12	40	0.77	0.23	0.76	0.24
27	36	17	18	18	1	16	0.94	0.060	0.50	0.50
28	95	57	31	64	7	50	0.88	0.12	0.67	0.33
29	10	0	10	0	0	0	-	-	0	1.0
30	5	3	3	2	1	2	0.67	0.33	0.40	0.60
31	1	0	1	0	0	0	-	-	0	1.0
32	55	38	23	32	6	32	0.84	0.16	0.58	0.42
33	6	0	6	0	0	0	-	-	0	1.0
34	24	0	24	0	0	0	-	-	0	1.0
35	52	36	13	39	2	34	0.94	0.060	0.75	0.25
36	14	0	14	0	0	0	-	-	0	1.0
37	22	19	5	17	4	15	0.79	0.21	0.77	0.23
38	12	1	9	3	0	1	1.0	0	0.25	0.75
39	8	0	8	0	0	0	-	-	0	1.0
40	30	18	1	29	5	13	0.72	0.28	0.97	0.030
41	45	37	9	36	1	36	0.97	0.030	0.80	0.20
42	16	0	16	0	0	0	-	-	0	1.0
43	12	7	5	7	0	7	1.0	0	0.58	0.42
44	2	2	0	2	0	2	1.0	0	1.0	0
45	1	1	0	1	0	1	1.0	0	1.0	0
46	5	3	1	4	0	3	1.0	0	0.80	0.20
47	5	5	1	4	1	4	0.80	0.20	0.80	0.20
48	4	0	4	0	0	0	-	-	0	1.0
49	13	9	5	8	0	9	1.0	0	0.62	0.38
50	6	4	5	1	2	2	0.50	0.50	0.17	0.83
51	1	0	1	0	0	0	-	-	0	1.0

52	11	9	0	11	0	9	1.0	0	1.0	0
53	20	11	9	11	0	11	1.0	0	0.55	0.45

13-Fold Testing

Table A.10 shows all the counted lesions and the evaluation metrics for the 11 patients used in the testing of the 13-fold trained model.

Table A.10: The table shows the number of counted lesions for all patients used for the testing of the 13-fold trained model.

Counting All Lesions in Patients Used in the Testing of the 13-Fold

Patient	AI	GT	FP_{AI}	TP_{AI}	FN_{GT}	TP_{GT}	TPR	FNR	PPV	FDR
1	73	63	12	61	20	43	0.68	0.32	0.84	0.16
2	7	0	7	0	0	0	-	-	0.0	1.0
3	59	69	14	45	30	39	0.57	0.43	0.76	0.24
4	8	8	4	4	4	4	0.50	0.50	0.50	0.50
5	2	2	1	1	1	1	0.50	0.50	0.50	0.50
6	11	0	11	0	0	0	-	-	0	1.0
7	4	0	4	0	0	0	-	-	0	1.0
8	14	2	12	2	2	0	1.0	0	0.14	0.86
9	8	6	1	7	0	6	1.0	0	0.88	0.12
10	35	5	31	4	0	5	1.0	0	0.11	0.89
11	16	20	4	12	13	7	0.35	0.65	0.75	0.25

Appendix B

Appendix B

This appendix will present most of the code structures implemented for this thesis. The author used an existing network implementation and scripts that were written by Eivind Lysheim, a previous master student at NTNU [5]. The code structures below are modified to run with png RGB images as compared to the original NiFTI images. Additionally, several evaluation metrics have been implemented in order to get a better understand of how the model is training and performing.

B.1 MATLAB Code for RGB Images

The following code is an example of how the multi-modal 3-channel image was made, i.e., the RGB image. The reader will find both the code for how to create and normalize the RGB, and to how to overlay the mask on the RGB image.

```
clear all; close all;
dwi_img = single(niftiread('093_DWI.nii.gz'));
pet_img = single(niftiread('093_LYM.nii.gz'));
t2_img = single(niftiread('093_T2.nii.gz'));
seg = niftiread('093_Segmentation.nii');
seg_binarized = 256*mat2gray(seg);

%For Cancer free patients
blank_mask = zeros(size(dwi_img));
seg_binarized = 256*mat2gray(blank_mask);

% Finding the head of the images only (the body is removed)
head = single(dwi_img(:,:,142:155))+single(pet_img(:,:,142:155))+single(t2_img(:,:,142:155));

% Masking out the air, in other words making a mask of the head
mask = zeros(size(head));
mask(head>500) = 1;

%Plotting kernel of histogram to localize the landmark
figure();
subplot(1,3,1);
fit_dwi = histfit(dwi_img(find(mask)),[],'kernel');
plot(fit_dwi(1).XData, fit_dwi(1).YData);
title('DWI');
```

```
subplot(1,3,2);
fit_pet = histfit(pet_img(find(mask)),[],'kernel');
plot(fit_pet(1).XData, fit_pet(1).YData);
title('PET');

subplot(1,3,3);
fit_t2 = histfit(t2_img(find(mask)),[],'kernel');
plot(fit_t2(1).XData, fit_t2(1).YData);
title('T2');

%Landmarks
mu_d = [41.32,15.45,14.49,29.36,4.385,29.97,18.5,9.998,19.72,61.47,64.5,20.97,19.27,24.48,
12.89,11.14,24.56,20.35,64.96,16.38,10.77,150.2,52.21,59.55,49.59,47.06,40.5,38.25,51.25,
149.1,132.5,53.48,57.03,14.49,55.28,193.8,85.68,59.67,11.34,75.35,71.82,69.3,67.79,48.61,
73.65,55.19,71.57,8.865,56.15,96.83,8.453,82.21,82.88,46.05,86.85,17.43,15.08,65.97,85.4,
257.5,133.1,46.51,68,107.4,44.05];

mu_p = [356.2,1880,806.7,661.5,596.6,803,879.8,613.1,
656.8,3562,3444,1120,945,694.6,956.2,
176.9,399,920.5,1300,957.5,692.2,1120,1323,983.2,818,
527.9,795.2,4241,1049,2788,3214,357,
1646,1099,235.8,3007,716.9,725.4,260.1,2614,981.8,
913.5,1071,855.5,924.6,2137,689.5,2038,1452,687.4,539.9,947.8,1348,812.3,820.2,915,423.6,
847.8,933.8,2733,1314,3536,1082,19190,982.3];

mu_t = [160.9,253.8,278.5,230.3,264.5,231,198.7,310.3,298.9,195.4,738.7,224.4,231.8,228.4,216.4,
167.7,329.5,192.9,195.6,174.9,289.2,804.7,765.6,165.2,186,163.1,181.5,153.2,137.7,607,838.1,
196.6,138,221.9,230.8,760.4,208.5,207,248.1,183.4,
234.4,187,196.4,168.3,232.2,159.6,163.6,156.8,172,
250.7,254,162.2,194.8,183.5,145.9,244.6,269.8,161.7,
219.4,791.3,175,198.1,201,667,168.8];

mu_dwi = mean(mu_d);
mu_pet = mean(mu_p);
mu_t2 = mean(mu_t);

I_norm_dwi = (dwi_img.*(128))./(mu_dwi);
I_norm_pet = (pet_img.*(128))./(mu_pet);
I_norm_t2 = (t2_img.*(128))./(mu_t2);

pets = I_norm_pet./6;
t2s = I_norm_t2./4;
dwis = I_norm_dwi./20;

rgbim(:,:,:,1) = uint8(pets);
rgbim(:,:,:,2) = uint8(t2s);
rgbim(:,:,:,3) = uint8(dwis);

for k = 1:size(rgbim,2)
    currim = rot90(squeeze(rgbim(:,k,:,:)),1);
    imwrite(currim,sprintf('093_RGB_%u.png',k))
end

for k=1:size(seg_binarized,2)
    currim = rot90(squeeze(seg_binarized(:,k,:)),1);
    imwrite(currim,sprintf('093_Mask_%u.png',k));
end

%Testing and looking at RGB, mask and the overlay of mask on the RGB
figure();
```

```
rgb_t = imread('093_RGB_65.png');
imshow(rgb_t);

figure();
seg_t = imread('093_Mask_65.png');
imshow(seg_t);

figure();
A = labeloverlay(rgb_t, seg_t,'Colormap','white','Transparency',0.25);
imshow(A);
```

B.2 Imported Functions and Libraries

```
#All libraries and imported data needed for training and testing

from skimage import data, transform
from skimage.util import random_noise
from skimage import exposure
from skimage import util
import natsort

import scipy
import os
import numpy as np
import matplotlib.pyplot as plt
from glob import glob
import random
from zipfile import ZipFile

import matplotlib.pyplot as plt
from tqdm import tqdm_notebook
from skimage.io import imread, imshow, concatenate_images
from skimage.transform import resize
import tensorflow as tf

from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras import models
from tensorflow.keras import optimizers
from tensorflow.keras import callbacks
from tensorflow.keras import metrics
from tensorflow.keras.callbacks import ModelCheckpoint, CSVLogger, TensorBoard

from skimage.transform import rotate, AffineTransform,resize
from skimage import filters,color, transform,exposure
from skimage.util import random_noise
from scipy import ndimage
import random
import matplotlib.pyplot as plt
from skimage.io import imread, imshow, concatenate_images,imsave
from skimage.morphology import label
from sklearn.model_selection import train_test_split
from PIL import Image, ImageEnhance
from PIL import ImageOps

import gc
from glob import glob
```

```
import tensorflow as tf
import tensorflow_addons as tfa
import datetime, os
import matplotlib.pyplot as plt
import nibabel as nib
import matplotlib.pyplot as plt
from skimage.transform import resize
from PIL import Image
import elasticdeform
import sys
import os
import warnings
if not sys.warnoptions:
    warnings.simplefilter("ignore")
os.environ['TF_CPP_MIN_LOG_LEVEL'] = '3'
import sklearn
from skimage.transform import rotate, AffineTransform,resize
import skimage
from skimage.util import random_noise
from scipy import ndimage
import random
import matplotlib.pyplot as plt
from skimage.io import imread, imshow, concatenate_images,imsave
from skimage.morphology import label
from sklearn.model_selection import train_test_split
from PIL import Image, ImageEnhance
import scipy
from keras.preprocessing.image import ImageDataGenerator, array_to_img, img_to_array, load_img
```

B.3 Pre-Processing of Data

This section shows the code used for the pre-processing of data in Python.

```
X_RGB = np.zeros((len(RGB_TRAIN), 155, 128, 3), dtype=np.float32)
Y_Lesion = np.zeros((len(Lesions_TRAIN), 155, 128, 1), dtype=np.float32)

for n, id_ in tqdm_notebook(enumerate(RGB_TRAIN), total=len(RGB_TRAIN)):
    # Load images
    img = load_img(id_)
    x_img = img_to_array(img)
    x_img = resize(x_img, (155, 128, 3), mode = 'constant', preserve_range = True)
    X_RGB[n] = x_img/255.0

for n, id_ in tqdm_notebook(enumerate(Lesions_TRAIN), total=len(Lesions_TRAIN)):
    #Load masks
    mask = img_to_array(load_img(id_))
    mask = resize(mask, (155, 128, 1), mode = 'constant', preserve_range = True)
    # Save images
    Y_Lesion[n] = mask/255.0

#Add all images and masks to a list
#Make python lists to store data

X_TRAIN_RGB = []
Y_TRAIN_Lesions = []

i = 0
print("Start")
```

```
#for RGB, Lesions in test:
for RGB, Lesions in zip(X_RGB,Y_Lesion):
    new_img_rgb = crop_rgb(RGB)
    new_lesions = crop_lesions(Lesions)

    #Add images into a long array
    add_image(new_img_rgb, X_TRAIN_RGB)
    add_image(new_lesions, Y_TRAIN_Lesions)
    #print("-----Agumentation-----")

    #Rotation#
    for x in range(0,1):
        new_img_rgb_rot, new_lesion_rot = random_rotate(new_img_rgb,new_lesions)
        add_image(new_img_rgb_rot,X_TRAIN_RGB)
        add_image(new_lesion_rot,Y_TRAIN_Lesions)

    #Flip
    img_rgb_updown = np.fliplr(new_img_rgb)
    img_lesion_updown = np.fliplr(new_lesions)
    add_image(img_rgb_updown, X_TRAIN_RGB)
    add_image(img_lesion_updown, Y_TRAIN_Lesions)

    #Noise
    for x in range(0,1):
        new_rgb_noise = add_random_noise(new_img_rgb)
        add_image(new_rgb_noise,X_TRAIN_RGB)
        add_image(new_lesions, Y_TRAIN_Lesions)

    #Blur
    for x in range(0,1):
        new_rgb_blur = image_blur(new_img_rgb)
        add_image(new_rgb_blur, X_TRAIN_RGB)
        add_image(new_lesions, Y_TRAIN_Lesions)

    #Contrast
    for x in range(0,1):
        new_rgb_contrast = improve_contrast(new_img_rgb)
        add_image(new_rgb_contrast, X_TRAIN_RGB)
        add_image(new_lesions, Y_TRAIN_Lesions)

    print(i)
    i += 1

#convert list to numpy array
X_TRAIN_RGB = np.asarray(X_TRAIN_RGB, dtype =np.float32)
Y_TRAIN_Lesions = np.asarray(Y_TRAIN_Lesions, dtype= np.float32)

#Check array shapes
print(X_TRAIN_RGB.shape)
print(Y_TRAIN_Lesions.shape)

#Delete unused lists, this will free memory
del new_img_rgb
del new_lesions

gc.collect()
print('Done')
```

B.4 2D U-Net

This sections presents the 2D U-Net code and architecture used to train the model in addition to some functions necessary for later use.

```
#UNET functions
import tensorflow as tf
from tensorflow.keras import metrics
from tensorflow import keras
from tensorflow.keras import layers
from tensorflow.keras import models
from tensorflow.keras import optimizers
from tensorflow.keras import callbacks
from tensorflow.keras import metrics
from tensorflow.keras.callbacks import ModelCheckpoint, CSVLogger, TensorBoard

from tensorflow.keras import backend as K
def dice_coeff(y_true, y_pred, smooth=1.):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    return (2. * intersection + smooth) / (K.sum(y_true_f) + K.sum(y_pred_f) +
    smooth)

def dice_coeff_loss(y_true, y_pred):
    return 1-dice_coeff(y_true, y_pred)

def CE_DL_loss(y_true, y_pred):
    def dice_loss(y_true, y_pred):
      y_pred = tf.math.sigmoid(y_pred)
      numerator = 2 * tf.reduce_sum(y_true * y_pred)
      denominator = tf.reduce_sum(y_true + y_pred)
      return 1 - numerator / denominator

    y_true = tf.cast(y_true, tf.float32)
    o = tf.nn.sigmoid_cross_entropy_with_logits(y_true, y_pred)
    + dice_loss(y_true, y_pred)
    return tf.reduce_mean(o)


#Filters, x-dim, y-dim and channels
def U_NET(filter1,x,y,channels):
    inputs = keras.Input(shape=(x,y,channels))

    conv1 = layers.Conv2D(filter1, (3, 3), activation='relu',
    padding='same')(inputs)
    conv1 = layers.BatchNormalization()(conv1)
    conv1 = layers.Dropout(0.1)(conv1)
    conv1 = layers.Conv2D(filter1, (3, 3), activation='relu', padding='same')(conv1)
    conv1 = layers.BatchNormalization()(conv1)
    pool1 = layers.MaxPooling2D(pool_size=(2, 2))(conv1)
    pool1 = layers.Dropout(0.10)(pool1)

#filter2 = 32
    conv2 = layers.Conv2D(2*filter1, (3, 3), activation='relu',
    padding='same')(pool1)
    conv2 = layers.BatchNormalization()(conv2)
    conv2 = layers.Dropout(0.1)(conv2)
    conv2 = layers.Conv2D(2*filter1, (3, 3), activation='relu',
    padding='same')(conv2)
```

```
    conv2 = layers.BatchNormalization()(conv2)
    pool2 = layers.MaxPooling2D(pool_size=(2, 2))(conv2)
    pool2 = layers.Dropout(0.10)(pool2)

#filter3 = 64
    conv3 = layers.Conv2D(2*2*filter1, (3, 3), activation='relu',
    padding='same')(pool2)
    conv3 = layers.BatchNormalization()(conv3)
    conv3 = layers.Dropout(0.2)(conv3)
    conv3 = layers.Conv2D(2*2*filter1, (3,3), activation='relu',
    padding='same')(conv3)
    conv3 = layers.BatchNormalization()(conv3)
    pool3 = layers.MaxPooling2D(pool_size=(2, 2))(conv3)
    pool3 = layers.Dropout(0.2)(pool3)

#filter4 = 128
    conv4 = layers.Conv2D(2*2*2*filter1,(3, 3), activation='relu',
    padding='same')(pool3)
    conv4 = layers.BatchNormalization()(conv4)
    pool4 = layers.Dropout(0.2)(conv4)
    conv4 = layers.Conv2D(2*2*2*filter1, (3, 3), activation='relu',
    padding='same')(conv4)
    conv4 = layers.BatchNormalization()(conv4)
    conv4 = layers.Dropout(0.3)(conv4)
    pool4 = layers.MaxPooling2D(pool_size=(2, 2))(conv4)
    pool4 = layers.Dropout(0.20)(pool4)

#filter5 = 256
    conv5 = layers.Conv2D(2*2*2*2*filter1, (3, 3), activation='relu',
    padding='same')(pool4)
    conv5 = layers.BatchNormalization()(conv5)
    conv5 = layers.Dropout(0.3)(conv5)
    conv5 = layers.Conv2D(2*2*2*2*filter1, (3, 3), activation='relu',
    padding='same')(conv5)
    conv5 = layers.BatchNormalization()(conv5)
    conv5 = layers.Dropout(0.2)(conv5)

########################################
#EXPANSIVE PATH
########################################

#filter6 = 128
    up6 = layers.Conv2DTranspose(2*2*2*filter1, (3, 3), activation="relu",
    strides = (2,2), padding="same")(conv5)
    up6 = layers.BatchNormalization()(up6)
    up6 = layers.concatenate([up6,conv4])
    conv6 = layers.Conv2D(2*2*2*filter1, (3, 3), activation='relu',
    padding='same')(up6)
    conv6 = layers.BatchNormalization()(conv6)
    conv6 = layers.Dropout(0.2)(conv6)
    conv6 = layers.Conv2D(2*2*2*filter1, (3, 3), activation='relu',
    padding='same')(conv6)
    conv6 = layers.BatchNormalization()(conv6)

#filter7 = 64
    up7 = layers.Conv2DTranspose(2*2*filter1, (3, 3), activation="relu",
    strides = (2,2), padding="same")(conv6)
    up7 = layers.BatchNormalization()(up7)
    up7 = layers.concatenate([up7,conv3])
    conv7 = layers.Conv2D(2*2*filter1, (3, 3), activation='relu',
```

```
    padding='same')(up7)
    conv7 = layers.BatchNormalization()(conv7)
    conv7 = layers.Dropout(0.2)(conv7)
    conv7 = layers.Conv2D(2*2*filter1, (3, 3), activation='relu',
    padding='same')(conv7)
    conv7 = layers.BatchNormalization()(conv7)

#filter8 = 32
    up8 = layers.Conv2DTranspose(2*filter1, (3, 3), activation="relu",
    strides = (2,2), padding="same")(conv7)
    up8 = layers.BatchNormalization()(up8)
    up8 = layers.concatenate([up8,conv2])
    conv8 = layers.Conv2D(2*filter1, (3, 3), activation='relu',
    padding='same')(up8)
    conv8 = layers.BatchNormalization()(conv8)
    conv8 = layers.Dropout(0.1)(conv8)
    conv8 = layers.Conv2D(2*filter1, (3, 3), activation='relu',
    padding='same')(conv8)
    conv8 = layers.BatchNormalization()(conv8)

#filter9 = 16
    up9 = layers.Conv2DTranspose(filter1, (3, 3), activation="relu",
    strides = (2,2), padding="same")(conv8)
    up9 = layers.BatchNormalization()(up9)
    up9 = layers.concatenate([up9,conv1], axis=3)
    up9 = layers.Dropout(0.2)(up9)
    conv9 = layers.Conv2D(filter1, (3, 3), activation='relu',
    padding='same')(up9)
    conv9 = layers.BatchNormalization()(conv9)
    conv9 = layers.Conv2D(filter1, (3, 3), activation='relu',
    padding='same')(conv9)
    conv9 = layers.BatchNormalization()(conv9)

    outputs = layers.Conv2D(1, ( 1, 1), activation='sigmoid')(conv9)

    model = tf.keras.Model(inputs=[inputs], outputs=[outputs])
    return model
```

B.5 Training

The following code shows the training:

```
#Load 2D U-Net from function
model_RGB = U_NET(16,144,128,3) #Channel = 3 due to RGB has 3 channels
#default threshold = 0.5 for metrics

model_RGB.compile(optimizer="rmsprop", loss="binary_crossentropy",
metrics=[dice_coeff,dice_coeff_loss,"accuracy","AUC","TruePositives","TrueNegatives",
"FalsePositives","FalseNegatives",tf.keras.metrics.Recall(thresholds=0.2),"Precision",
tf.keras.metrics.MeanIoU(num_classes=2)])

#Print information about U-NET - dimensions etc
model_RGB.summary(line_length=120)

#Model weights (used when performing k-fold validation)
model_RGB.save_weights('reset.h5')
model_RGB.load_weights('reset.h5')
```

```
factor = 6*112 # Number of images of each patient when data augmentation is performed
number_patients = int(len(X_TRAIN_RGB)/factor)

# ---- 13-Fold Cross-Validation ----
for x in range(0,13):
    print("-----Training␣for␣for␣fold␣", x, "␣-----")
    model_RGB.load_weights('reset.h5') #Resetting weights
    print("loop:␣",x)
    pic_ = int(x * factor*13)
    print("pic_␣",pic_)
    pic = pic_ + factor*13
    print("pic␣", pic)

    x_val = X_TRAIN_RGB[pic_:pic]
    print("x_val␣",x_val.shape)

    y_lesions_val = Y_TRAIN_Lesions[pic_:pic]
    print("y_lesion_val␣",y_lesions_val.shape)

    x_train = np.delete(X_TRAIN_RGB,slice(pic_,pic),axis=0)
    print("x_train␣",x_train.shape)

    y_lesions_train = np.delete(Y_TRAIN_Lesions,slice(pic_,pic),axis=0)
    print("y_lesion_train␣",y_lesions_train.shape)

    ###################################
    # Define data loaders.
    #Pre-fetches a bunch of images and expands the dimensions on the fly during training.
    #This is to reduce the memory used

    train_loader = tf.data.Dataset.from_tensor_slices((x_train, y_lesions_train))
    validation_loader = tf.data.Dataset.from_tensor_slices((x_val, y_lesions_val))
    print("train␣dataset")
    batch_size = 30

    # Augment the on the fly during training.
    train_dataset = (
        train_loader.shuffle(len(x_train))
        .batch(batch_size)
        .prefetch(30)
        )

    print("Validation␣dataset")
    # Only rescale.
    validation_dataset = (
        validation_loader.shuffle(len(x_val))
        .batch(batch_size)
        .prefetch(30)
        )

    #### Training #####
    import os
    #Defines where the weights will be stored
    output_directory = "/home/solvikn/LYMPHOMA-DATA/Master/weights4w/"

    #Save model for each k-fold - fold 0 gives 0 in the .h5 etc.
    mcp_save = ModelCheckpoint(os.path.join(output_directory,"model_best"+str(x)+".h5"),
    save_best_only=True)

    #Train network for 30 epochs (an example)
```

```
    epochs = 30
    results = model_RGB.fit(train_dataset,validation_data=validation_dataset,epochs=epochs,
    shuffle=True,verbose=2, callbacks=[mcp_save])

    ##Example for plotting DS learning curve ##
    plt.figure(figsize=(8, 8))
    plt.title("Dice␣Score␣Learning␣Curve")
    plt.plot(results.history["dice_coeff"], label = "Training")
    plt.plot(results.history["val_dice_coeff"], label = "Validation")
    plt.plot( np.argmax(results.history["val_dice_coeff"]), np.max(results.history["val_dice_coeff"]),
    marker="x", color="r", label="Best␣model")
    plt.xlabel("Epochs")
    plt.ylabel("Dice␣Score")
    plt.legend();
    plt.show()
    print(np.max(results.history['val_dice_coeff']))
```

B.6 Results and Post-Processing

This sections provides the code used in the post-processing of the results:

```
#Load weigths
import natsort

vekter = sorted(glob("/home/solvikn/LYMPHOMA-DATA/Master/weights_4fold/*.h5"))
#print(vekter)

#Load images to make predictions on
RGB_TRAIN =natsort.natsorted(glob("/home/solvikn/LYMPHOMA-DATA/Master/TRAIN/085/RGB/*.png*"),)

#Load lesions - will be used to evaluate the performance of the network
Lesions_TRAIN = natsort.natsorted(glob("/home/solvikn/LYMPHOMA-DATA/Master/TRAIN/085/Mask/*.png*"))

X_RGB = np.zeros((len(RGB_TRAIN), 155, 128, 3), dtype=np.float32)
Y_Lesion = np.zeros((len(Lesions_TRAIN), 155, 128, 1), dtype=np.float32)

for n, id_ in tqdm_notebook(enumerate(RGB_TRAIN), total=len(RGB_TRAIN)):
    # Load images
    img = load_img(id_)
    x_img = img_to_array(img)
    x_img = resize(x_img, (155, 128, 3), mode = 'constant', preserve_range = True)
    X_RGB[n] = x_img/255.0

for n, id_ in tqdm_notebook(enumerate(Lesions_TRAIN), total=len(Lesions_TRAIN)):
    #Load masks
    mask = img_to_array(load_img(id_))
    mask = resize(mask, (155, 128, 1), mode = 'constant', preserve_range = True)
    # Save images
    Y_Lesion[n] = mask/255.0

#Make lists of RGB images and lesions
X_TEST_RGB =[]
Y_TEST_LESIONS = []
i = 0

for RGB, Lesions in zip(X_RGB,Y_Lesion):
    new_img_rgb = crop_rgb(RGB)
    new_lesions = crop_lesions(Lesions)
```

```
    #Decompose the 3D images into a long array of 2D images

    add_image(new_img_rgb, X_TEST_RGB)
    add_image(new_lesions, Y_TEST_LESIONS)
    print("new_img_rgb max: ", np.max(new_img_rgb))

    print(i)
    i += 1

X_TEST_RGB = np.asarray(X_TEST_RGB, dtype = np.float32)
Y_TEST_LESIONS = np.asarray(Y_TEST_LESIONS, dtype = np.float32)

dice_score_RGB = []
dice_loss_RGB = []
sensitivity_ = []
specificity_ = []
precision = []
recall_ = []
accuracy_ = []
loss_ = []
NPV = []
FPR = []
TPR = []
TN = []
TP = []
FN = []
FP = []

for i in range(0,len(vekter)):
    print("----------New Image----------")
    print("Round ", i)
    print(vekter[i])
    #print("----------New Image----------")

    height_ = 0
    print("heigth_ ", height_)
    height = 112 #112 coronal slices
    print("height ", height)

    model_RGB = U_NET(16,144,128,3)
    model_RGB.compile(optimizer="rmsprop", loss="binary_crossentropy",
    metrics=[dice_coeff,"accuracy"])
    model_RGB.load_weights(vekter[i])

    #----------------- Testing RGB Image --------------------#

    X_TEST_RGB__ = X_TEST_RGB[height_:height,:,:,:]
    print("X_TEST_RGB__ ",X_TEST_RGB__.shape)
    preds_test_RGB = model_RGB.predict(X_TEST_RGB__, batch_size = 1, verbose = 2)
    print("preds_test_RGB ", preds_test_RGB.shape)

    preds_test_RGB_t = (preds_test_RGB).astype(np.float32)
    preds_test_RGB_t = preds_test_RGB_t[:,:,:,0]
    print("preds_test_RGB_t ",preds_test_RGB_t.shape)

    Y_TEST_LESIONS_ = Y_TEST_LESIONS[height_:height,:,:,0]
    X_TEST_RGB_ = X_TEST_RGB[height_:height,:,:,0]

    X_TEST_RGB_unormalized_ = X_TEST_RGB_unormalized[height_:height,:,:]
```

```
    print("XTEST_RGB␣unorm␣shape",X_TEST_RGB_unormalized_.shape)

    dice_RGB = dice_coeff(Y_TEST_LESIONS,preds_test_RGB)
    print("dicescore␣RGB:␣", dice_RGB)
    dice_score_RGB.append(dice_RGB)

    Loss = BinaryCrossEntropy(Y_TEST_LESIONS,preds_test_RGB)
    loss_.append(Loss)

    #spe_RGB = specificity(Y_TEST_LESIONS,preds_test_RGB)
    #specificity_.append(spe_RGB)

    prec,recall,acc,tp,fp,fn,tn,npv,sen,spe,fpr,tpr = confusion(Y_TEST_LESIONS,preds_test_RGB)
    precision.append(prec)
    recall_.append(recall)
    accuracy_.append(acc)
    TP.append(tp)
    FP.append(fp)
    FN.append(fn)
    TN.append(tn)
    NPV.append(npv)
    sensitivity_.append(sen)
    specificity_.append(spe)
    FPR.append(fpr)
    TPR.append(tpr)

    #---------

    #Save masks as NIFTI:
    nib.Nifti1Image(preds_test_RGB,affine=np.eye(4,4))
    nib.save(new_preds,"Pred4_085.nii.gz")
    maske = nib.Nifti1Image(Y_TEST_LESIONS,affine=np.eye(4,4))
    nib.save(maske,"Maske4_085.nii.gz")

print("---------Summary---------")
print("Lossscore_RGB=␣", loss_)
print("Lossaverage_RGB=␣", np.average(loss_))
print("{:.4f}".format(np.average(loss_)))
print()
print("dicescore_RGB=␣", dice_score_RGB)
print("diceaverage_RGB=␣", np.average(dice_score_RGB))
print("{:.4f}".format(np.average(dice_score_RGB)))
print()
print("sensitivity_RGB=␣", sensitivity_)
print("sensitivity_average_RGB=␣", np.average(sensitivity_))
print("{:.4f}".format(np.average(sensitivity_)))
print()
print("specificity_RGB=␣", specificity_)
print("specificity_average_RGB=␣", np.average(specificity_))
print("{:.4f}".format(np.average(specificity_)))
print()
print("precision_RGB=␣", precision)
print("precision_average_RGB=␣", np.average(precision))
print("{:.4f}".format(np.average(precision)))
print()
print("recall_RGB=␣", recall_)
print("recall_average_RGB=␣", np.average(recall_))
print("{:.4f}".format(np.average(recall_)))
print()
print("accuracy_RGB=␣", accuracy_)
```

```
print("accuracy_average_RGB=␣", np.average(accuracy_))
print("{:.4f}".format(np.average(accuracy_)))
print()
print("NPV_RGB=␣", NPV)
print("NPV_average_RGB=␣", np.average(NPV))
print("{:.4f}".format(np.average(NPV)))
print()
print("TN_RGB=␣", TN)
print("TN_average_RGB=␣", np.average(TN))
print("{:.4f}".format(np.average(TN)))
print()
print("TP_RGB=␣", TP)
print("TP_average_RGB=␣", np.average(TP))
print("{:.4f}".format(np.average(TP)))
print()
print("FN_RGB=␣", FN)
print("FN_average_RGB=␣", np.average(FN))
print("{:.4f}".format(np.average(FN)))
print()
print("FP_RGB=␣", FP)
print("FP_average_RGB=␣", np.average(FP))
print("{:.4f}".format(np.average(FP)))
```

B.7 Helper Functions

This sections shows additional functions used in the data augmentation and other useful functions called in the code above.

```
#Read image
def read_png(file):
    img = np.array(Image.open(file))
    return img

#Plot image
def plot_image(file_):
    plt.imshow((file_))
    plt.axis("Off")
    plt.show()

def crop_rgb(RGB):
    resized_RGB = RGB[6:150,0:128,0:3]
    return resized_RGB

def crop_lesions(Lesions):
    resized_lesion = Lesions[6:150,0:128]
    return resized_lesion

#Add images to list, needed when doing data augmentation
def add_image(img, list_):
    list_.append(img)
    return list_

######Data augmentation######
def random_rotate(image_1,label):
    angle = random.randint(0,360)
    image_1 = ndimage.rotate(image_1,angle,reshape=False)
    label = ndimage.rotate(label,angle,reshape=False)
```

```
    return image_1,label

def add_random_noise(image):
    image_ = np.array(image)
    noise = random.randint(0,5) / 100
    img_noise = random_noise(image_, var=noise**2)
    return img_noise

def improve_contrast(image):
    image_ = np.array(image)
    x = random.randint(0,10) / 10
    v_min, v_max = np.percentile(image_, (x, 99.8))
    img_contrast = exposure.rescale_intensity(image_, in_range=(v_min, v_max))
    return img_contrast

def image_blur(image_):
    image = np.array(image_)
    img_blur = ndimage.uniform_filter(image, size=(3,3,3))
    return img_blur

#Loss
def BinaryCrossEntropy(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    y_pred_f = K.clip(y_pred_f, K.epsilon(), 1 - K.epsilon())
    term_0 = (1 - y_true_f) * K.log(1 - y_pred_f + K.epsilon())
    term_1 = y_true_f * K.log(y_pred_f + K.epsilon())
    return -K.mean(term_0 + term_1, axis=0)

#Confusion matrix
def confusion(y_true, y_pred):
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    smooth = 1
    y_pred_pos = K.clip(y_pred_f, 0, 1)
    y_pred_neg = 1 - y_pred_pos
    y_pos = K.clip(y_true_f, 0, 1)
    y_neg = 1 - y_pos
    tp = K.sum(y_pos * y_pred_pos)
    fp = K.sum(y_neg * y_pred_pos)
    fn = K.sum(y_pos * y_pred_neg)
    tn = K.sum(y_neg * y_pred_neg)
    prec = (tp ) / (tp + fp )
    recall = (tp ) / (tp + fn )
    acc = (tp + tn )/(tp + fn +tn + fp )
    npv = tn/(tn+fn)
    sen = 1-(fn/(fn+tp))
    spe = tn/(tn+fp)
    fpr = fp/(fp+tn)
    tpr = tp/(tp+fn)
    return prec, recall, acc, tp, fp, fn, tn, npv, sen, spe, fpr, tpr
```

B.8 Code for Counting Cancer Lesions

This section shows the code in MATLAB to count the cancer lesions in both the ground truth and the predicted masks.

```
Mask = niftiread('Maske4_085.nii.gz');
GT = imbinarize(Mask);
AI = imbinarize(niftiread('Pred4_085.nii.gz'));

gt = bwconncomp(GT);
ai = bwconncomp(AI);

all_lesions_GT = gt.NumObjects;
all_lesions_AI = ai.NumObjects;

TP_gt = 0;
FN_gt = 0;

for idx=1:all_lesions_GT
    ind = cell2mat(gt.PixelIdxList(idx));
    t = sum(GT(ind).*AI(ind));
    if t > 0
        TP_gt = TP_gt+1;
    elseif t == 0
        FN_gt = FN_gt+1;
    end
end

TP_ai = 0;
FP_ai = 0;

for idx_ai=1:all_lesions_AI
    ind_ai = cell2mat(ai.PixelIdxList(idx_ai));
    t_ai = sum(AI(ind_ai).*GT(ind_ai));
    if t_ai > 0
        TP_ai = TP_ai+1;
    elseif t_ai == 0
        FP_ai = FP_ai+1;
    end
end

EN = all_lesions_AI
AN = all_lesions_GT
FP_ai
TP_ai
FN_gt
TP_gt
TPR_gt = round(TP_gt/AN,2)
FPR_gt = round(1 - TPR_gt,2)
Precision = round(TP_ai/EN,2)
FDR_ai = round(FP_ai/EN,2)
```

Sølvi Knapstad

Automated Segmentation of Metastatic Lymph Nodes in Lymphoma Patients

NTNU
Norwegian University of
Science and Technology