A study of machine learning and deep learning methods and their application to medical imaging

David Wallis

université PARIS-SACLAY

A study of machine learning and deep learning methods and their application to medical imaging

Une étude des méthodes d'apprentissage automatique et d'apprentissage profond et de leur application à l'imagerie médicale

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°575 : Electrical, optical, bio : Physics and engineering (EOBE)
Spécialité de doctorat : Imagerie et physique médicale
Unité de recherche : Laboratoire d'Imagerie Translationnelle en Oncologie (LITO)
U1288 Inserm/Institut Curie
Référent : Faculté des sciences d'Orsay

Thèse présentée et soutenue à Paris-Saclay, le 01 juin 2021, par

David WALLIS

Composition du Jury

Ronald BOELLAARD Professeur, UMCG, University of Groningen	Président
Johan NUYTS Professeur, NMMI, KU Leuven	Rapporteur & Examinateur
Julia SCHNABEL Professeure, BMEIS, KCL London	Rapporteuse & Examinatrice
Antoine GRIGIS Ingénieur de recherche, Neurospin, Université Paris-Saclay	Examinateur

Direction de la thèse

Irène BUVAT Directrice de recherche, CNRS, Université Paris-Saclay	Directrice de thèse

Thèse de doctorat

NNT : 2021UPAST057

Synthèse en Français

L'analyse automatique des images médicales est un domaine de recherche extrêmement dynamique. L'objectif de ce doctorat était d'explorer ce domaine, en utilisant le *machine learning* et le *deep learning* pour résoudre des problèmes d'imagerie médicale clinique. Voici un bref résumé des chapitres présentant les travaux originaux réalisés.

Chapitre 5 - Détection automatisée des ganglions lymphatiques médiastinaux pathologiques avec un modèle de deep learning Dans le chapitre 5, nous utilisons les réseaux de neurones convolutionnels (CNNs) pour automatiser la détection des ganglions lymphatiques médiastinaux dans les images TEP/TDM obtenues avec du fluorodeoxyglucose (FDG). Il s'agit d'une étape importante pour la bilan d'extension du cancer du poumon, mais elle est associée à une erreur élevée, même pour les experts (sensibilité et spécificité de 0,59 et 0,97 respectivement [1]). Ce chapitre, qui constitue notre première grande tentative d'application du deep learning aux problèmes médicaux, décrit aussi la progression de nos recherches. Notre ensemble de données se composait de 125 images TEP/TDM au FDG du corps entier, incluant 172 ganglions médiastinaux pathologiques identifiés par un médecin. Nous ne nous intéressons qu'à la région médiastinale, donc nous construisons d'abord un algorithme pour sélectionner uniquement la région thoracique. Nous testons ensuite plusieurs méthodes pour identifier les ganglions, en discutant en détail de ce qui a fonctionné et de ce qui n'a pas fonctionné. Le modèle final est un processus en deux phases. Dans la première phase, les régions candidates sont trouvées en utilisant un réseau U-Net pour segmenter les régions coupe par coupe. Dans la deuxième phase, ces régions sont divisées en cubes 3D qui se chevauchent et qui sont classés comme positifs ou négatifs. Ces classifications sont utilisées pour créer une carte des ganglions pathologiques prédits pour chaque patient. Par rapport aux prédictions du médecin, notre modèle atteint une sensibilité de 0,87 (intervalle de confiance à 95% [0,74 à 0,94]) et une spécificité de 0,94 [0,90 à 0,97], avec 0,40 [0,21 à 0,68] faux positifs par patient. Comme mentionné précédemment, il y a une erreur élevée associée aux prédictions des médecins, ainsi, nous ne disposons pas d'une référence exacte. Il est donc approprié de comparer la différence entre les prédictions de notre modèle et

celles du médecin à la différence entre les prédictions du médecin et celles d'un second médecin. Nous avons donc fait labelliser l'ensemble de tests par un second médecin. Le score kappa entre les résultats de notre modèle et les prédictions du médecin était 0,75 [0,66 à 0,85], comparé à 0,67 [0,52 à 0,81] entre les deux médecins. Ces résultats montrent que notre modèle peut détecter avec succès des ganglions pathologiques avec une performance comparable à celle d'un médecin. Il s'agit d'une amélioration par rapport à l'état de l'art et la première étude que nous connaissons qui part directement des images TEP/TDM au FDG du corps entier pour aboutir à la détection des ganglions lymphatiques médiastinaux envahis. Les travaux futurs doivent étendre cette étude à un plus grand nombre de patients et étudier plus en profondeur la manière dont le modèle se comporte lors les images sont acquises avec une machine différente.

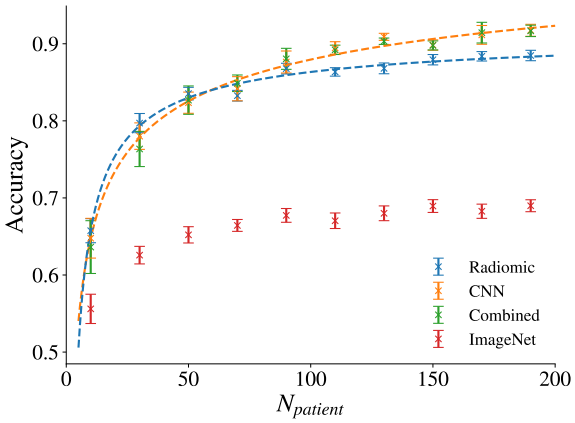
Chapitre 6 - Distinction entre l'endocardite sur prosthèse valvulaire et l'inflammation non spécifique Au chapitre 6, nous utilisons des méthodes similaires pour distinguer l'endocardite sur prothèse valvulaire (EVP) de l'inflammation non spécifique (INS) sur les images cardiaques TEP/TDM au FDG. Le EVP est une complication grave de la chirurgie d'implantation d'une valve cardiaque, avec un taux de mortalité de 22,8% [2]. Nous disposions d'une cohorte de 97 patients labellisés par un radiologue, incluant 26 cas de EVP et 71 cas de NSI. Notre modèle le plus performant était une architecture 3D ResNet, qui a atteint une AUC de 0,79 [0,61 à 0,94]. Nous avons comparé ces résultats à ceux d'une collègue, qui a créé des modèles univariés et bivariés en utilisant des caractéristiques radiomiques. Elle a obtenu des résultats comparables aux nôtres, avec un modèle bivarié utilisant SUV_{max} et GLCM Dissimilarity donnant une AUC de $0,80 \pm 0,06$. Cependant, aucun de ces modèles n'était significativement meilleur qu'un modèle univarié utilisant uniquement le SUV_{max}, qui a conduit à une AUC de $0,68 \pm 0,05$. Ces résultats sont décevants mais suggèrent que davantage de données sont nécessaires pour exploiter la puissance de ces modèles plus complexes.

Chapitre 7 - Comparaison des caractéristiques radiomiques et profondes Dans la seconde moitié de la thèse, nous nous concentrons sur les problèmes méthodologiques associés au machine learning en imagerie médicale. Au chapitre 7, nous testons expérimentalement la performance, l'interprétabilité et la stabilité de modèles radiomiques et CNN sur trois ensembles de données. Le premier est un ensemble d'images cérébrales 2D, composé de 3064 images IRM pondérée T1 avec produit de contraste, contenant des tumeurs qui ont été classées comme des méningiomes, des gliomes ou des tumeurs de l'hypophyse. Le deuxième est un ensemble de données TDM pulmonaires 3D accessible au public (l'ensemble de

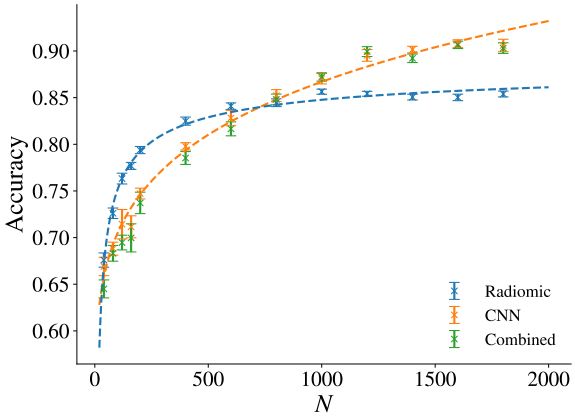
données LUNA (Lung Nodule Analysis)) comprenant plus de 700 000 régions suspectées de contenir un nodule pulmonaire, dont 1 166 sont pathologiques. Enfin, l'ensemble de données médiastinales du chapitre 5 a été utilisé. Dans nos premières expériences, nous comparons les performances obtenues avec des caractéristiques radiomiques et des caractéristiques CNN avec des tailles des données différentes. Les courbes résultantes sont présentées dans la Figure 1. Nous avons trouvé que les deux types de caractéristiques présentent des tendances similaires, en suivant des courbes de loi de puissance. Cependant, les modèles basés sur les CNNs s'améliorent plus rapidement lorsque la taille des ensembles de données augmente, ce qui démontre leur potentiel accru avec plus de données. Nous avons également combiné les deux types de caractéristiques, mais cela n'a amélioré les performances d'aucun des ensembles de données, les résultats correspondant principalement aux résultats basés sur les CNN.

Ensuite, nous testons l'interprétabilité des modèles radiomiques et CNN. Pour les modèles CNN, nous utilisons des cartes de sensibilité aux occlusions et des modèles Grad-CAM. Pour chaque ensemble de données, nous essayons d'interpréter les modèles globalement (c'est-à-dire d'expliquer quelles caractéristiques générales les modèles utilisent) et localement (c'est-à-dire d'expliquer les prédictions pour les cas individuels). Quelques exemples sont présentés dans la Figure 2. Nous constatons que dans l'ensemble, les cartes ne sont pas particulièrement utiles, donnant souvent des résultats difficiles à expliquer malgré leur aspect esthétique. En revanche, les modèles radiomiques sont relativement simples (nous utilisons des SVM linéaires), de sorte qu'il est facile de voir quelles caractéristiques contribuent à une classification (voir la figure 3 pour un exemple). Cependant, les caractéristiques elles-mêmes sont souvent très compliquées. Par exemple, beaucoup des caractéristiques radiomiques les plus importantes sont basées sur les ondelettes, qu'il est difficile de relier à une signification biologique quelconque. Pour interpréter ces caractéristiques, nous essayons de les mettre en corrélation avec des alternatives plus compréhensibles. Nous reconstruisons ensuite les modèles radiomiques avec ces alternatives, pour voir si la substitution peut être faite tout en maintenant la performance. Les résultats sont mitigés: pour l'ensemble des données cérébrales, notre modèle interprétable atteint une exactitude de 0,70 contre 0,85 avec le modèle original. Pour l'ensemble de données LUNA, le modèle original atteint une exactitude de 0,85 ; le modèle plus interprétable donne une exactitude de 0,70. Pour l'ensemble de données médiastinales, le modèle original atteint une exactitude de 0,80 par rapport à une exactitude de 0,70 en utilisant l'alternative interprétable.

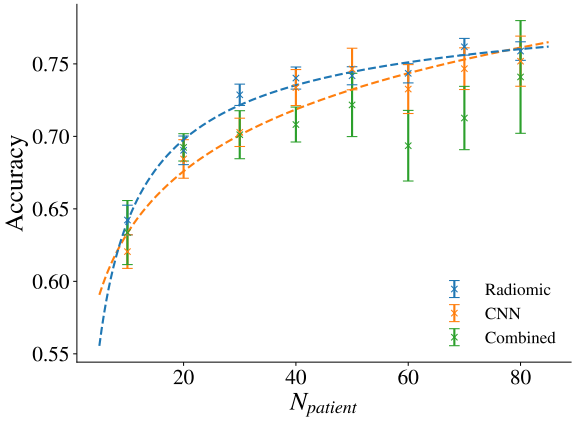
Ensuite, nous proposons une nouvelle méthode pour interpréter les modèles CNN en utilisant des caractéristiques radiomiques. Nous essayons de recréer chacune des cinq principales caractéristiques profondes des modèles CNN en utilisant des caractéristiques



(a) Brain Dataset

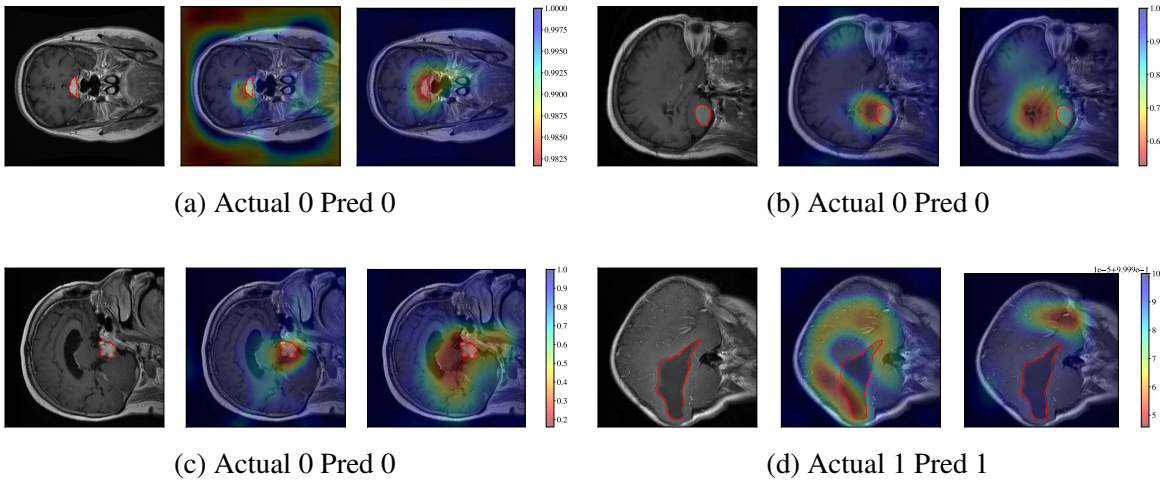


(b) LUNA Dataset



(c) Mediastinal Dataset

Fig. 1 Courbes d'apprentissage pour les ensembles de données cérébrales, LUNA et médiastinales, montrant les performances des caractéristiques radiomiques, CNN, combinées et (pour l'ensemble de données cérébrales) ImageNet. Les courbes de loi de puissance ont été adaptées aux données radiomiques et CNN



(a) Actual 0 Pred 0

(b) Actual 0 Pred 0

(c) Actual 0 Pred 0

(d) Actual 1 Pred 1

Fig. 2 Chaque ensemble d'images montre l'image IRM originale (à gauche), la carte grad-CAM (au centre) et la carte d'occlusion (à droite) pour un exemple tiré de l'ensemble de données cérébrales. Le contour rouge montre la délimitation de la tumeur. Le rouge sur les cartes Grad-CAM représente une forte intensité (ce qui signifie que le CNN se concentrait sur cette zone) alors que le bleu représente une faible intensité. Pour les cartes d'occlusion, les zones rouges indiquent une baisse de la prédiction lorsque cette zone a été cachée (c'est-à-dire que cette zone est importante pour la classification), et le bleu indique l'inverse. L'échelle sur chaque carte d'occlusion est différente, c'est pourquoi les échelles sont indiquées. La classe 0 correspond aux méningiomes, la classe 1 aux gliomes et la classe 2 aux tumeurs de l'hypophyse

radiomiques. La Figure 4 présente une comparaison entre les caractéristiques profondes originales et les caractéristiques radiomiques recréées pour l'ensemble des images cérébrales. Nous constatons que pour les images cérébrales, les caractéristiques profondes peuvent être recréées fidèlement, avec des corrélations de Pearson (r) entre 0,84 et 0,90. Pour l'ensemble de données LUNA, les résultats sont moins satisfaisants (r pour les cinq principales caractéristiques entre 0,71 et 0,85), et pour l'ensemble de données médiastinales, la méthode est en échec (r entre 0,44 et 0,64). Ces résultats suggèrent que pour certains problèmes seulement, les caractéristiques profondes peuvent être interprétées avec précision en utilisant les caractéristiques radiomiques.

Nous testons ensuite la stabilité des CNN obtenus lorsque l'apprentissage est répété dans les mêmes conditions. Des études ont montré que les CNN obtenus en répétant l'apprentissage à l'identique peuvent donner des prédictions individuelles différentes, même si les performances globales sont stables. Nous voulions voir dans quelle mesure cet effet était important à l'échelle médicale. Nous avons donc entrainé des modèles de CNN plusieurs fois, en comparant les performances globales et les performances au niveau de l'image. Nous avons constaté qu'avec un grand ensemble de données, il y avait des désaccords dans 7,4% des cas pour les images cérébrales, 17,0% des cas pour le LUNA et 2,2% des cas pour les

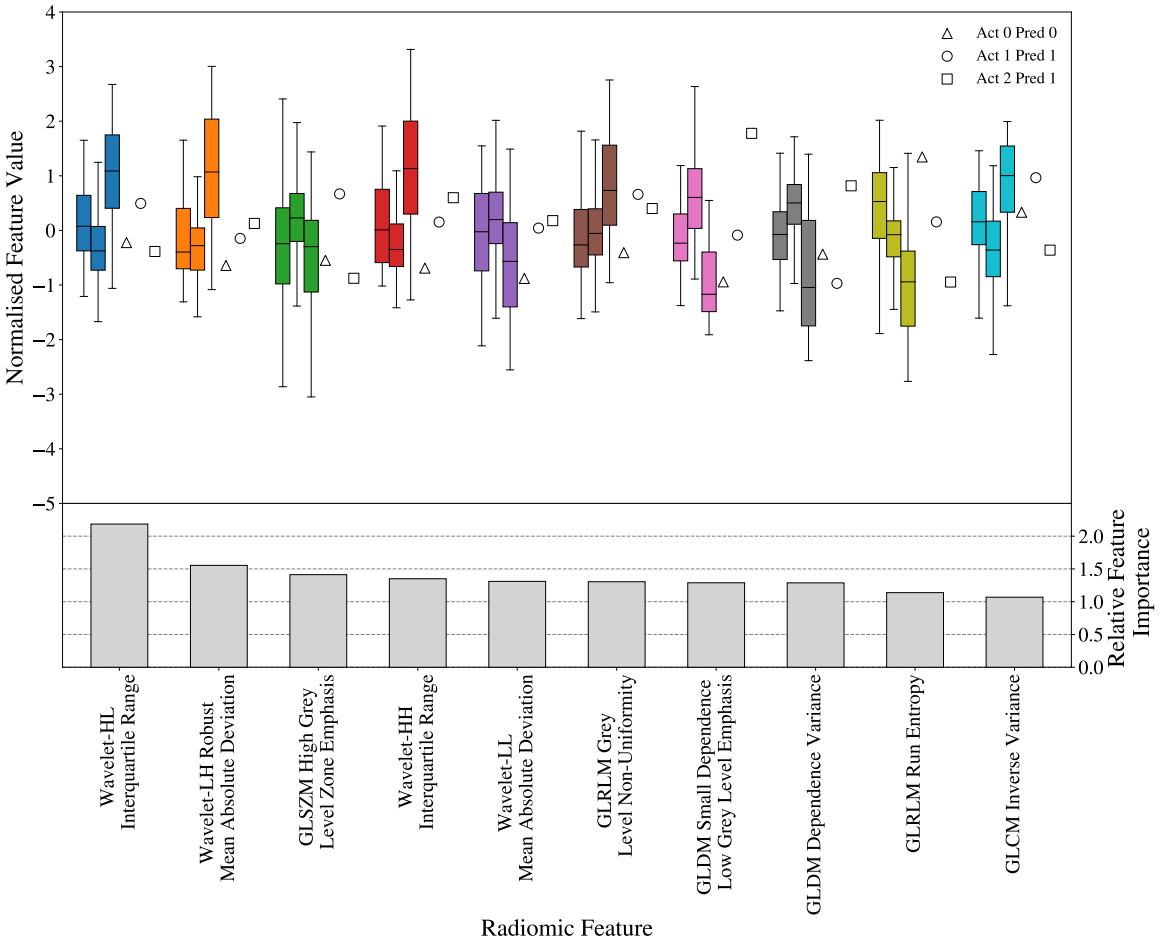


Fig. 3 Le boxplot montre comment les trois classes de l'ensemble de données cérébrales sont différenciées par chacune des dix caractéristiques radiomiques les plus importantes. □, ○, et △ sont trois exemples de cas. En comparant leurs valeurs aux boxplots, nous pouvons déduire quelles caractéristiques sont importantes pour leur classification. Les barres grisées représentent l'importance des caractéristiques (définie comme le coefficient moyen du SVM) de chacune des dix caractéristiques. La classe 0 correspond aux méningiomes, la classe 1 aux gliomes et la classe 2 aux tumeurs de l'hypophyse

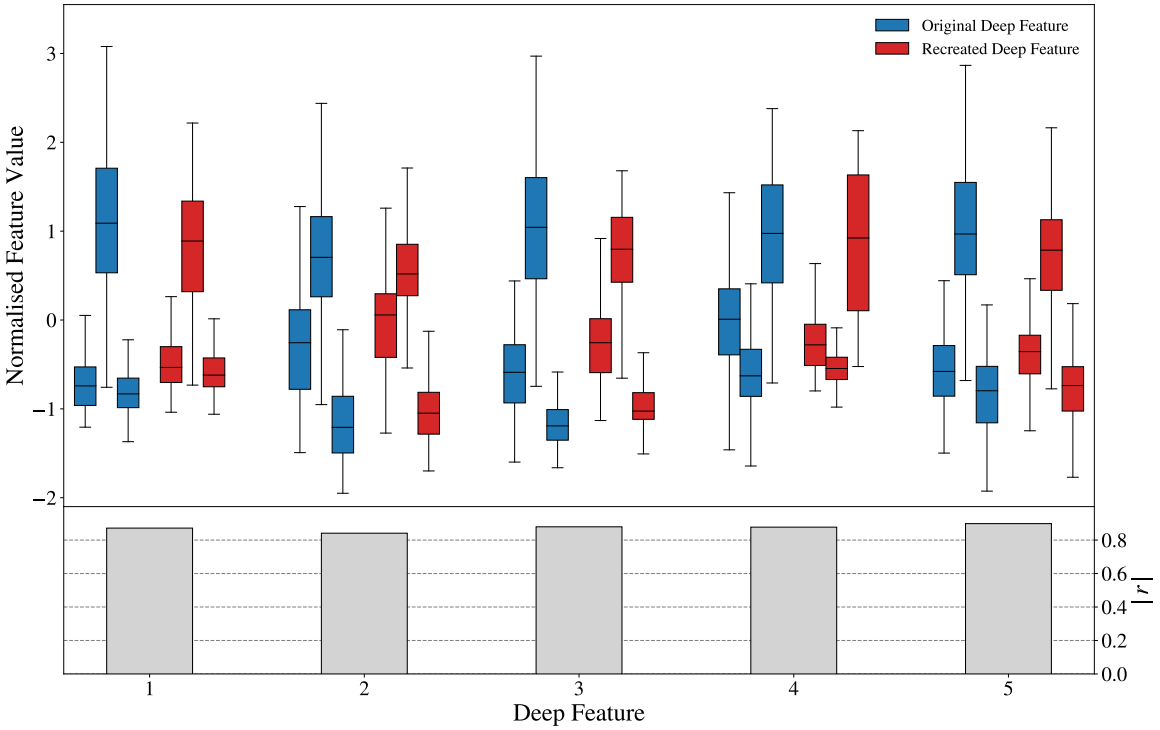


Fig. 4 Le boxplot compare les distributions des cinq principales caractéristiques profondes et de leurs homologues créées à l'aide d'un random forest regressor, pour l'ensemble des données cérébrales. Les barres grisées montrent l'importance relative (définie comme le coefficient du SVM) de chaque caractéristique profonde

ganglions médiastinaux. Pour les petits ensembles de données, les désaccords étaient plus importants, soit 10,8% pour les données cérébrales, 22,8% pour les données LUNA et 43% pour les données médiastinales. Ces résultats montrent clairement que les CNN individuels ne sont pas assez stables pour être utilisés dans un cadre clinique, en particulier lorsque la taille de l'ensemble de données utilisé pour l'apprentissage est petit.

Enfin dans ce chapitre, nous examinons comment la segmentation affecte les performances des modèles de classification. Pour les données cérébrales, nous comparons les performances des modèles entraînés avec des segmentations précises à celle des modèles entraînés avec des boîtes délimitant grossièrement les tumeurs. Nous constatons que pour les modèles radiomiques et les modèles CNN, une segmentation précise n'est pas nécessaire, et que les modèles entraînés sans segmentation précise sont plus performants.

Chapitre 8 - Adaptation d'un modèle aux données provenant d'une autre machine Dans le dernier chapitre, nous testons différentes méthodes pour adapter les modèles CNN aux données provenant d'une autre machine. C'est un problème courant avec les modèles de deep learning qui, bien qu'ils soient très performants pour classer des données contrôlées, échouent souvent lorsqu'on leur présente des données provenant d'une source différente. Sans la capacité de généralisation, ces modèles auront du mal à être utilisés dans la pratique clinique. Nous avons acquis un deuxième ensemble de données médiastinales à partir d'une machine plus récente. Comme au chapitre 6, nous avons fait labelliser les images par un médecin, ce qui a donné 75 patients incluant 65 ganglions. Nous avons utilisé ces images et les images de la première machine pour réaliser plusieurs expériences :

- **Apprentissage sur les données de la machine 1** Entraînement du CNN sur seulement les données de la machine 1 puis test sur les données de la machine 2.
- **Apprentissage sur un mélange des données provenant des 2 machines** Apprentissage du CNN sur un mélange 50/50 de données provenant des machines 1 et 2, puis test sur les données de la machine 2.
- **Transfer Learning** Entraînement du CNN sur les données de la machine 1 puis raffinement du modèle sur les données de la machine 2. Utiliser une moitié des données pour l'entraînement initial, puis une moitié pour affiner le modèle fin (la quantité totale de données utilisées pour l'entraînement est donc la même que celle des deux premières expériences).
- **Transfer Learning (davantage de données)** Utilisation du transfer learning comme pour l'expérience précédente, mais utilisation de l'ensemble des données pour l'entraînement initiale.

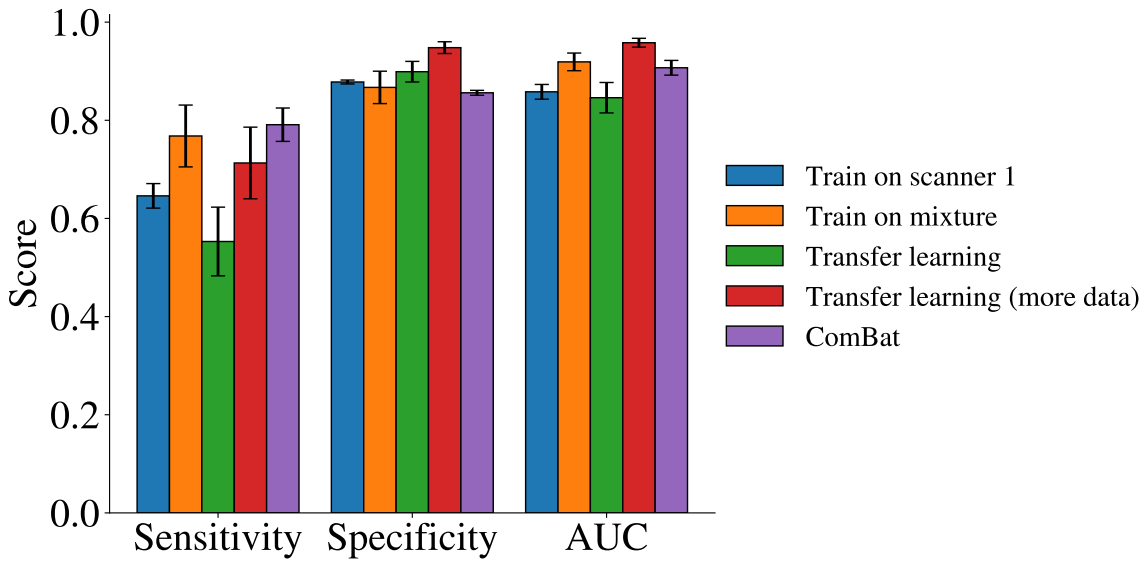


Fig. 5 Sensibilité, spécificité, et AUC scores pour les différentes méthodes d'adaptation

- **ComBat** Entrainement du CNN sur les données de la machine 1 puis extraction des caractéristiques de la couche finale et harmonisation des caractéristiques pour les recaler sur les valeurs mesurées sur la machine 2. Utilisation de ces caractéristiques harmonisées pour construire un SVM afin de classer les données de la machine 2.

Les résultats sont présentés dans la figure 5. L'application du modèle directement aux données du scanner 2 a donné de mauvaises performances, avec une sensibilité de $0,65 \pm 0,03$ et une spécificité de $0,878 \pm 0,004$. L'apprentissage sur un mélange des données a amélioré les performances significativement, avec une sensibilité de $0,77 \pm 0,06$ et une spécificité de $0,87 \pm 0,03$. L'utilisation de transfer learning avec une moitié des données à chaque étape a donné de mauvais résultats (sensibilité $0,55 \pm 0,07$ et spécificité $0,90 \pm 0,02$), mais l'utilisation de davantage de données a amélioré les performances (sensibilité $0,71 \pm 0,07$ et spécificité $0,95 \pm 0,01$). Enfin, l'utilisation de ComBat pour harmoniser les caractéristiques a permis d'obtenir une sensibilité de $0,79 \pm 0,03$ et une spécificité de $0,856 \pm 0,005$.

Ces résultats montrent que notre modèle créé au chapitre 5 peut être adapté aux données d'un deuxième scanner. Il s'agit d'une preuve importante de la robustesse et de la généralisation du modèle. Nous avons testé plusieurs méthodes d'adaptation et proposé une nouvelle méthode, ComBat. Chacune des méthodes testées présente des avantages et des inconvénients, mais ComBat pourrait être une option viable.

Bien que ces résultats soient un début prometteur, il est clair que beaucoup plus de travail doit être fait pour expliquer et résoudre le problème de l'adaptation d'un modèle aux données provenant d'une autre machine.

Acknowledgements

First and foremost I would like to thank my supervisor, Irène Buvat. Your continuous support, insights, and knowledge of the field were indispensable, and your infectious enthusiasm and energy for your work is an inspiration.

Thanks to my friends and colleagues at the *Institut Curie*, *CEA*, *HYBRID*, and elsewhere for their advice, questions, and moral support on the long road to completion. Thanks also for making me feel welcome in a strange new land and answering my endless questions on French language and bureaucracy. In particular, thanks to Christophe Nioche for help with (and for creating!) the LIFEx software and for being a general computing guru. A big thank you also to Michaël Soussan for his time and expertise, and for putting up with my constant demands for more data.

I would also like to thank the uncountable unnamed heroes of *Stack Exchange*, *GitHub*, and across the internet, for helping write, debug, and fix my code. Their combined contribution to global scientific research must be colossal.

Thanks to my parents for all their support and for instilling in me a curiosity and passion for science. I especially thank them both for their patience in meticulously proofreading all 250+ pages of this thesis, which could not exactly be described as 'easy reading'.

Finally, my love and thanks to Sinead for keeping me sane and ensuring that for at least a couple of hours each day I was not thinking about coding and cancer.

Table of contents

List of figures

List of tables

Chapter 1

Introduction

1.1 Motivation

Technology and human innovation never cease to surprise and amaze. Medicine has always been one of the most dynamic areas of scientific innovation and one where the benefits of this innovation can most clearly be seen. Going to the doctor for breast cancer treatment one hundred years ago would probably have resulted in extremely disfiguring surgery and a five-year survival rate of only 40% [3]. Five-year survival rates are now over 90% and a much better understanding of the disease means treatment can be tailored to individual cases [4].

This pace of change and development continues into the 21st century. The current period has been called the *fourth industrial revolution*, as more advanced technology and the use of large quantities of data transform manufacturing and industrial processes. In medicine, artificial intelligence (AI) and data-driven techniques are promising to fundamentally change the way clinical medicine is conducted. This is a nascent field of research; volatile and fluid, a melting pot of different professions, with many questions unanswered. My motivation in undertaking this PhD was to bring my scientific expertise and enthusiasm to this domain, and be a part of and contribute to this new revolution.

1.2 Contribution of the PhD

The first contribution of this PhD is the use of AI, specifically machine learning (ML), to automate and solve medical imaging tasks, achieving state-of-the-art results in the automated detection of pathological lymph nodes in the mediastinum. Secondly, we use several different medical problems to examine broader problems in the field. We ask how the scanner type,

dataset size, and analysis technique affect the performance of ML models, and ask whether these models can be adequately trusted and interpreted in the high-risk domain of medical imaging.

1.3 Summary of Chapters

The rest of the thesis is organised as follows:

Chapter 2 introduces the different imaging modalities used in the thesis and explains the theory of radiomics, the analysis of medical images using mathematical features. We outline the methodology and discuss the state-of-the-art and current challenges.

Chapter 3 gives an overview of the theoretical basis for machine learning models and methods.

Chapter 4 introduces convolutional neural networks and their application to medical imaging datasets.

Chapter 5 presents our work building a model to automatically detect pathological mediastinal lymph nodes using deep learning.

Chapter 6 employs deep learning techniques to automatically distinguish between prosthetic valve endocarditis and non-specific inflammation.

Chapter 7 demonstrates several experiments to compare the performance, interpretability, and stability of radiomic and deep learning-based models.

Chapter 8 proposes a new method for harmonising multi-scanner deep learning models and compares it to existing methods.

Appendix A details the software and hardware used during the thesis.

Appendix B describes the radiomics features used.

Appendix C lists all the CNN architectures used in the thesis.

Appendix D gives details of a multi-scale method used in Chapter 5.

Chapter 2

Medical Imaging and Radiomics

In 1895 Wilhelm Röntgen accidentally discovered X-rays. Experimenting with his new discovery, Wilhelm took a picture of his wife's hand - a radiograph - using X-rays, and thus the field of medical imaging was born. Versatile and with the advantage of being non-invasive, medical images have become ubiquitous in clinical analysis. With the development of more imaging methods (*modalities*), such as positron emission tomography (PET) and magnetic resonance imaging (MRI), different internal processes and structures can be visualised. As of 2010 five billion medical imaging studies had been conducted worldwide [5]. In this section we briefly describe the imaging modalities used in this thesis.

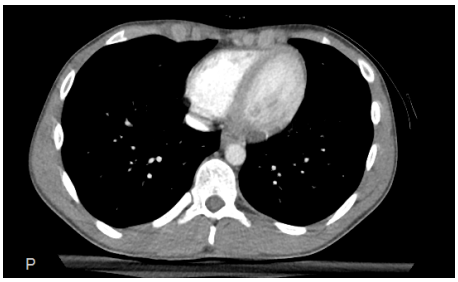
2.1 Computed Tomography

Using the same principle as Röntgen's radiograph, computed tomography (CT) uses differences in X-ray absorption to reconstruct detailed 3D images of the inside of the body. Multiple X-rays are taken from different angles by a rotating machine while simultaneously the patient moves through the machine. The signals produced are processed to reconstruct a 3D CT image, with voxel values representing the level of attenuation of the ionising X-rays due to the different tissues in the body. Materials with lower atomic numbers (such as air) absorb little radiation so appear dark, whereas those with higher atomic numbers (such as bone or metal) absorb more radiation so appear bright. These values are converted to Hounsfield units (HU), which give an attenuation coefficient relative to water

$$HU = 1000 \cdot \frac{\mu - \mu_{water}}{\mu_{water} - \mu_{air}} \tag{2.1}$$

where μ, μ_{water}, and μ_{air} are the linear attenuation coefficients of the voxel, water, and air respectively (HU are therefore dimensionless). Typical HU values are -1000 for air, 0 for

(a) Sagittal CT of the head without contrast. Taken from [9]



(b) Axial CT of the lungs with contrast. Taken from [10]

(c) Coronal CT of normal cervical vertebrae without contrast. Taken from [11]

Fig. 2.1 Three example CT scans. The air in the lungs and around the scans has a low HU value so appears black. The bones have high HU values so appear white

water, and from +300 to +2000 for bone [6]. Radiocontrast agents are sometimes injected to highlight specific structures that would otherwise be hard to delineate, with images taken *precontrast* and *postcontrast*. The most common radiocontrast agents are iodine-based, which allow visualisation of vascular structures [7]. CT images have spatial resolutions of the order of 1 mm in-plane (inter-plane resolutions are typically lower) and are able to discriminate between materials with attenuation differences as low as 0.1% [8]. They are used in a huge range of medical applications, from brain to lung to intestinal scans [6]. See Figure 2.1 for some example images.

2.2 Magnetic Resonance Imaging

Developed in the 1970s, MRI imaging uses an external magnetic field to excite certain nuclei (usually hydrogen due to its abundance in the body), allowing their locations to be mapped. In the presence of a large magnetic field $\mathbf{B}_0$ (typically of the order of a few Tesla) the hydrogen atoms (i.e. protons) will precess (not in phase) around the direction of the field with the *Larmor frequency* $\omega_0 = \gamma|\mathbf{B}_0|$, where γ is the nucleus-specific gyromagnetic ratio. This gives a net magnetisation vector $\mathbf{M}_0$ in the $\mathbf{B}_0$ direction (by convention the z-direction). Application of a radio frequency (RF) pulse can realign the magnetisation $\mathbf{M}$ to be in the xy-plane or reverse it to be in the $-z$-direction. This also makes the protons become in-phase. The *relaxation* to equilibrium after the pulse is described by the Bloch equations and governed by the relaxation times T_1 and T_2 [12].

T_1, or the spin-lattice relaxation time, is the decay constant for the recovery of the z-component of the magnetisation. If $\mathbf{M}$ has been aligned into the xy-plane such that $M_z(t=0)=0$, then

$$M_z(t) = |\mathbf{M}_0|(1-e^{-t/T_1}) \tag{2.2}$$

T_2, the spin-spin relaxation time, is the decay constant for the transverse magnetisation $\mathbf{M}_{xy}$ and is caused by the loss of phase coherence of the spins over time

$$M_{xy}(t) = M_{xy}(t=0)e^{-t/T_2} \tag{2.3}$$

Differences in hydrogen content and chemical structure lead to different T_1 and T_2 values. During relaxation electromagnetic energy is released, and thus by measuring this response different tissues can be differentiated and an MRI image can be reconstructed. To localise the response an additional magnetic field gradient is applied such that the total field strength varies in space. The Larmor frequency then varies with position, allowing spatial localisation of signals.

A wide array of different MRI *sequences* can be created by varying pulse sequences and field gradients, allowing emphasis of different tissue types. Typically several MRI sequences will be taken in a single MRI scan. *T_1-weighted* sequences emphasise differences in T_1 relaxation times. Resulting images have lower intensities for tissues with high water content, such as tumours, and higher intensities for fat. *T_2-weighted* images measure T_2 relaxation times, resulting in higher intensities for high water content tissues and lower intensities for fat. In the brain, Proton Density-*weighted* (PD-*weighted*) images have a higher contrast between grey matter (high intensity) and white matter (low intensity). There are a multitude of other sequences designed to highlight different anatomical features or functional processes, making MRI imaging very versatile. Some examples are shown in Figure 2.2. Additionally, contrast agents (most commonly gadolinium-based) can be injected or ingested to change relaxation times, giving further sequencing options [13].

MRI is commonly used in brain, cardiovascular, and musculoskeletal imaging [14]. Anatomical MRI images are of high spatial resolution, usually around 1 mm in-plane [15] [16]. However, unlike CT and PET imaging, there is no absolute unit scale. This means that the voxel values of one MRI scan are not related to those of a second scan, even when using the same MRI scanner. Careful consideration therefore needs to be given to the standardisation of MRI images[1] [17] [18] [19].

[1] This is the case for the anatomical MRI images used in this thesis. Functional MRI images have lower resolutions, typically 4-5 mm, and can be expressed in absolute units

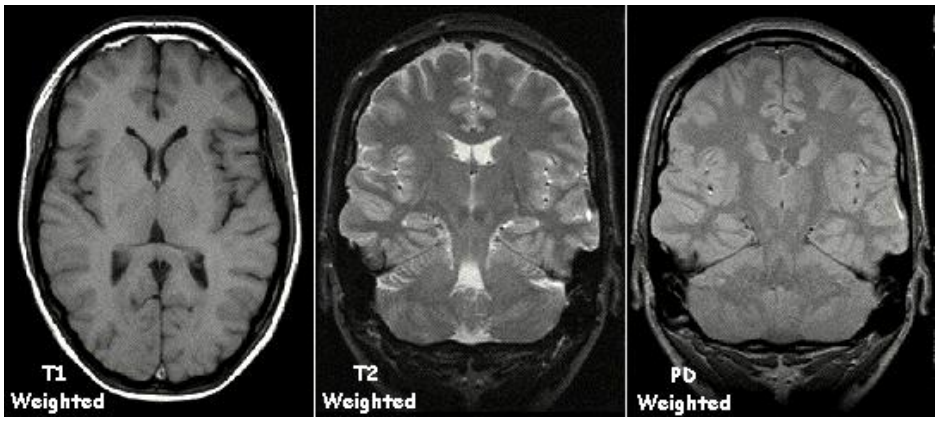


Fig. 2.2 T_1-weighted, T_2-weighted, and PD-weighted MRI images of the brain. Different contrasts between the sequences allow different anatomical features to be identified. Taken from [20]

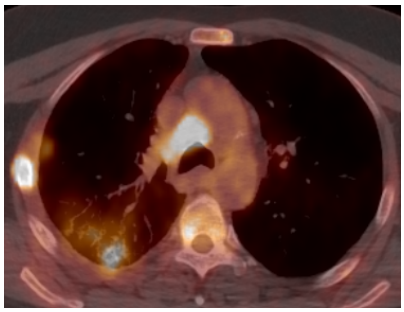
2.3 Positron Emission Tomography

PET is a functional imaging technique that uses radioactive substances to measure changes in body processes. A compound containing radioactive material (a *radiotracer*) is injected into the body. As the radiotracer decays it emits positrons, which annihilate with electrons to produce pairs of photons. These photons are detected by a scanner outside the patient, and these signals are used to reconstruct an image of radioactive activity in the body. The image is then converted into the dimensionless standardised uptake value (SUV) to account for injection and body weight differences between patients[2]

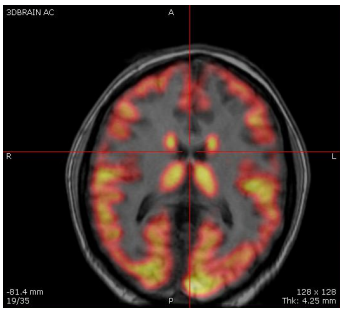
$$SUV = \frac{\text{radioactivity concentration}}{\text{injected dose}} \cdot \text{body weight} \tag{2.4}$$

Different radiotracers are used to measure different processes. The most common radiotracer is fluorodeoxyglucose labelled with fluorine-18 (^{18}F-FDG), which is a marker of glucose consumption and therefore metabolic activity. FDG-PET is widely used in oncology to detect tumours and monitor tumour changes [21] [22] [23]. PET images have lower resolutions than CT or MRI images (usually around 4 mm). Because functional information is more useful in conjunction with anatomical information, modern PET scanners are often combined with CT or MRI scanners (PET/CT or PET/MR). The two scans can then be performed almost simultaneously, reducing registration error (in PET/CT the acquisitions are sequential, whereas in PET/MR they can be simultaneous or sequential). Some examples of FDG-PET

[2]To make SUV dimensionless the body volume is required, not the body weight. Usually the body weight is measured and implicitly converted into body volume assuming that the average mass density of the human body is close to 1 g/mL



(a) Axial PET/CT of the lungs



(b) Axial PET/MR of the brain. Taken from [24]

Fig. 2.3 PET/CT and PET/MR scan examples. High uptake on the PET (indicating metabolic activity) is indicated in orange

images are shown in Figure 2.3. Note that in this thesis all references to PET refer to FDG-PET.

2.4 Radiomics

The name *radiomics* is inspired by other data-driven medical fields such as genomics and proteomics, with the *radio* prefix signifiying a relation to medical imaging. In traditional analysis of medical images evaluation is mostly qualitative (except for a few quantitative measurements such as tumour size and volume), relying on the individual physician's experience. This introduces intra and interobserver variation and can lead to important features being missed [25] [26]. Radiomics attempts to solve these problems by using mathematical functions to extract features from medical images that aid in clinical decisions. Features typically represent morphological or textural properties of the images and can be simple features such as tumour volume or sphericity, or more complex higher-order features representing tumour heterogeneity or necroses. Because they are quantitative these features bring consistency and reproducibility to radiology-based decisions. They can also find information imperceptible to the human eye and thus further our understanding and improve our results.

2.4.1 Methodology

Extracting radiomic features from images usually consists of three steps:

1. **Region Selection** Generally radiomic features are calculated over a specific region of interest (ROI) delineated by segmentation (e.g. a tumour). Image segmentation can be

done manually (by a radiologist), semi-automatically (using a segmentation algorithm e.g. region-growing or thresholding), or fully automatically (using a deep learning algorithm e.g. U-Net).

2. **Preprocessing** To extract features which are consistent and reproducible across a dataset, images first need to be preprocessed. Studies have shown that the robustness of radiomic features is highly dependent on preprocessing settings [27] [28]. During preprocessing voxels are interpolated to give isotropic voxel spacings, outliers are removed, and values are discretised. For many machine learning models features are also rescaled to have zero mean and unit variance. Discretisation involves grouping values by binning. This reduces noise-dependency of features and is essential for some texture features [29]. Bins are set using either a fixed bin size or by fixing the total number of bins. Fixed bin size is preferred when voxel values represent absolute units (e.g. PET and CT), whereas in general a fixed bin number is recommended when units are arbitrary and differences between values are important (e.g. MRI) [30] [31] [32] [33]. However, there is no absolute consensus, and some studies have called this into question. Similarly, there is no consensus on the number of bins to use. A balance needs to be struck between bins being small enough to retain discriminatory power and large enough to contain sufficient samples. It is crucial that discretisation choices made in a study are stated in any corresponding publication, so experiments can be correctly replicated.

3. **Feature Extraction** Once images are processed the radiomic features can be extracted. In this thesis we extract features using the open-source PyRadiomics software, and mathematical details of all features can be found in the documentation [34]. We use first-order features computed directly from the histogram of image values (e.g. skewness, mean, and variance). These features are global, and do not depend on the locations of the voxels. We also extract local-scale textural features from the Grey Level Co-occurence Matrix (GLCM), Neighbouring Grey Tone Difference Matrix (NGTDM), and Grey Level Dependence Matrix (GLDM), and regional-scale textural features from the Grey Level Run Length Matrix (GLRLM) and Grey Level Size-Zone Matrix (GLSZM). Additionally, we extract first-order features after applying wavelet-based, Laplacian of Gaussian (LoG), exponential, and gradient filters. These filters are applied before feature extraction and alter the image, allowing extraction of different information that can be clinically useful [35] [36] [37]. Full details of all radiomic features used can be found in Appendix B. Once extracted radiomic features can be used in machine learning models, as described in the next chapter.

2.4.2 State-of-the-Art and Current Challenges

Radiomic models have been successfully deployed to classify tumours, predict survival times, and predict therapy response, across the breadth of oncology and beyond [25] [26] [38]. The field has huge potential and is evolving rapidly [39].

Being a field in its infancy, radiomics does face a few teething problems. Lack of standardisation, insufficient reporting, and lack of open-source datasets mean the reproducibility of radiomic studies is often poor. A review of 77 oncology-related radiomic studies found that "the overall scientific quality and reporting of radiomics studies is insufficient" [40]. Most studies are retrospective, so imaging protocols are not controlled. Multiple studies across different modalities have shown that many features are not robust to image acquisition and reconstruction settings, segmentation, image interpolation, discretisation, and image normalisation. These factors mean most radiomic studies remain single-centre, proof-of-concept studies, with no hope of clinical use. To tackle these problems, in 2016 the Image Biomarker Standardisation Initiative (IBSI) was initiated, which gives a set of common parameter definitions to adhere to for radiomic studies [31]. In 2017 the Radiomics Quality Score (RQS) was proposed, which assesses the quality of radiomic studies and gives a set of guidelines [41]. To harmonise data originating from different scanners and institutions, in 2018 the *ComBat* method was applied to radiomic features [42].

Another problem associated with many radiomic features is their lack of interpretability or link to other clinical factors [43]. This means that they remain abstract mathematical concepts, rather than furthering biological understanding. Although some work has been done to interpret these features, this is a rarity in radiomics publications [44].

Chapter 3

Machine Learning

Machine learning is the study of computer algorithms that improve automatically through experience

Tom Mitchell, *Machine Learning*, 1997

A breakthrough in machine learning would be worth ten Microsofts

Bill Gates, 2004

Despite these lofty statements, in its most basic form an ML model uses data $\mathbf{x}$ to predict an outcome y. Over the last couple of decades the influence of these models has grown, and they can be found in almost every aspect of our life, from drug design and email spam detection to weather predictions and stock market analysis.

The last 20 years have also seen an explosion in the use of ML algorithms in the medical imaging domain, as shown in Figure 3.1. Programs have been built to automate tasks across the gamut of clinical oncology, from tumour detection and segmentation to therapy decisions [45] [46] [47] [48]. In many cases these programs are beginning to outperform human experts. With more data, higher quality images, and more powerful computers, machine learning-based automation is predicted to fundamentally change the way clinical medicine operates [49] [48] [50].

Broadly speaking, machine learning algorithms can be split into three categories; supervised, unsupervised, and reinforcement learning. Supervised learning uses labelled data (data for which one has the results) to train a model, then uses this model to make predictions. In unsupervised learning data is unlabelled, and models are built to find structures or patterns within the data (e.g. to cluster the data). In reinforcement learning the model learns through

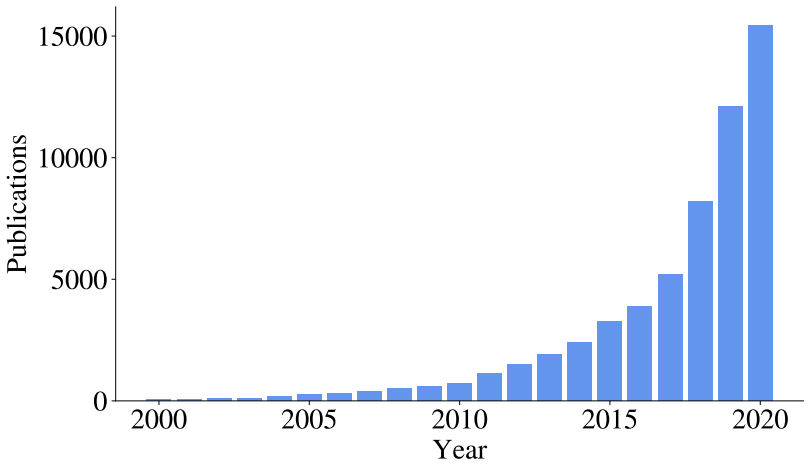


Fig. 3.1 Results on *PubMed* using the keyword search "Machine Learning" by year up to 2020, showing the rapid growth of interest in machine learning [51]

a feedback loop while performing a series of actions. An example would be the *AlphaGo* program that in 2015 became the first computer program to beat a professional *Go* player [52]. In this work we will mainly focus on supervised learning.

3.1 General Principles

Notation There is no agreed convention for mathematical notation in machine learning. We will denote the training set as $(\mathbf{x}^{(i)}, y^{(i)})$ where $1 \leq i \leq n$ represent the n training examples. $\mathbf{x}$ is the input data and y is the target variable (or class). Each training example $\mathbf{x}^{(i)}$ is made up of m features

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_m \end{bmatrix} \tag{3.1}$$

3.1.1 Training a Model

Given an input (e.g. a set of radiomic features), we want to build a model to make a prediction (e.g. whether the patient has cancer). Mathematically we want to find a function

$$f(\mathbf{x}) = y \tag{3.2}$$

If y is discrete the problem is a *classification problem* (e.g. whether the patient has cancer). If y is continuous it is a *regression problem* (e.g. predicting survival time). This function is found (or at least approximated) by using the data and labels in the training set to train a machine learning model. This is achieved through optimisation of the *weights* of the model, which we will denote θ_i. For example, for a linear model

$$f_\theta(\mathbf{x}) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 \tag{3.3}$$

we would optimise the weights θ_i. To do this we use a *loss function* (also called a *cost function*). The loss function measures how close $f(\mathbf{x})$ is to y. The weights are usually randomly initialised then iteratively optimised during training through a process such as *gradient descent*. The aim of gradient descent is to minimise the loss function, and this is achieved through a series of iterative gradient updates

$$\theta_i^{t+1} = \theta_i^t - \alpha \frac{\partial J(\theta_i^t)}{\partial \theta} \tag{3.4}$$

where J is the loss function and t is the iteration in the gradient descent. α is the learning rate and controls the magnitude of the iterative update. This is an important parameter to set during training. Too small and gradient descent will take a long time to converge; too large and it will never converge. α is usually kept constant, but many learning rate scheduling techniques have been proposed that change α as training progresses (see [53] for a detailed comparison). An example of gradient descent for a convex loss function is shown in Figure 3.2. In real-world examples with many more features the loss function topology can be much more complicated, with many local minima and divergences. Careful choice of training parameters is therefore key.

3.1.2 Evaluating a Model

Once the model is sufficiently trained, it needs to evaluated. A model could work well on the training data but fail to generalise and perform poorly on new data. This is called *overfitting*. A model could also not be powerful enough to accurately capture trends in a dataset, called *underfitting*. Figure 3.3 illustrates these scenarios. To ensure an accurate evaluation of a model's performance, data is usually split into a training, validation, and test set. The training data is used to train the model. The validation data is used to evaluate the performance when adjusting the model's parameters (e.g. learning rate, loss function). The test set is used for final evaluation, only when the model has been fully optimised. This is crucially important for ensuring that the results of any experiment can be trusted. These validation and test sets

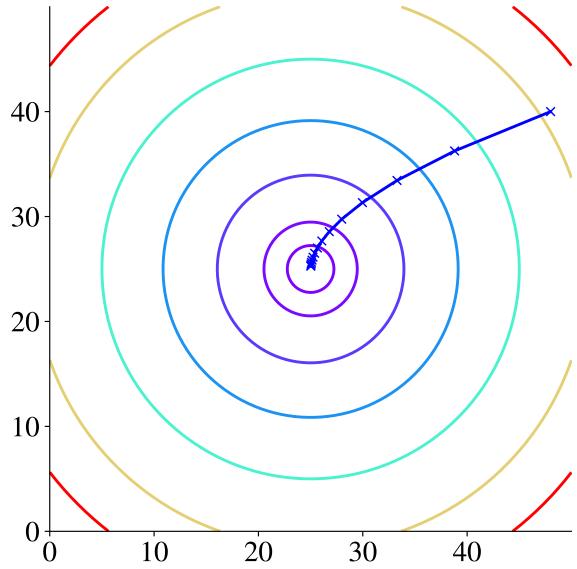
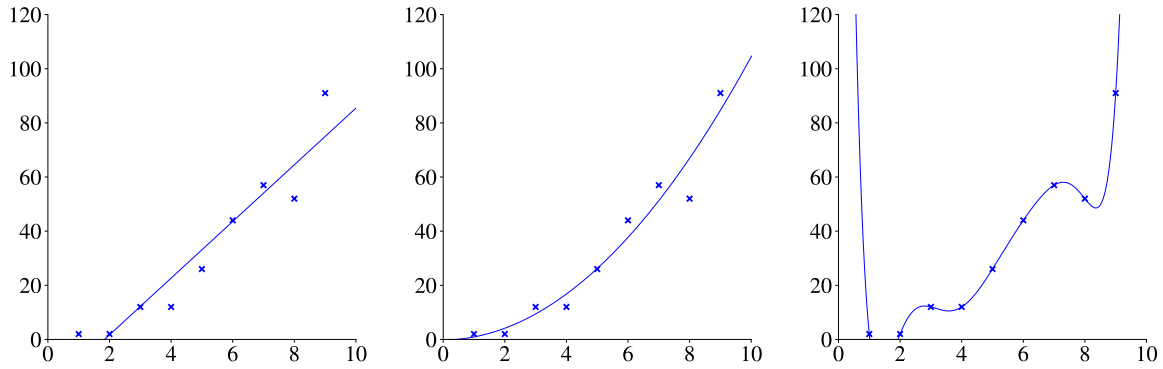


Fig. 3.2 An example of gradient descent on a quadratic convex loss function. The circles show the contours of the loss function, which is high around the edges (in red) and low in the centre (in blue). The trajectory of the gradient descent is shown in blue, with Xs marking the successive iterations. The learning rate was kept constant

need to be large enough that the performances of different experiments can be differentiated significantly.

In some instances, especially in the medical domain where datasets are small ($\sim$ 100 images), *cross-validation* may be used instead of a test set. In this case there is no test set. The dataset is split into k random groups (often called *k-fold cross-validation*), with one group reserved for validation. Once optimised using this first split, the model is trained k times, each time with a different validation set. The results can then be averaged to give an estimate of the model's performance. In this way there is less bias than using a single validation set result (which through optimising the training parameters may overfit the data). In the extreme case that k is equal to the size of the dataset (and thus the validation set contains only one datapoint), this is called leave-one-out-cross-validation (LOOCV). However, because there is no completely independent test set, these cross-validation schemes produce much weaker measures of performance. This is especially true when there are a lot of training parameters to optimise (such as in deep learning models). It has been shown that even when using LOOCV, the performance can be significantly overestimated compared to when using an independent test set [54].



(a) Linear fit (b) Quadratic fit (c) 8^{th} degree polynomial fit

Fig. 3.3 Examples of underfitting and overfitting using least squares regression. In (a) we have fit a straight line to the points, which slightly underfits the data. Adding another parameter, (b) is a quadratic curve, which fits the data well. (c) is an 8^{th} degree polynomial fit. Although it fits these datapoints well (the training data), it will probably not generalise well to new datapoints and is therefore overfitting the data

3.1.3 Other Optimisation Algorithms

Gradient descent is the simplest optimisation algorithm, using only first-order derivatives of the loss function. It is easy to understand and to implement. However, it has been largely superseded by more recent optimisation algorithms, the most relevant of which we briefly explain here:

Stochastic and Mini-Batch Gradient Descent In standard gradient descent the weights and biases are only updated after calculating the loss function over the whole dataset. If the dataset is very large this could take an impractically long time to reach convergence.

One solution to this problem is to split the dataset into batches and update the parameters after each batch. Because parameters are being updated more frequently convergence is faster, but by using a reasonable batch size we maintain stability in the loss across batches. The batch size is an important parameter to choose when optimising a neural network. In the case when the batch size is one this is called stochastic gradient descent (SGD), although this term is often also used for mini-batch gradient descent.

Adaptive Moment Estimation Rather than have a single learning rate, the Adaptive Moment Estimation (Adam) optimiser maintains separate learning rates for each parameter and adapts them as learning unfolds. Full details can be found in the publication [55]. Adam still uses mini-batches and has a parameter α which sets the overall learning rate

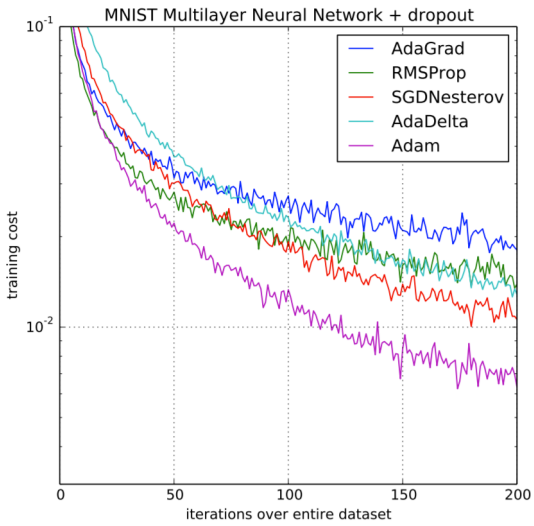


Fig. 3.4 Converge rates of several optimisation algorithms using a multilayer neural network on a handwritten classification challenge. Taken from [55]

of parameters. Adam is probably the most commonly used optimisation algorithm and is generally recommended as the go-to [56] (although as ever in machine learning there are exceptions, see [57]). Figure 3.4, taken from the original Adam publication, compares the converge rates of several optimisation algorithms, showing Adam is the quickest.

3.1.4 Loss Functions

The topology of the loss function determines the updates during gradient descent and thus the eventual parameters of our optimised model. It must therefore be a good representation of the performance metric we are trying to maximise (i.e. minimising the loss must lead to maximising the performance). Here we define loss functions relevant to this thesis:

Mean Square Error Loss Mean Square Error (MSE) is a simple quadratic loss, commonly used in regression problems

$$J(\theta) = \frac{\sum_{i=1}^{n} (\hat{y}^{(i)} - y^{(i)})^2}{n} \tag{3.5}$$

where $\hat{y}_\theta^{(i)} \equiv f_\theta(\mathbf{x}^{(i)})$ are the predictions. Because it is quadratic, more weight is given to outliers a long way from the predicted value. When the prediction $\hat{y}$ is near y the gradient is small, meaning gradient descent is slower and therefore more accurate.

Cross-Entropy Loss Cross-entropy is the default loss function for classification problems and requires input values between zero and one. In a binary classification problem the cross-entropy loss is defined as

$$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}[y^{(i)}\log(\hat{y}^{(i)}) + (1-y^{(i)})\log(1-\hat{y}^{(i)})] \tag{3.6}$$

which is equivalent to

$$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=0}^{1} y_j^{(i)}\log(\hat{y}_j^{(i)}) \tag{3.7}$$

where the summation j is over the classes 0 and 1. For multi-class problems with K classes this generalises as

$$J(\theta) = -\frac{1}{n}\sum_{i=1}^{n}\sum_{j=1}^{K} y_j^{(i)}\log(\hat{y}_j^{(i)}) \tag{3.8}$$

DICE Loss Relating to the DICE score (see 3.4), the DICE loss is often used in segmentation problems

$$J(\theta) = 1 - \sum_{i=1}^{n}\frac{2\hat{y}^{(i)}y^{(i)}}{\hat{y}^{(i)} + y^{(i)} + 1} \tag{3.9}$$

Hinge Loss The hinge loss is used for binary classification where the two classes are defined as $+1$ and -1. If $\hat{y}^{(i)}$ and $y^{(i)}$ have the same sign (meaning the prediction is correct) and $|\hat{y}^{(i)}| \geq 1$ the loss $J(\theta) = 0$. If not, there is a penalty that is linear with $\hat{y}^{(i)}$. Hinge loss is most often used when training support vector machines (SVMs), as expounded in 3.2.2.

$$J(\theta) = \sum_{i=1}^{n}\max(0, 1-\hat{y}^{(i)}y^{(i)}) \tag{3.10}$$

3.1.5 Class Balancing

Often in ML problems we are faced with class imbalance (e.g. far more negative diagnoses than positive). This poses a problem when using a loss function, as the model could easily reduce the loss by predicting only the majority class. This problem can be reduced by undersampling the majority class or by oversampling the minority class. We can also weight the classes in the loss function, applying a proportionally larger penalty for misclassification

of the minority class. Thirdly, synthetic data can be created using a technique such as the synthetic minority over-sampling technique (SMOTE) [58] or data augmentation (see 4.3.1).

3.2 Machine Learning Algorithms

A huge range of different machine learning algorithms have been proposed to utilise the above formulation. In this section we will focus on those used in this thesis, but it is worth noting that there are dozens of options, each with their own strengths and weaknesses and supporters and detractors [59] [60] [61].

3.2.1 Random Forest Classifier

Originally proposed in 2001 [62], *random forest classifiers* are one of the most popular machine learning classifier algorithms and have been used in many radiomic studies [63] [64] [65]. They are easy to understand and interpret, and computationally inexpensive to build and run.

3.2.1.1 Decision Trees

Decision trees are the building blocks of random forests. Figure 3.5 illustrates an example. The dataset (at the root node of the tree) is iteratively split into subsets using splitting rules based on the features at each internal (non-leaf) node within the tree. This process is repeated until subsets only contain a single class (at a leaf node), or another criterion is met (e.g. a limit on the maximum depth of the tree). Any new data (i.e. a test set) can be predicted by applying the same rules and using the modal classes at each leaf node. The best split at each internal node is found using a metric that measures the homogeneity of the classes in the resulting nodes (e.g. Gini Impurity, Information Gain) [66]. For example, in Figure 3.5 the first split is made by colour, as this increases the homogeneity of the resulting groups more than splitting by underlining.

3.2.1.2 Random Forests

One problem with decision trees, especially if they are very deep, is their tendency to overfit the data. Random forests reduce this problem. As the name suggests, random forests contain several decision trees that operate as an ensemble. Each tree makes a prediction, and the class with the most votes is taken as the forest's prediction.

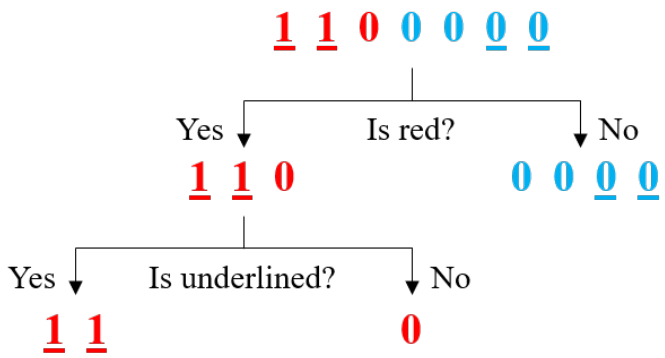


Fig. 3.5 An example decision tree. Two classes (0 and 1) are to be sorted by their features (colour and underlining). At the first internal node the data is split by colour, then at the second by underlining. Each leaf node then contains only a single class and the process is therefore finished

The key to the success of a random forest is to ensure that different trees are uncorrelated. In this way while some trees may be wrong, many other trees will be right, so as a group the trees are able to improve the performance. To enforce non-correlation *bagging* is used. For a dataset of size n each individual tree randomly samples n data *with replacement*, resulting in different trees. Additionally, at each internal node the tree can only pick from a random subset of the features (*feature bagging*). These force variation among the trees, leading to lower correlation and hence more robust models.

3.2.1.3 Random Forest Regression

Random forests can also be used to predict continuous variables, using a random forest regressor (RFR). These work in the same way as a normal random forest, but instead of test data being given the modal values at the leaf nodes, they are given the mean value.

3.2.1.4 Feature Importance

Random forests can be directly used to rank the importance of features, as described in [62]. Once a random forest is trained, for a feature x_i the values are randomly permuted among the training data. The *out-of-bag* error for each tree in the forest (the error associated with data not in a tree's training data due to bagging) is averaged and compared to the out-of-bag error without random permutation of the feature. Features which produce larger differences in these scores (i.e. permuting the feature results in a larger performance drop) are more important.

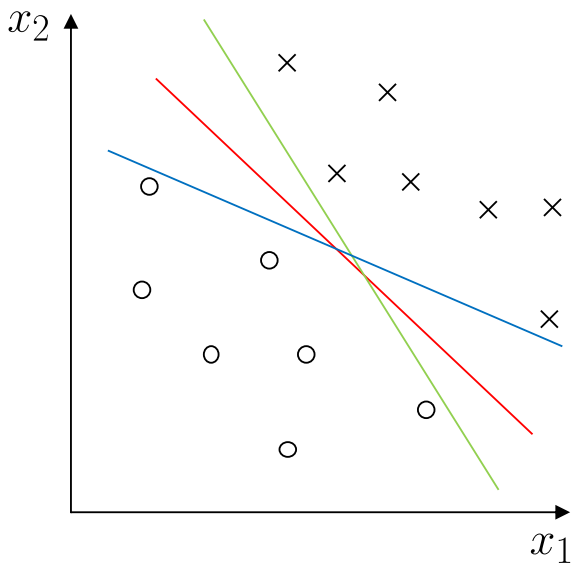


Fig. 3.6 Two classes ($\times$ and $\circ$) separated by three candidate hyperplanes using the features x_1 and x_2

3.2.2 Support Vector Machine

Originally postulated by Vapnik and Lerner in 1963 [67], the SVM is another popular supervised learning algorithm that has been used as a classifier in innumerable radiomic studies [68] [69] [70].

Given a set of training data with $y^{(i)}$ defined as either 1 or -1 (representing two classes), the aim of an SVM is to separate these classes using an $(m-1)$ dimensional hyperplane. As shown in Figure 3.6 there are many possible hyperplanes; the aim of the SVM is to find the plane with the maximum margin i.e. the largest distance between it and datapoints from both classes.

Any hyperplane can be written as

$$\mathbf{w} \cdot \mathbf{x} + b = 0 \tag{3.11}$$

where $\mathbf{w}$ is the normal to the hyperplane and b is the offset from the origin. As demonstrated in Figure 3.7, if the data are completely separable our aim is to select two parallel, maximally separated margins

$$\mathbf{w} \cdot \mathbf{x} + b = 1 \text{ (anything above this boundary has } y = 1) \tag{3.12}$$

$$\mathbf{w} \cdot \mathbf{x} + b = -1 \text{ (anything below this boundary has } y = -1) \tag{3.13}$$

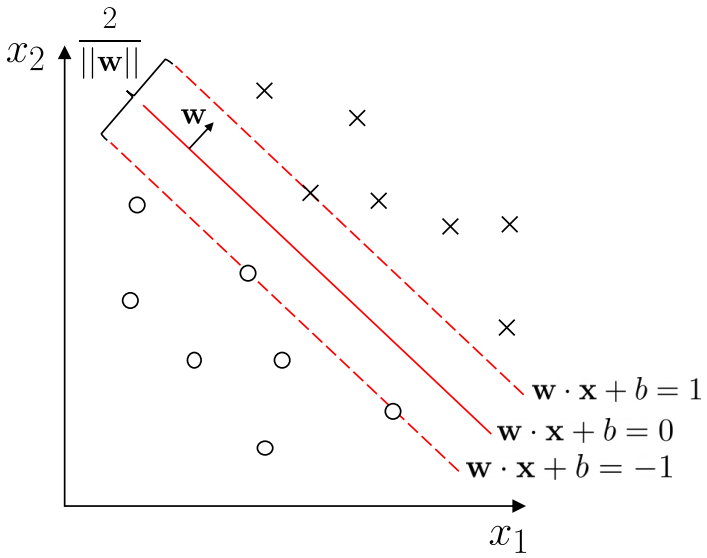


Fig. 3.7 The optimal hyperplane with its margins separating two classes ($\times$ and $\circ$) using the features x_1 and x_2. $\mathbf{w}$ is the normal to the plane and b is the offset from the origin. The distance between the margins is then $\frac{2}{||\mathbf{w}||}$

The distance between these margins is $\frac{2}{||\mathbf{w}||}$, so to maximise this distance we need to minimise $\frac{1}{2}||\mathbf{w}||^2$. We also need to ensure that we have no datapoints within the margins, so apply the constraints

$$\mathbf{w} \cdot \mathbf{x}^{(i)} + b \geq 1 \text{ if } y^{(i)} = 1 \tag{3.14}$$

$$\mathbf{w} \cdot \mathbf{x}^{(i)} + b \leq -1 \text{ if } y^{(i)} = -1 \tag{3.15}$$

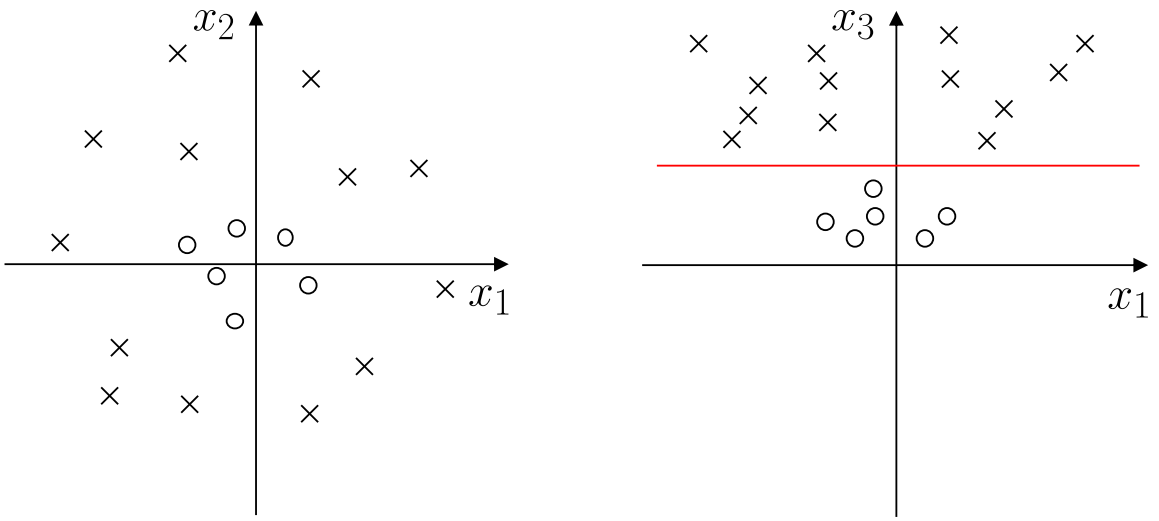
or

$$y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 \quad \forall \ 1 \leq i \leq n \tag{3.16}$$

However, most problems are not perfectly separable. In these cases we relax the condition in Equation 3.16 and allow samples $\mathbf{x}^{(i)}$ to be a distance $\zeta^{(i)}$ from their margin (*soft margins*). Any points to the wrong side of their margin are penalised by a value proportional to their distance from the margin using the hinge loss. The optimisation problem then becomes

$$\min_{\mathbf{w},b,\zeta^{(i)}} \frac{1}{2}||\mathbf{w}||^2 + C\sum_{i=1}^{n} \zeta^{(i)} \tag{3.17}$$

$$s.t. \ y^{(i)}(\mathbf{w} \cdot \mathbf{x}^{(i)} + b) \geq 1 - \zeta^{(i)} \quad \zeta^{(i)} \geq 0 \quad \forall \ 1 \leq i \leq n \tag{3.18}$$



(a) Two classes ($\times$ and $\circ$) are not linearly separable using the features x_1 and x_2

(b) After applying the kernel transformation $x_3 = x_1^2 + x_2^2$, the classes become linearly separable

Fig. 3.8 Two classes before (a) and after (b) a kernel transformation

The parameter C is set by the user and determines the size of the penalty for misclassified datapoints. It is thus an important parameter when optimising SVM models.

3.2.2.1 Kernel-Based Methods

For datasets that are not linearly separable, a solution is to use the *kernel trick*. The aim is to add another dimension that separates the data using an appropriate transformation k. For example, we see the classes in Figure 3.8a are not linearly separable. Adding another dimension $x_3 = x_1^2 + x_2^2$ (Figure 3.8b) allows us to separate the classes easily.

Commonly used kernels include the polynomial kernel

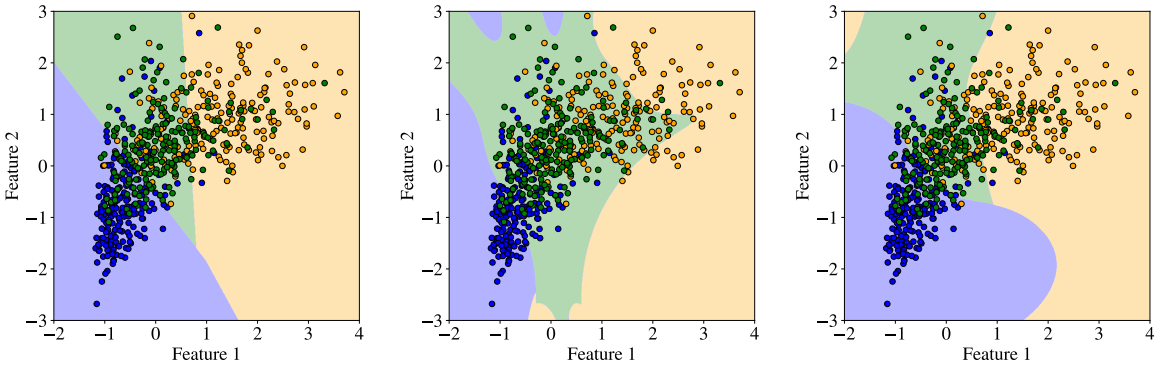
$$k(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i \cdot \mathbf{x}_j + 1)^d \tag{3.19}$$

where d is the degree of the polynomial and the Gaussian radial basis function (RBF) kernel

$$k(\mathbf{x}_i, \mathbf{x}_j) = \exp(-\gamma ||\mathbf{x}_i - \mathbf{x}_j||^2) \quad \gamma \geq 0 \tag{3.20}$$

where γ is a user-defined parameter determining the range of the kernel-based interaction. More examples can be found at [71].

More complicated kernels are more powerful, but because of this added complexity take more time to train, have more parameters to optimise, and are more liable to overfitting.



(a) Linear kernel (b) Polynomial (degree 7) kernel (c) RBF kernel

Fig. 3.9 Decision boundaries for SVMs trained with different kernels on a three-class brain MRI dataset. The colours indicate the three classes

Figure 3.9 shows the decision boundaries of SVMs trained using linear, polynomial, and RBF kernels on a three-class brain MRI dataset. We see the polynomial-based model is probably overfitting and would not generalise well on test data.

3.2.2.2 Multi-Class Data

For problems with more than two classes, the usual approach is to reduce the multi-class problem to multiple two-class problems. To do this either a *one-vs-all* or *one-vs-one* setup are used. For a K-class one-vs-all problem an SVM is built for each class, with that class fitted against all other classes (K SVMs are therefore needed). For each datapoint the class for which the classification confidence ($\mathbf{w} \cdot \mathbf{x}_i + b$) is greatest is chosen. In one-vs-one one SVM is built per pair of classes (so for K classes $K(K-1)/2$ SVMs are needed). For each datapoint the class with the most votes is then selected. In the event of a tie the class with the highest aggregate classification confidence is chosen.

3.2.3 Neural Networks

As the name suggests, neural networks were inspired by biological networks of neurones in the brain. Originally proposed by McCulloch and Pitts in 1943, a neural network is made up of nodes [72]. Each node takes m input features $x_1, ..., x_m$ multiplied by weight $w_1, ..., w_m$, plus a bias term b, and applies an *activation function* $h()$ to give an output y (see Figure 3.10)

$$y = h(\mathbf{w}\mathbf{x} + b) \tag{3.21}$$

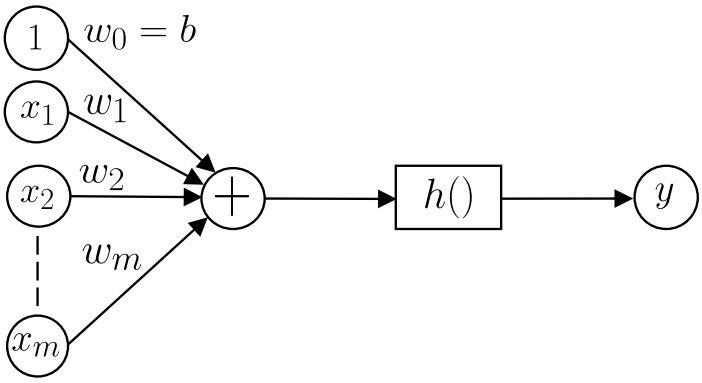


Fig. 3.10 A single neural network neurone. The input signals **x** are multiplied by the weights **w**, then the activation function $h()$ is applied to give the output y. The w_0 term is usually called the bias, b

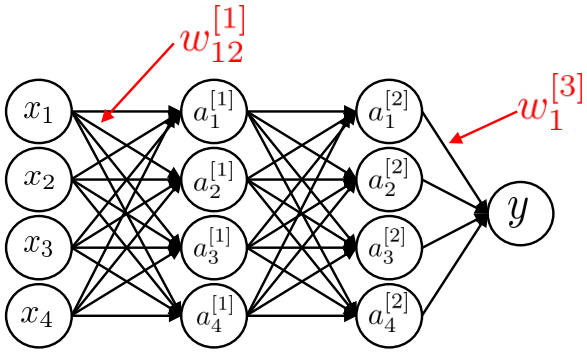


Fig. 3.11 A simple multi-layer fully connected neural network. The input signal **x** passes through hidden layers to give an output y. $a_i^{[j]}$ represents the output of the ith node in the jth hidden layer. Each connection between nodes has an associated weight. $w_{12}^{[1]}$, the weight connecting the first node in the first layer to the second node in the second layer, and $w_1^{[3]}$, the weight connecting the first node in the third layer to the final output, are shown

To create a neural network many neurones are combined in successive layers, such that the output of neurones in one layer passes into the input of neurones in the next layer. In this way more complex functions can be predicted. These layers that do not constitute input or output nodes are called *hidden layers*. Figure 3.11 shows a small example of a *fully connected* network (one in which each neurone in one layer is connected to all the neurones in the next layer). $a_i^{[j]}$ represents the ith node in the jth hidden layer, which is calculated by applying the activation function to the weighted sum of all connected nodes from the previous layer. For example $a_1^{[1]} = h(\sum_i w_{i1}^{[1]} x_i + b_1^{[1]})$, where $w_{i1}^{[1]}$ and $b_1^{[1]}$ are the weights and bias relevant to this node. Applying these functions across the whole network gives the output y.

The weights and biases of a network make up the set of weights which need to be optimised (equivalent to θ_i above). They are generally randomly initialised then iteratively optimised during training through gradient descent or another optimisation algorithm. This

is a two-step process; first, the inputs $\mathbf{x}^{(i)}$ are fed into the network in the *feedforward* step to give a set of predictions $\hat{y}^{(i)}$. These are then compared to the true values $y^{(i)}$ using the loss function. The gradients of the loss function with respect to the weights are used to update the weights (as in Equation 3.4), and these updates are then propagated back through the network using the *backpropagation* algorithm. Using the chain rule, the gradient of the loss function with respect to the weight $w_{ij}^{[k]}$ can be written as

$$\frac{\partial J}{\partial w_{ij}^{[k]}} = \frac{\partial J}{\partial z_j^{[k]}} \frac{\partial z_j^{[k]}}{\partial w_{ij}^{[k]}} \tag{3.22}$$

where $z_j^{[k]}$ is the input to node j in layer k before applying the activation function, i.e. $a_j^{[k]} = h(z_j^{[k]})$ and $z^{[k+1]} = w^{[k]}a^{[k]} + b^{[k]}$. The first term is usually called the *error* term and is denoted

$$\delta_j^{[k]} = \frac{\partial J}{\partial z_j^{[k]}} \tag{3.23}$$

The second term can be written as

$$\frac{\partial z_j^{[k]}}{\partial w_{ij}^{[k]}} = \frac{\partial}{\partial w_{ij}^{[k]}} \left(\sum_{l=0}^{r^{[k-1]}} w_{lj}^{[k]} o_l^{[k-1]} \right) = o_i^{[k-1]} \tag{3.24}$$

where the sum is over the r nodes in the previous layer and $o_i^{[k-1]}$ is the output of node i from this previous layer. Thus the partial derivative of the loss function J with respect to a weight $w_{ij}^{[k]}$ is

$$\frac{\partial J}{\partial w_{ij}^{[k]}} = \delta_j^{[k]} o_i^{[k-1]} \tag{3.25}$$

If the neurone is in the output layer then $a_j^{[k]} = y$ and the first term becomes

$$\delta_j^{[k]} = \frac{\partial J}{\partial y} \tag{3.26}$$

which can easily be calculated and is dependent on the loss function chosen. For hidden neurones a bit more work is needed. Using the chain rule again gives

$$\delta_j^{[k]} = \frac{\partial J}{\partial z_j^{[k]}} = \sum_{l=0}^{r^{[k+1]}} \frac{\partial J}{\partial z_l^{[k+1]}} \frac{\partial z_l^{[k+1]}}{\partial z_j^{[k]}} \tag{3.27}$$

Using the definition of $z_l^{[k+1]}$

$$z_l^{[k+1]} = \sum_{j=1}^{r^{[k]}} w_{jl}^{[k+1]} h(z_j^{[k]}) \tag{3.28}$$

where $h(x)$ is the activation function for the hidden layer, we get

$$\delta_j^{[k]} = \sum_{l=0}^{r^{[k+1]}} \frac{\partial J}{\partial z_l^{[k+1]}} w_{jl}^{[k+1]} h'(z_j^{[k]}) = h'(z_j^{[k]}) \sum_{l=0}^{r^{[k+1]}} \delta_l^{[k+1]} w_{jl}^{[k+1]} \tag{3.29}$$

Thus the error term in the kth layer $\delta_j^{[k]}$ is dependent on the errors $\delta_l^{[k+1]}$ in the $(k+1)$th layer. The errors propagate backwards from the output layer to the input layer, which is where the process gets its name. The weights are then updated using an optimisation algorithm, exactly as in 3.1.1.

3.2.3.1 Activation Functions

The activation function $h(x)$ introduces non-linearity, without which the network would collapse to a simple linear regression model. It must be monotonically increasing and differentiable. Figure 3.12 illustrates the following common activation functions:

Tanh

$$h(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{3.30}$$

The *tanh* activation function satisfies the above conditions and is symmetrical about zero (see Figure 3.12a). This is desirable, as it means that gradients do not shift in a particular direction. However, it also suffers from the *vanishing gradient problem*. Because the output of the activation function is limited to between -1 and 1, at the ends of this range a large change in the input of the function causes only a small change in the output (i.e. the gradient is close to zero). For shallow networks this is not such a big problem, but for deeper networks successive backpropagation through the layers can lead to gradients becoming very small in the initial layers of the network. This means the weights and biases in these layers cannot be updated effectively, impacting the performance of the model.

Rectified Linear Unit

$$h(x) = \max(0, x) \tag{3.31}$$

An alternative which does not suffer from the vanishing gradient problem is the rectified linear unit (ReLU) activation function (see Figure 3.12b). It is also computationally inexpensive, which, given activation functions typically need to be calculated millions of times in deep networks, is an important consideration. It is the default recommendation for modern neural networks [73]. One issue with ReLU is that large weights can result in the activation function input at a node always being negative, regardless of the input into the network. This leads to a node that will forever output zero, called a *dying ReLU* node.

Leaky ReLU

$$h(x) = \max(\varepsilon x, x) \tag{3.32}$$

Leaky ReLU attempts to solve the dying ReLU problem by adding a small negative gradient (ε) for inputs less than zero (see Figure 3.12c). The parameter ε is user-defined, but generally set to 0.01.

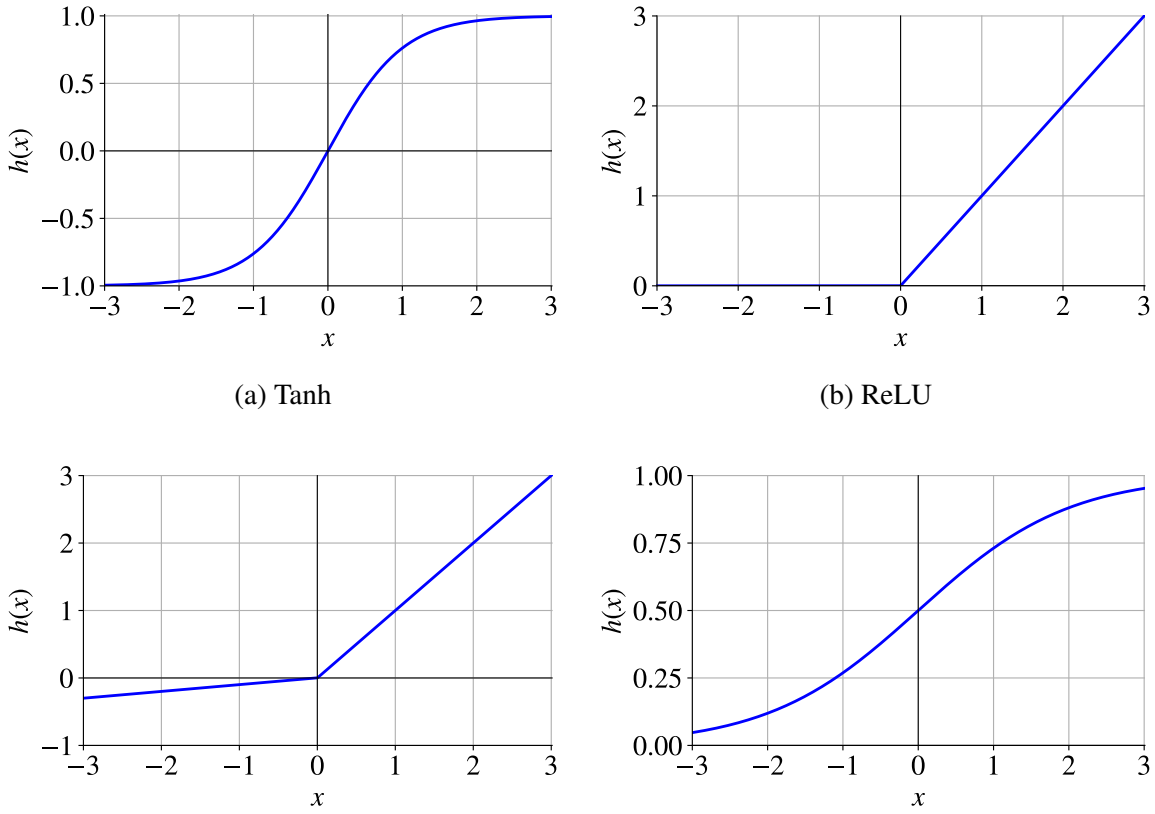
Softmax

$$h(x)_i = \frac{e^{x_i}}{\sum_{j=1}^{K} e^{x_j}} \text{ for } i = 1, ..., K \tag{3.33}$$

In a K-class classification problem the final layer of a network typically consists of K nodes, each corresponding to one of the classes. The *softmax* activation function normalises the outputs proportionally to the exponentials of their inputs so they sum to one (see Figure 3.12d). This means that they can be used to calculate cross-entropy loss. These are often interpreted as measures of model confidence, however this is not the case and these outputs need to be treated with caution [74] [75]. For binary classification problems this reduces to a *sigmoid* function.

3.2.4 Training a Neural Network

When optimising a neural network there are millions of parameter (often called *hyperparameter*) combinations. The number of layers, nodes per layer, loss function, activation function, optimisation algorithm, learning rate, and batch size can all significantly affect the model performance [76] [77] [78]. Frustratingly, there are few hard rules as to which combinations of parameters will work well or not so well. Numerous hyperparameter optimisation algorithms have been developed to automate this procedure [79] [80], but it is also important to think carefully about the problem. Existing model architectures e.g. from studies of similar problems or datasets can be good starting points. To understand why a model is failing it is useful to plot learning curves. Learning curves plot the loss (or another performance



(a) Tanh

(b) ReLU

(c) Leaky ReLU

(d) Softmax (2 classes)

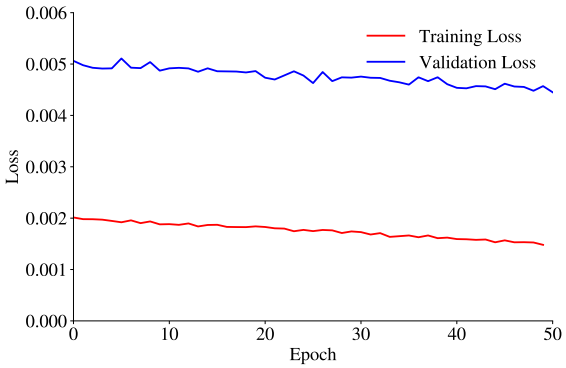
Fig. 3.12 Graphs of four common activation functions

metric e.g. accuracy) of the training and validation data across epochs. Figure 3.13 illustrates some schematic examples showing typical problems encountered when training. In Figure 3.13a we see an underfitting case where the training loss is not decreasing. This is common when the model does not have the capacity to capture the complexity of the dataset and probably indicates that a more complex model is needed. Figure 3.13b shows another case of underfitting. The loss is still decreasing at the end of training, indicating that further reduction would be possible with more epochs. Figure 3.13c shows the learning curve for a model that has overfit the data. The training data is being perfectly fit, but this is not generalising to the validation data. In this case either a simpler model or *regularisation* may be required. Regularisation puts constraints on models to discourage excessive complexity. Some examples are discussed in Chapter 4. *Early stopping* could also be used to stop training earlier (at around epoch ten), before overfitting starts to occur. Figure 3.13d demonstrates an example of a learning curve with unrepresentative validation data. Although the training loss behaves as expected, the validation loss is noisy, meaning it does not provide a reliable indicator of performance. More validation data may be needed. Finally, Figure 3.13e shows a good fit. Both the training and validation loss decrease to a stable value, and the validation loss is similar to the training loss.

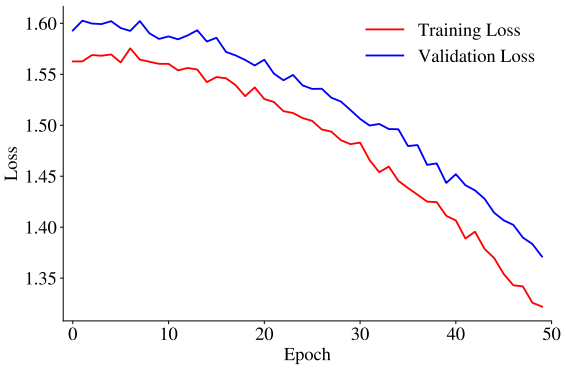
3.3 Feature Selection

If we have a large amount of features relative to our dataset size we are in danger of overfitting our data. To avoid this we can reduce the amount of features by using *feature selection*. Reducing the number of features also simplifies the model, making it easier to interpret and speeding up training. Various techniques exists and there is no best feature selection method [81] [82] [83]. They can be *unsupervised*, meaning they do not use the target variable (e.g. correlation to remove redundant features), *wrapper methods*, which evaluate multiple models using different feature subsets (e.g. recursive feature elimination), or *filter methods*, which rank features based on some measure against the target variable (e.g. correlation with the target). The appropriate method depends on whether the input and output data are numerical or categorical. Here we mention a few feature selection methods relevant to this thesis:

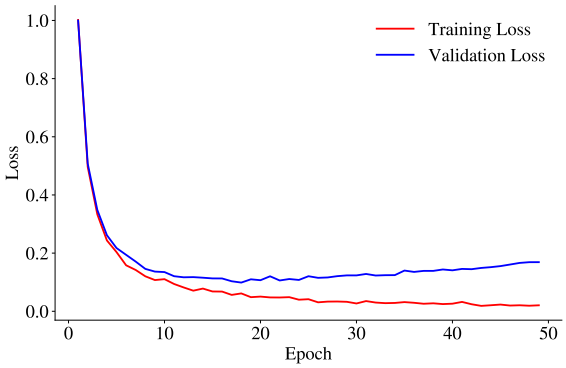
Correlation of Features If the input data are numerical and two features are correlated above a threshold there is a level of redundancy, which can impact model stability and interpretability [84]. One solution is to remove correlated features, an example of unsupervised feature selection. To choose which features to remove, the mean absolute correlation with respect to the whole feature set is calculated, and the feature with the largest mean absolute



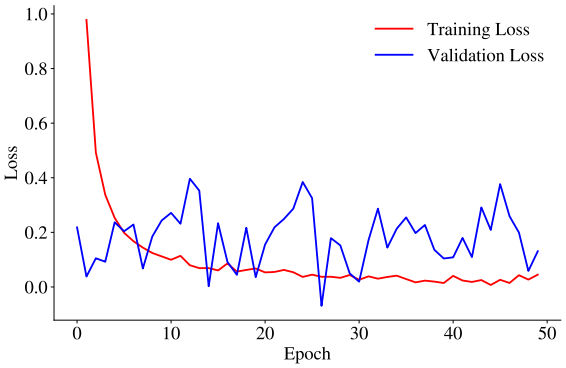
(a) Underfitting (no learning)



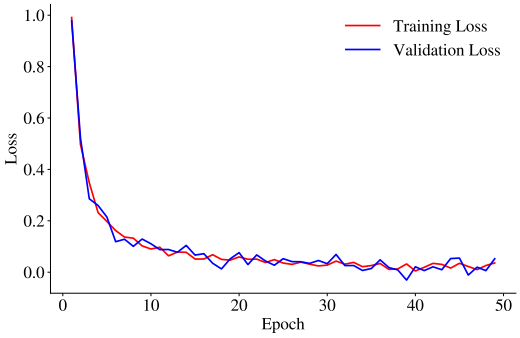
(b) Underfitting (insufficient training)



(c) Overfitting



(d) Unrepresentative validation set



(e) Good fit

Fig. 3.13 Schematic learning curves showing common problems when training neural networks

Table 3.1 Confusion matrix explaining the TP, TN, FP, and FN values. This is used to derive the evaluation metrics in this section

	Actual Positive	Actual Negative
Predicted Positive	TP	FP
Predicted Negative	FN	TN

correlation is removed. Care needs to be taken to ensure that the deviation from perfect correlation does not encode information relevant to the problem.

Univariate Analysis The data are split into subsets based on the target variable. Univariate analysis is performed to test the significance of the difference between the subsets for each feature. Features without significant differences between the subsets are removed. The disadvantage of univariate analysis is that important information relying on multivariate analysis may be missed. Common statistics used are the analysis of variance (ANOVA) F-test for numerical input and output data and the χ^2 test for numerical input and categorical output data. These are filter methods.

Recursive Feature Elimination Recursive feature elimination (RFE) works by recursively removing features, building models, and discarding the least important features. This process is repeated until the desired amount of features remain. The model could be any machine learning model (e.g. SVM RFE). This is an example of a wrapper method.

3.4 Evaluation Metrics

To effectively train and test a model it is important to have an evaluation metric relevant to the problem. Here we list and explain the metrics relevant to this thesis, referencing the confusion matrix in Table 3.1. We use the abbreviations TP (True Positives), TN (True Negatives), FP (False Positives), and FN (False Negatives).

Accuracy Accuracy gives the percentage of correct predictions:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.34}$$

Sensitivity Sensitivity is the percentage of positive cases correctly labelled as positive. It is also called the true positive rate or recall:

$$Sensitivity = \frac{TP}{TP+FN} \tag{3.35}$$

Specificity Specificity is the percentage of negative cases correctly labelled as negative. It is also called the true negative rate:

$$Specificity = \frac{TN}{TN+FP} \tag{3.36}$$

Youden Index The Youden Index (or Youden's J Statistic) is defined as:

$$Y = Sensitivity + Specificity - 1 \tag{3.37}$$

Precision Precision is the proportion of predicted positives that are true positives. It is also called the positive predictive value (PPV):

$$Precision = \frac{TP}{TP+FP} \tag{3.38}$$

Negative Predictive Value The negative predictive value (NPV) is the proportion of predicted negatives that are true negatives:

$$NPV = \frac{TN}{TN+FN} \tag{3.39}$$

Receiver Operating Characteristics Curve The sensitivity and specificity can be seen as opposing metrics. If we decrease the threshold required for a result to be classed as positive we will increase the sensitivity (as we are more likely to identify positive cases), but decrease the sensitivity (as we will now have more false positives). The receiver operating characteristics (ROC) curve plots sensitivity vs (1-specificity) at several thresholds, demonstrating this trade-off. An example is shown in Figure 3.14. Often in medical cases we would set the threshold to favour a high sensitivity, as we want to minimise the FNs. The area under the ROC curve (AUC) is commonly used to measure performance independent of this threshold choice. An AUC of 0.5 represents random guessing and an AUC of 1 perfect classification.

Free-Response ROC Curve The free-response ROC (FROC) curve is similar to the ROC curve, except that (1-specificity) on the x-axis is replaced by the number of false positives per

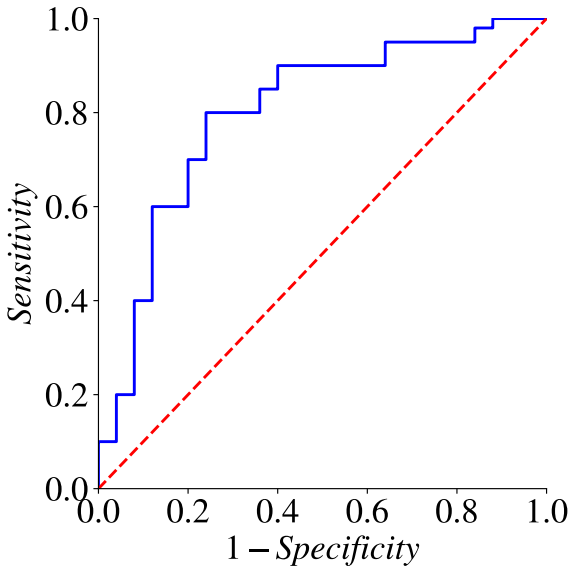


Fig. 3.14 ROC AUC curve. The blue line shows an example ROC curve with an AUC of 0.8. The red line is the reference line representing random guessing (AUC of 0.5)

image. It is often used in medical applications to compare results obtained by an algorithm to those by an expert [85] [86].

Cohen's Kappa Coefficient Cohen's kappa coefficient (κ) is a measure of the agreement between several classifications of the same set of data (e.g. two radiologists classifying a set of patients) [87]. It is defined as

$$\kappa = \frac{p_o - p_e}{1 - p_e} = 1 - \frac{1 - p_o}{1 - p_e} \tag{3.40}$$

where p_o is the relative observed agreement among classifications (equivalent to accuracy) and p_e is the probability of chance agreement

$$p_e = \frac{1}{N^2} \sum_k n_{k1} n_{k2} \tag{3.41}$$

where k is the number of categories, N is the number of observations, and n_{ki} is the number of times observer i predicted category k.

Standard coefficient benchmarks taken from Landis and Koch are: 0.81–1 almost perfect agreement, 0.61–0.8 substantial agreement, 0.41–0.6 moderate agreement, 0.21–0.4 fair agreement [88].

Jaccard Index The Jaccard Index is a measure of the similarity between two samples. In medical imaging it can be used to calculate the similarity between two regions (e.g. comparing an algorithmic segmentation against a reference segmentation) and is defined as:

$$\text{Jaccard Index} = \frac{|X \cap Y|}{|X \cup Y|} \tag{3.42}$$

where X and Y are the two segmentations.

Sørensen–Dice coefficient Similar to the Jaccard index, the Sørensen–Dice coefficient (or *DICE Score*) also measures the similarity between two regions. It is defined as

$$DICE = \frac{2|X \cap Y|}{|X| + |Y|} \tag{3.43}$$

The DICE score ranges from zero, representing no agreement, to one, representing perfect agreement. It is not a true metric, as it does not satisfy the triangle inequality (unlike the Jaccard Index), but has become the standard for comparison of medical segmentations [89]. [90] showed that the use of the DICE Score or Jaccard Index as a loss function does not make a significant difference in several medical segmentation applications. Given the DICE score's ubiquity, we use this in our work.

Confidence Intervals Confidence intervals (CIs) give the confidence level that the value of a parameter is within the proposed range. For example, a CI of 95% indicates that there is a 95% chance that a parameter is within the defined range. We use exact Clopper-Pearson confidence intervals in this thesis [91].

Coefficient of Variation The coefficient of variation c_v is a measure of the variability of a distribution with respect to the mean

$$c_v = \frac{\sigma}{\mu} \tag{3.44}$$

where μ is the mean and σ is the standard deviation.

Pearson Correlation Coefficient The Pearson correlation coefficient (r) tests the correlation between two continuous variables

$$r = \frac{\sum(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum(x_i - \bar{x})^2 \sum(y_i - \bar{y})^2}} \tag{3.45}$$

where x_i and y_i are the values of the two variables and $\bar{x}$ and $\bar{y}$ are their respective means.

Student's t-test The Student's t-test is used to test the null hypothesis that the means of two sets of data are equal. It assumes that the means follow normal distributions and that the populations have the same variance and number of data. The test statistic t is defined as

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sigma\sqrt{\frac{2}{n}}} \tag{3.46}$$

where n and σ are the number of data and standard deviation of each population, and $\bar{x}_1$ and $\bar{x}_2$ are the means of the two populations. This and the degrees of freedom ($2n-2$ for the student's t-test [92]) can be used to determine the p-value (the probability of obtaining a result at least as extreme as the result observed [93]) using a table of values as in [94].

Welch's t-test Welch's t-test is a modified version of the Student's t-test for two populations that have unequal variances and/or unequal sample sizes [95]. Welch's statistic t is defined as

$$t = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}} \tag{3.47}$$

and the associated degrees of freedom ν are approximated by

$$\nu \approx \frac{(\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2})^2}{\frac{\sigma_1^4}{n_1^2\nu_1} + \frac{\sigma_2^4}{n_2^2\nu_2}} \tag{3.48}$$

where n_i and σ_i are the number of data and standard deviation of each population, and $\nu_i = n_i - 1$. We can then use t and ν to test the null hypothesis (and find the p-value).

ANOVA F-test The ANOVA F-test is used when more than two populations are being compared [96]. The ANOVA F-test statistic is defined as

$$F = \frac{\text{between-group variability}}{\text{within-group variability}} \tag{3.49}$$

where the between-group variability is

$$\sum_{i=1}^{K} \frac{n_i(\bar{x}_i - \bar{x})^2}{K-1} \tag{3.50}$$

where $\bar{x}_i$ denotes the sample mean of the i^{th} group, n_i is the number of data in the i^{th} group, $\bar{x}$ denotes the overall mean of the data, and K denotes the number of groups.

The within-group variability is defined as

$$\sum_{i=1}^{K} \sum_{j=1}^{n_i} \frac{(x_{ij} - \bar{x}_i)^2}{N - K} \tag{3.51}$$

where x_{ij} is the j^{th} datapoint in the i^{th} out of K groups and N is the overall population size.

The ANOVA F-statistic will be large if between-group variability is large compared to within-group variability.

Chapter 4

Convolutional Neural Networks

Regular neural networks as described in 3.2.3 use fully connected layers, meaning every neurone in one layer is connected to every neurone in the next. However, this does not scale well to images. For example, one layer of 16 neurons with an input image of 256x256 pixels would lead to that layer having over one million weights. Given most architectures have several layers, this quickly leads to a network of intractable size.

Convolutional neural networks (CNNs) are a type of neural network designed for images, which exploit spatial relationships between data. Rather than being fully connected, neurones are only connected to local regions of the input data. This significantly reduces the number of parameters, making them well suited to image-based problems.

One of the first demonstrations of a CNN was in 1998, when LeCun et al. achieved success using their LeNet-5 network for handwritten document recognition [97]. However, CNNs still required huge amounts of computational power for high resolution images. Because of this they spent the next 14 years in the academic wilderness, before bursting back onto the scene when AlexNet won the ImageNet large scale visual recognition challenge (ILSVRC) in 2012[1] [98]. Faster training was made possible by a GPU-based implementation, and a ReLU activation function, data augmentation, and dropout led to increased performance.

Since then CNNs have blossomed, with networks becoming larger and deeper (i.e. having more layers) as computer power increases. They have surpassed experts in many radiology, histopathology, and dermatology applications, and look set to be at the centre of the trend towards computer-assisted medicine [48] [49] [50].

[1] The ImageNet database that this challenge is based on consists of over 14 million natural images categorised into over 20,000 categories and is the de facto benchmark for computer vision classification algorithms

4.1 Basic Principles

A key difference between CNNs and the algorithms described in Chapter 3 (*traditional* machine learning algorithms) is that CNNs use images as direct inputs into the network, whereas the traditional algorithms use features preextracted from the images (e.g. radiomic features). CNNs are extracting information from the images completely independently, without any priors or human-derived constraints. This gives CNNs more flexibility and power, but leads to issues of interpretability (see 4.4).

The basic mechanism of training a CNN remains the same as for regular neural networks, using forward and backpropagation to iteratively update the weights. Images are usually rescaled before training, either so all values lie between zero and one or by negating the mean and dividing by the standard deviation (so the mean is zero and the standard deviation is one). There is no consensus on which method is better [99]. Networks are built of several different types of layer, described below:

4.1.1 Convolutional Layer

A convolutional layer consists of a set of learnable filters (*kernels*), which are typically small compared to the image. Each filter is convolved with the input image to give an output image representing the response to that filter (see Figure 4.1). Therefore, a convolutional layer with N_c filters acting on an image will produce N_c convolved images. The weights of the filters are trainable, so during training the filters adapt to pick out features useful for the problem. As shown in Figure 4.1, because of edge effects the output image will be smaller than the input image (by $S_f - 1$ pixels, where S_f is the size of the filter). To control the size of the output image, it can be useful to use *zero-padding*. To do this the images are padded by adding zeros around the outside, such that the input and output size are the same. The *stride*, the number of pixels moved at each step during the convolution, can also be adjusted. With a stride of one we convolve with the filter at each source pixel, and thus our output image is the same size as our input image (with zero-padding). With a stride of two we would produce an output image half the dimension of the input image (with zero-padding).

Because the filters are small they have a small *receptive field*, meaning each filter is only connected to a small region of the input image. Features extracted in early layers are therefore lower-level, such as edges and colour gradients. With successive convolutional layers this receptive field grows, and later convolutional layers learn more complicated, large-scale features from the images [100].

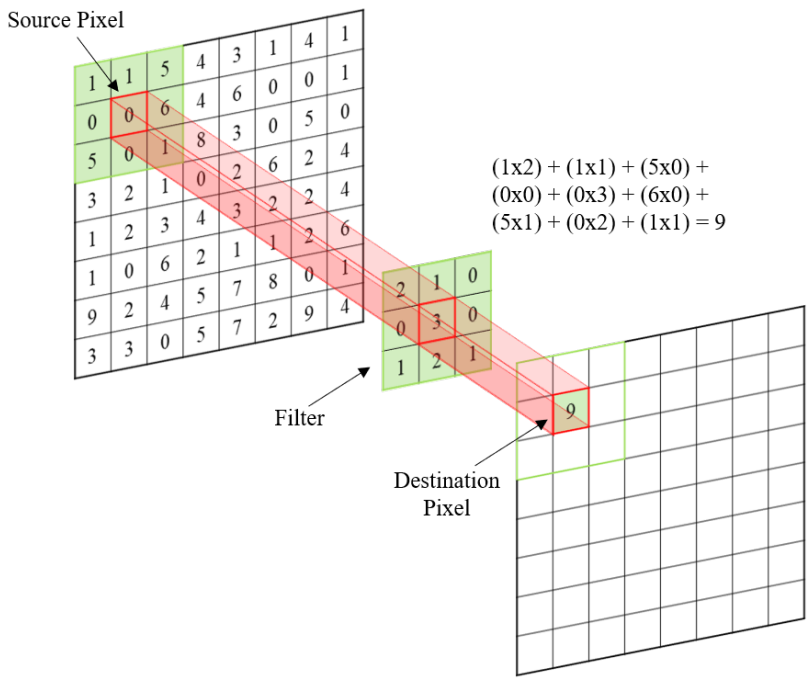


Fig. 4.1 Example of a convolutional layer. The filter is convolved with the source pixel (i.e. the dot product is taken) to give the destination pixel value. This is repeated across the input image to give the output image. Each convolutional layer will have many filters, each with independently trainable weights

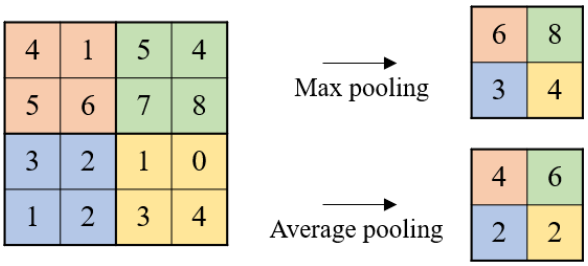


Fig. 4.2 Examples of max pooling and average pooling operations with filter size two and stride two

4.1.2 Pooling Layer

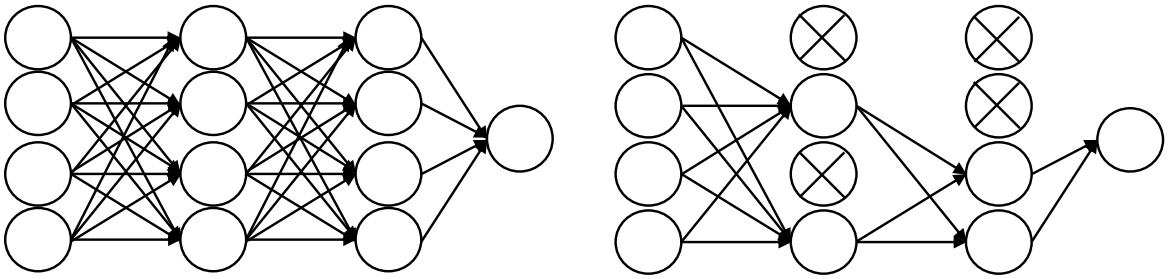
Pooling layers are often placed in between convolutional layers to progressively reduce the size of the image. This reduces the amount of parameters in the network, speeding up computation and helping to control overfitting. Pooling layers slide a filter over the image, typically applying a *max* or *average* operation at each position. Again, the size and stride of the filter can be specified. Figure 4.2 demonstrates these pooling operations with filter size two and stride two. Although near-ubiquitous in current research, some recent studies have questioned their utility given the amount of information discarded and have proposed alternatives or even removed them completely [101] [102] [103].

4.1.3 Batch Normalisation Layer

Batch normalisation was first introduced in [104], where the authors improved on the top result in the 2014 ILSVRC using only a fraction of the training steps. Batch normalisation scales the outputs of hidden layers to have zero mean and unit variance, on a batch-by-batch basis. This typically leads to quicker training and potentially acts as a form of regularisation [105]. Although originally devised to reduce internal covariate shift, later studies have shown that this is not the case, and the reason batch normalisation works is still unclear [105] [106]. Batch normalisation layers are usually put after the activation layers.

4.1.4 Dropout Layer

Dropout layers act as a form of regularisation by dropping out (i.e. removing) a random set of neurones from that layer. This means that in forward propagation their contribution to further activations is removed, and in backpropagation weight updates are not applied to these neurones. This helps regularise the network by enforcing redundancy and making the network less sensitive to specific neurone weights (see Figure 4.3) [107]. The proportion



(a) Network before dropout

(b) Network after dropout of 50% in the two hidden layers

Fig. 4.3 Visualisation of a network before (a) and after (b) dropout

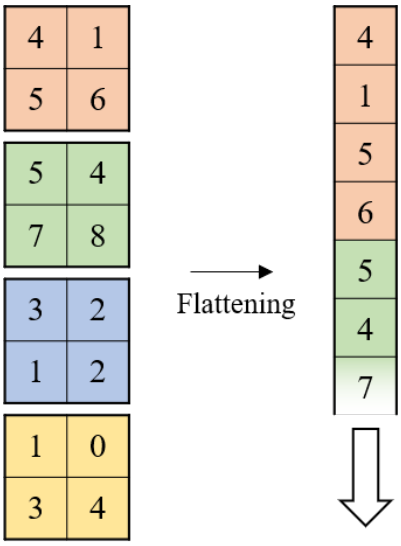
of nodes removed is set by the user. Dropout is only used during training, not evaluation. Dropout layers can also be used in regular neural networks.

4.1.5 Fully Connected Layer

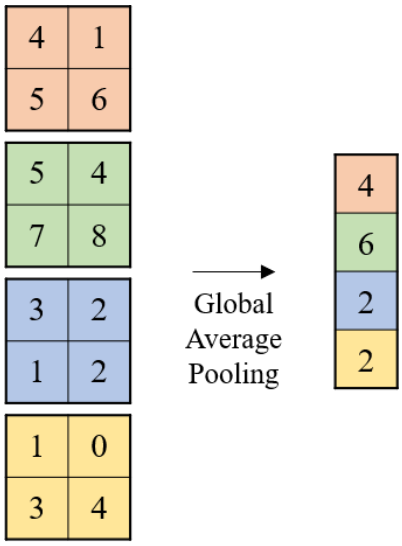
Fully connected layers are often used at the end of networks and are akin to the regular neural networks described in 3.2.3. The outputs from the convolutional part of the network are converted into a 1D vector, either by simple flattening or by using a global average pooling operation (see Figure 4.4 for a visual explanation of both processes), and passed through a series of fully connected layers before classification.

4.1.6 Full Network Architecture

An example of a typical CNN architecture, *VGGNet* (which won the ILSVRC in 2014), is shown in Figure 4.5 [108]. It consists of a series of convolutional and max pooling layers, followed by fully connected layers. The input size is 224x224x3 pixels (representing three colour channels), and the output of the first set of convolutional layers is 224x224x64 pixels (there are therefore 64 filters in the first convolutional layer). After each pooling layer the spatial dimension halves and the number of filters doubles (this is convention, not a requirement). There is no reduction in dimension after convolutional layers, showing that zero-padding is being used. ReLU activation functions are used throughout, with a final 1000-node softmax layer to normalise the outputs (as there are 1000 classes).



(a) Flattening. Every individual value from the previous layer is combined to create one 1D vector



(b) Global average pooling. The average of each filter map is taken to create a 1D vector with length equal to the number of filter maps in the previous layer

Fig. 4.4 Flattening and global pooling examples. The four 2x2 matrices in each image represent four filter maps in a convolutional layer. These are converted into 1D vectors by (a) flattening or (b) global average pooling. Fully connected layers can be added after this conversion

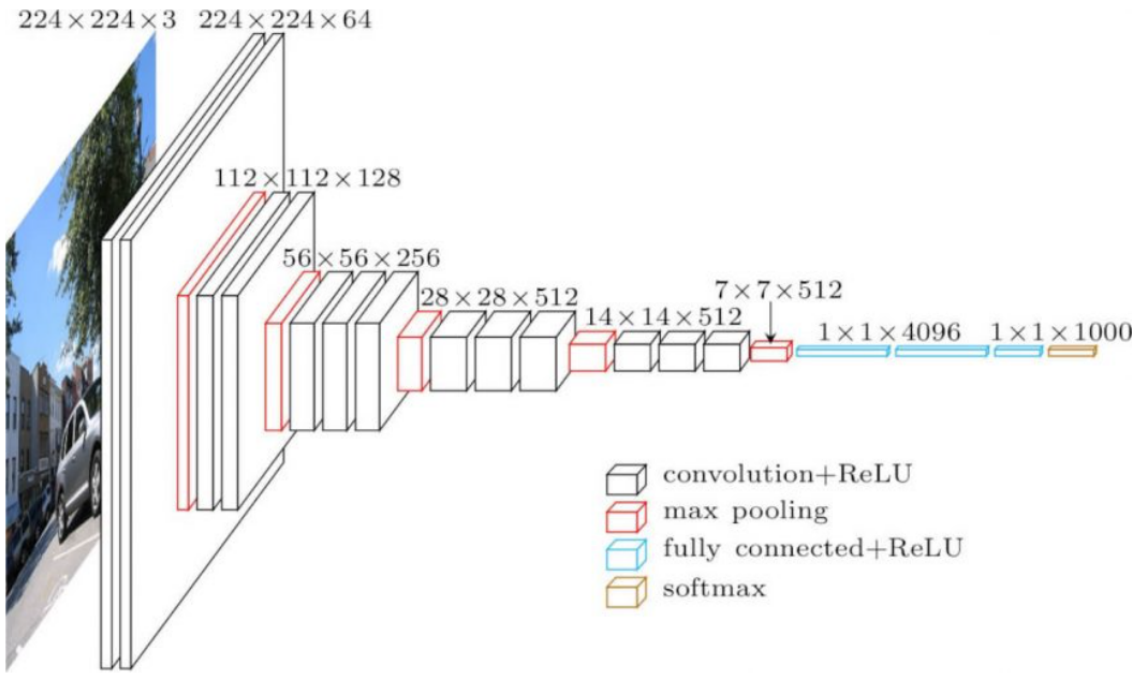


Fig. 4.5 VGGNet architecture, which won the 2014 ILSRVC challenge. Taken from [109]

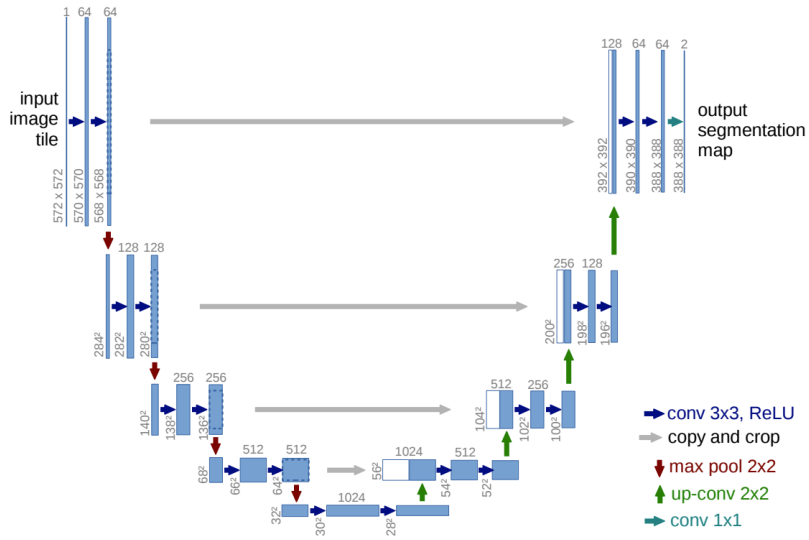


Fig. 4.6 Architecture of U-Net, which won the 2015 ISBI cell tracking challenge. Taken from [111]

4.2 State-of-the-Art Networks

4.2.1 U-Net

Segmentation is an important part of medical imaging processing and analysis, crucial to ensuring the relevant ROIs are identified [110]. Manual segmentation is time-consuming and prone to inter and intraobserver bias, which has led to deep learning-based alternatives becoming very popular. U-Net is one of the most popular segmentation models, having won the International Symposium on Biomedical Imaging (ISBI) cell tracking challenge in 2015 and since been used in hundreds of papers and applications [110] [111] [112]. The architecture is shown in Figure 4.6. For segmentation the output must be the same size as the input, so the network consists of two parts. The left part is a *contracting path*, which is a typical convolutional architecture consisting of a series of convolutional and pooling operations, as in VGGNet above. The right part is an *expansive path*, consisting of a series of upsampling operations. To maintain spatial context, at every upsampling operation the output is concatenated with the outputs from the corresponding downsampling layer (the grey arrows).

4.2.2 ResNet

Deeper networks generally perform better, but training very deep networks is hindered by the vanishing gradient problem (which leads to early layer weights not being updated) and *degredation* [113] [114] [115]. ResNet solves this problem by allowing gradients to flow

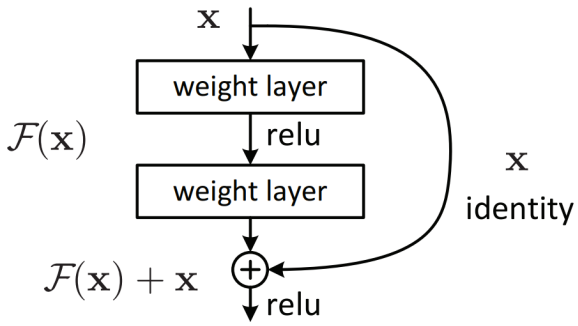


Fig. 4.7 Residual learning block of ResNet, showing a skip connection. Taken from [115]

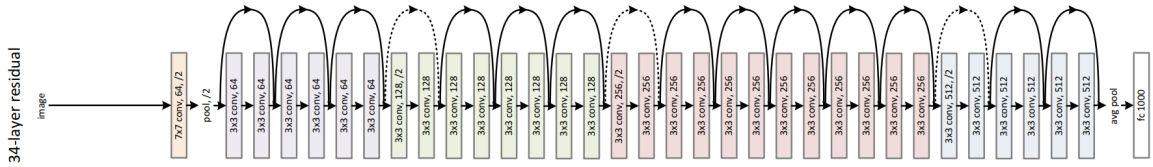


Fig. 4.8 34-layer ResNet architecture showing the skip connections. Taken from [115]

directly through *skip connections* (see Figure 4.7). The original ResNet paper tested networks with different numbers of layers (see Figure 4.8 for the 34-layer ResNet architecture) and achieved first place in the 2015 ILSVRC classification task [115].

4.2.3 Inceptionv3

Released by Google in 2015, Inceptionv3 aimed to increase performance while utilising computational power as efficiently as possible. The network uses multiple filter sizes at each layer, factorises larger filters into smaller and asymmetrical filters, and uses label smoothing for regularisation. Full details can be found in the original paper [116]. The model demonstrated substantial performance gains on the ILSVRC 2012 classification challenge despite having far fewer parameters than networks such as VGGNet. The Inceptionv3 architecture is shown in Figure 4.9.

4.3 Dealing with Data Scarcity

Most popular computer vision datasets consist of tens of thousands (or more) images, and well-known networks such as the above are trained on these huge datasets. But in the medical imaging domain we are not so lucky. Typically medical datasets are of the order of hundreds of images [118]. Different techniques have been developed to deal with this paucity of data:

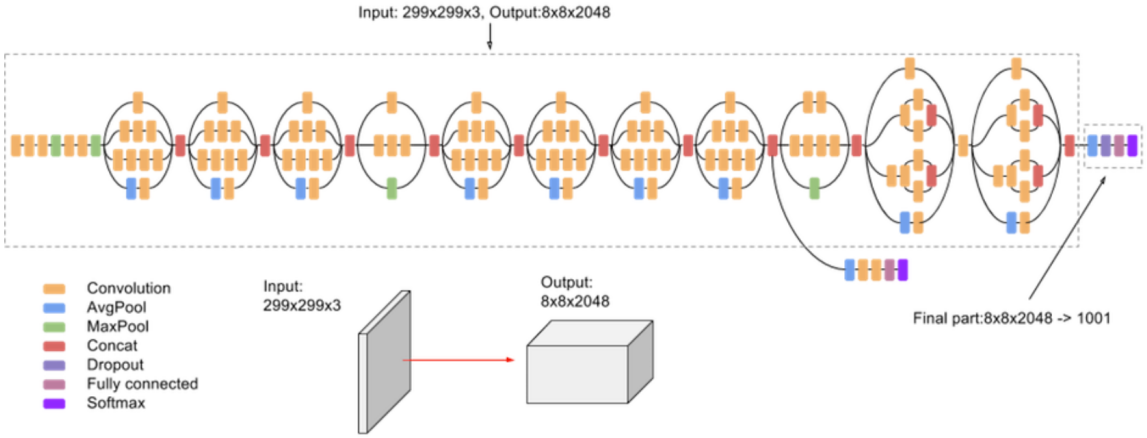


Fig. 4.9 Inceptionv3 architecture. The architecture is complicated, with many parallel convolutions at different filter sizes. Taken from [117]

4.3.1 Data Augmentation

Data augmentation uses transformations such as flips, translations, and addition of random noise during training, to increase the dataset size. It is commonly used to improve performance on medical imaging tasks [119] [120] [121].

Completely new synthetic images have also been used for augmentation, created using general adverserial networks (GANs) [122] [123]. These are not used in this thesis so are not discussed further.

4.3.2 Transfer Learning

Transfer learning (TL) involves applying the weights of a network pretrained on a different task (ideally one with a large amount of data available e.g. ImageNet) to a task for which one has fewer data. While the tasks may differ, low-level features learnt in the initial layers (such as edges and lines) will be common to both (although recently this has been called into question [124] [125] [126]). The network is then *fine-tuned*, often with initial layers' weights frozen. The amount of layers frozen depends on how similar the two tasks are; if the tasks are very similar the features should be similar and most of the layers should be frozen, if the tasks are quite different most layers should be left unfrozen. Studies have shown a benefit to transfer learning even between the very different domains of natural images (e.g. ImageNet) and medical images [127] [128] [129] [130]. TL has also been used to adapt models for use with data from different scanners or from different medical centres [131] [128] [132].

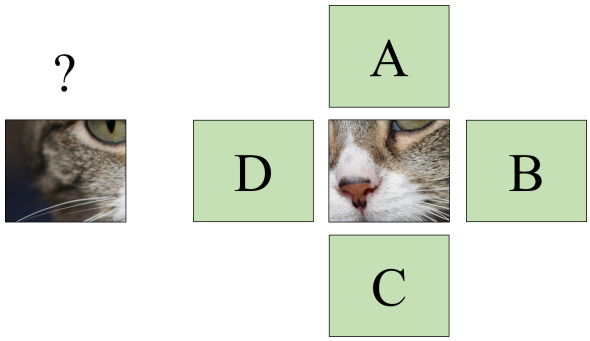


Fig. 4.10 Self-supervised learning jigsaw solving problem. The CNN is trained to predict at which orientation (A, B, C, or D) the patch should be placed. Random gaps between the patches increase the difficulty

4.3.3 Self-Supervised Learning

Sometimes there is not a comparable large database to TL with (e.g. 3D medical images). In these cases self-supervised learning can be used to pretrain a CNN using the same dataset, with a task that does not require the data to be labelled [133]. Examples include colourising black and white images [134] [135], inpainting [136], and jigsaw solving problems [137]. In this thesis we use the jigsaw solving problem described in [137]. A 2D example is shown in Figure 4.10. Two patches at random orientations to each other are chosen from the image, and the CNN has to predict their orientation (in 3D there would be six possible orientations). Random displacements (*jitter*) of each patch by a few pixels can be used to increase the difficulty of the task. Training the CNN using this task therefore allows the CNN to learn features from the images without the need for labelling.

4.4 Interpreting CNNs

One of the most problematic features of CNNs, particularly in the medical domain, is their lack of interpretability. With millions of weights in the hidden layers, CNNs are often found in the same sentence as the undesirable term *black box*. Without understanding how CNNs make their decisions, doctors will not be confident using them in clinical situations. This opacity also means that we do not learn from the CNNs, so cannot use them to further our biological understanding of problems.

A plethora of methods for interpreting CNNs have been proposed in recent years [138] [139] [140]. *Visualisation methods* express a feature of the CNN as an image, which can then be interpreted by a human user. These are often in the form of *saliency maps*, which highlight regions of the input image salient to the CNN's decision. This can be achieved

through backpropagation of gradients (e.g. gradient explanation [141], Smoothgrad [142], guided-backprop [103], grad-CAM [143], or DeepLIFT [144]), or by perturbation methods (e.g. occlusion sensitivity [100] or meaningful perturbation [145]). *Distillation methods* build models that mimic the CNN's behaviour but with inherently interpretable features (e.g. [146] [147] [148], which link CNN feature maps to semantic features from images). Models can also be built to be *intrinsically interpretable*, outputting an explanation of the decision as well as the decision itself. Examples include joint training methods, which add an additional explanation-based task during CNN training [149].

Despite this abundance of methods, the reliability and utility of CNN interpretation methods have been questioned. Many of the more popular approaches have been shown to not meet key criteria such as sensitivity, comprehensibility, consistency, and utility [150] [151] [152]. Any conclusions should therefore be treated with caution and if possible compared with other indicators or expert opinion. In this thesis we use gradient-weighted class activation mapping (grad-CAM) and occlusion sensitivity methods:

4.4.1 Grad-CAM

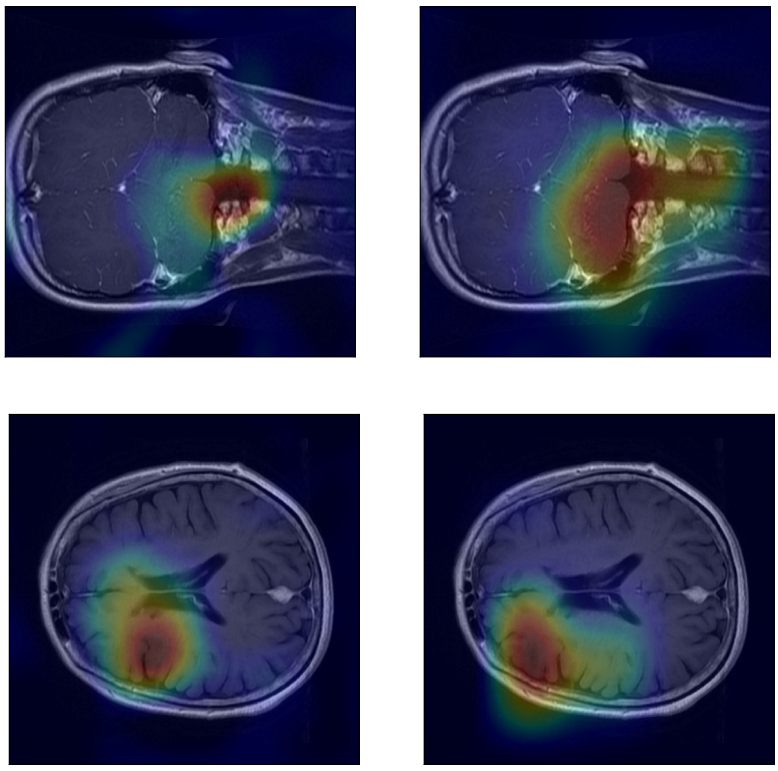
Grad-CAM is one of the most popular methods for CNN interpretation and generally seen to be reliable [153] [154]. For a class c, Grad-CAM first computes the gradient of the output neurone y_c (before the softmax function) with respect to the feature maps A^k of the final convolutional layer. These gradients are then global-average-pooled to obtain the importance of each of these feature maps α_k^c. For the two-dimensional case this gives

$$\alpha_k^c = \frac{1}{N}\sum_{i,j}\frac{\partial y^c}{\partial A_{ij}^k} \tag{4.1}$$

where i and j are the two dimensions and N is the number of pixels in each feature map. The feature maps are then summed to give location-based heatmaps, using α_k^c as weights

$$L_{ij}^c = \text{ReLU}(\sum_k \alpha_k^c A_{ij}^k) \tag{4.2}$$

This gives heatmaps with the same dimensions as the final convolutional layer. The ReLU is applied so that only positive contributions to the gradient are considered. Some examples are shown in Figure 4.11. We see high intensity areas centred on the tumours, indicating that these areas have a large influence on the output classification. Because Grad-CAM uses the final convolutional layer, the resulting heatmap is quite coarse, lacking finer pixel-level details.



Grad-CAM maps Occlusion sensitivity maps

Fig. 4.11 Examples of Grad-CAM and occlusion sensitivity maps for a CNN trained to classify tumours in T2-weighted brain MRI images, overlaid on the original MRI images. The images on the left show two Grad-CAM images, with the higher intensity regions in red showing where the CNN is focusing. The images on the right are the corresponding occlusion maps. The red areas indicate a drop in the prediction when that area is covered (i.e. that area is important for the classification)

4.4.2 Occlusion Sensitivity

Another way of determining which part of the image the CNN is using is by occluding part of the image (i.e. setting a part of the image, usually a square or cube, to zero). The probability of a certain class can then be plotted as the occluding region is moved across the image, at each point running the image through the CNN. The size of the occluding region will depend on the task. The stride (how much to move the region at each test) is also defined by the user and affects the time taken to run. Some examples are shown in Figure 4.11. The probability is low when the tumour region is occluded, indicating that this area is important to the classification.

Chapter 5

Automated Detection of Pathological Mediastinal Lymph Nodes Using Deep Learning

5.1 Problem Background

The mediastinum is the central area of the thorax, surrounded by the two lungs. It contains vital structures including the heart, great vessels, trachea, and essential nerves, as well as the lymph nodes of the central chest (see Figure 5.1) [155]. These lymph nodes are important for the proper functioning of the immune system, and inflammation or enlargement of the nodes can indicate the presence of disease. Lymph nodes within the chest are labelled for medical purposes, with node labels 1-9 indicating mediastinal lymph nodes (see Figure 5.2).

Lung cancer is the most commonly diagnosed cancer and the leading cause of cancer death worldwide, with non-small-cell lung cancer (NSCLC) comprising more than 85% of these cases [158] [159]. NSCLC typically metastasises to the hilar (lymph nodes on the border of the mediastinum) and mediastinal lymph nodes, and the presence of metastases significantly impacts the staging, prognosis, and patient management. Five-year survival rates are 54% for patients without any metastases and 27% for patients with mediastinal metastases [160]. Tumour progression, prognosis evaluation, and decisions on treatment plans are mainly dependent on this staging, and thus it is critically important to accurately detect the mediastinal cancer nodes.

Currently an FDG-PET/CT scan is acquired and the nuclear medicine physician/radiologist examine all slices (see Figure 5.3 for some examples), which is time-consuming and prone to errors. This slice-by-slice reading also limits analysis to 2D planes, losing potentially useful

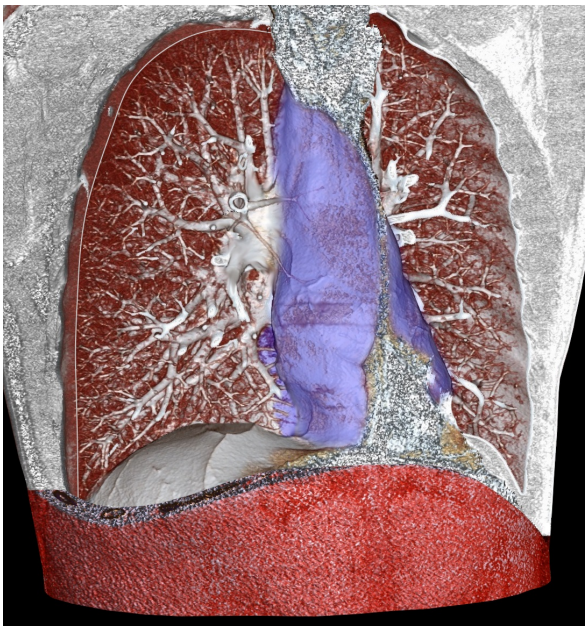


Fig. 5.1 Artificially coloured 3D rendering of a chest CT. The mediastinum is highlighted in blue. Taken from [156]

3D information. From a meta-analysis the sensitivity and specificity of lesion-based analyses have been shown to be 0.59 and 0.97 respectively [1]. This means that a large number of nodes are false-negatively judged. Additionally, the agreement between experienced observers is limited. Inter-observer agreement, defined by the kappa score (κ), has been shown to range from 0.48-0.88, depending on the type of node (agreement was lower for aortopulmonary nodes (κ=0.48-0.55) but higher for the inferior and superior nodes (κ=0.71-0.88)) [161]. To improve the sensitivity and reliability of mediastinal lymph node staging, a more sophisticated classification system, such as one utilising machine learning algorithms, is needed.

5.2 Previous Work

As mentioned in Chapter 4, deep learning and CNNs have led to huge excitement in the medical community. CNNs are being proposed to solve problems across the breadth of medical research, and lung cancer is no exception. There are numerous studies that use CNNs to detect pulmonary nodules on CT scans, often using large datasets [163] [164]. [165] used a cohort of 14,851 patients from the publicly available National Lung Screening Trial to predict risk of lung cancer, achieving an AUC of 0.94 and outperforming radiologists.

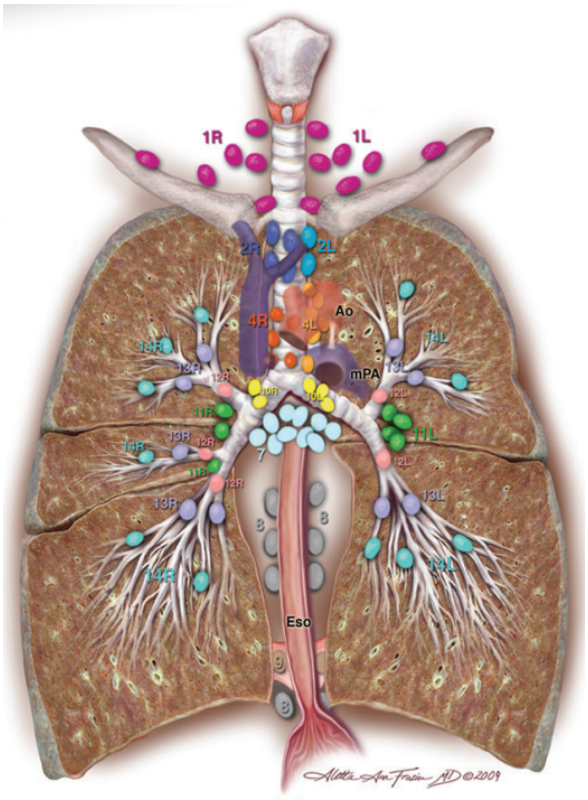


Fig. 5.2 International Association for the Study of Lung Cancer Nodal Chart (8th ed.). Nodes numbered 1-9 are mediastinal nodes. Taken from [157]

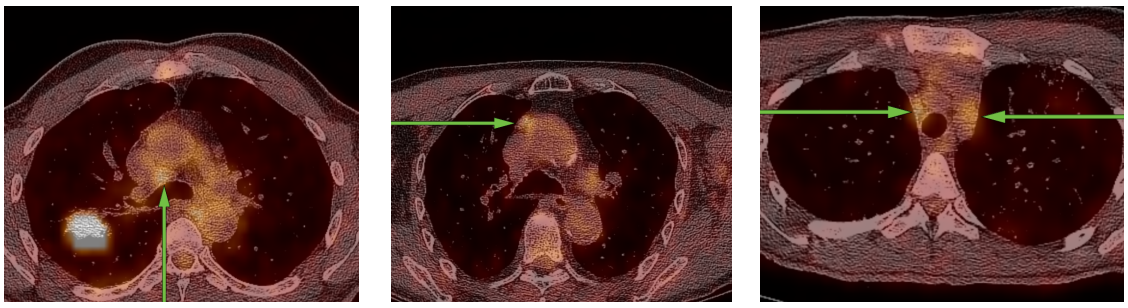


Fig. 5.3 Examples of mediastinal nodes on FDG-PET/CT scans with the nodes indicated by the arrows. The nodes appear as high uptake on the PET (in orange) at a lymph node location on the CT (in greyscale). Images captured using *LIFEx* [162]

[119] trained a CNN to predict overall two-year survival for NSCLC patients undergoing radiotherapy and surgery (AUC 0.70 and 0.71 respectively). Deep learning-based methods have also been employed to solve PET/CT problems similar to ours. Using whole-body PET/CT scans, [166] used a 3D CNN to detect diseased lymph nodes in patients with a sensitivity of 0.85 and four FPs/patient.

However, there has been less progress towards the automated detection of pathological mediastinal lymph nodes. [167] used anatomical characteristics such as the nodes' shape and intensity range to identify lymph nodes from solely CT scans, achieving a sensitivity of 0.82 but a PPV of only 0.13. Similarly, [168] used a blob-like structure enhancement filter followed by an SVM to detect nodes with a sensitivity of 0.84 with nine FPs/volume. The aim of these studies was to automatically detect any lymph nodes, rather than specifically pathological nodes. In [169] they used a CNN to automatically detect enlarged lymph nodes (indicating disease), again from only CT scans. The input data consisted of 2.5D patches (three orthogonal slices) centred on the lymph node locations. With a cohort of 86 patients they achieved a sensitivity of 0.70 with three FPs/patient.

Several PET thresholding methods have also been proposed, as described in the meta-analysis in [170]. These studies used features such as PET SUV_{max} and PET SUV_{mean} to classify nodes as pathological. Using an *Activity > Background PET* threshold, average sensitivity and specificity across 18 studies were 0.77 and 0.90 respectively. Using an $SUV_{max} \geq 2.5$ criterion gave an average sensitivity and specificity of 0.81 and 0.79 respectively. However, the meta-analyses for both methods showed high inter-study heterogeneity. All these studies required SUV-based characteristics of the nodes and thus required the nodes to first be located manually by an expert. [171] assessed the inter-physician variation when identifying suspicious mediastinal nodes on PET scans. Fourteen physicians with varying levels of experience were asked to identify mediastinal nodes on 27 PET scans. Using a consensus judgement as a gold standard, sensitivity scores were between 0.58 and 0.85. Sensitivity scores were similar for less experienced and more experienced physicians.

Given the diversity of mediastinal node shapes, sizes, and locations, using solely the CT or PET is limiting and leads to large variations in opinion even between experienced physicians. [172] showed that using PET/CT for nodal staging of NSCLC confers significantly higher sensitivity and specificity than only contrast-enhanced CT and higher sensitivity than only PET. Using both PET and CT information, [173] correctly categorised scans as (N0 or N1) vs (N2 or N3) with an accuracy of 0.99, compared to an accuracy of 0.72 by an expert

reader[1]. They used diagnostic features of the primary tumours and lymph nodes (nodal uptake/background, node size, primary size, and primary SUV_{max}) to train a neural network and evaluate on a cohort of 133 patients. Finally, with a cohort of 168 patients, [175] used a 2.5D CNN (using six axial slices of size 51x51 mm^2) to diagnose mediastinal lymph node metastasis, achieving a sensitivity and specificity of 0.84 and 0.88 respectively (AUC 0.91). They compared this to diagnosis using diagnostic features (e.g. node size, SUV_{max}) and found no significant difference in performance (AUC 0.87-0.92 using diagnostic features).

Furthering this work, our aim was to design a system to automatically classify mediastinal nodes directly from PET/CT scans, without the need for prior annotation. This system could then be used by a physician to quickly identify high-risk lymph nodes, saving time and increasing diagnosis consistency. To achieve this clinical goal we built a fully automated 3D end-to-end algorithm. We compare our results to those of experienced physicians and show comparable classification accuracy.

5.3 Dataset

Details of the dataset are shown in Table 5.1. A cohort of 134 NSCLC patients who underwent FDG-PET/CT scans (Gemini TF; Philips Medical Systems, Best, the Netherlands) were studied. This study was approved by an institutional review board. PET/CT was performed 60 minutes after intravenous injection of 3 MBq/kg of FDG, with 105 second acquisition per bed position. CT images were obtained without a contrast medium. Images were reconstructed using a blob ordered subset–time of flight list-mode iterative algorithm (2 iterations, 33 subsets, including attenuation and scatter corrections). PET voxels were 4x4x4 mm^3. CT in-plane resolution varied from 0.58 mm to 1.05 mm and the inter-plane resolution was 1 mm. All scans were analysed by an experienced dual-board certified radiologist (15 years' experience in thoracic imaging) who marked the positions of pathological mediastinal nodes. Nine patients were excluded from the study because of multimetastatic disease involving lung, pleura, and the mediastinum leading to unreliable nodal identification. This gave a total of 124 patients comprising 172 positive nodes. Twenty-nine patients (comprising 52 nodes) were reserved for testing and only used for the final evaluation of the model. For hyperparameter optimisation a subset of the training set (a validation set of 24 patients and

[1]From the staging system for classification of cancer spread, where the N refers to the extent of nearby lymph node involvement. N0 indicates no regional lymph node metastasis, N1 metastasis in ipsilateral peribronchial and/or ipsilateral hilar lymph nodes and intrapulmonary nodes, including involvement by direct extension, N2 indicates metastasis in ipsilateral mediastinal and/or subcarinal lymph node(s), and N3 indicates metastasis in contralateral mediastinal, contralateral hilar, ipsilateral or contralateral scalene, or supraclavicular lymph node(s) [174]

Table 5.1 Summary of the mediastinal dataset, showing the numbers of patients and positive nodes in each cohort

Cohort	Patients	Positive Nodes
Total Patients	134	-
Excluded Patients	9	-
Training Set	72	85
Validation Set	24	35
Test Set	29	52

35 nodes) was used. During final training the validation set was included in the training set. For some early experiments we only had 58 patients available. These cases are noted.

5.4 Preprocessing - Method

There was a significant amount of preprocessing to be done before the scans could be used in a deep learning network.

5.4.1 Image Resolution

The resolution of CT scans is higher than that of PET, so we first needed to ensure that the coordinates of both modalities matched up. To do this we used a cubic spline interpolation to upsample the PET resolution to that of the CT (spline interpolation is a reliable and computationally efficient method for interpreting medical images [176]). Upsampling to the CT resolution was used so that we did not lose any information from the CT in this initial preprocessing step. In further sections we test different image resolutions to find the optimum for the task.

5.4.2 Finding the Mediastinal Region

The raw images covered varying portions of the body; some were whole body scans, whereas others contained only the chest region. We needed a method to cut the scans down to just the mediastinal regions, so we did not have huge areas not relevant to the task. This proved difficult because of differences in the shapes of the lungs and the different scan sizes. Several options were considered:

5.4.2.1 Using a Lung Atlas

One option considered was to use a CT lung *atlas*. Each scan would be deformed to match this atlas, and thus the mediastinal region would be in the same place for each scan. This is common in brain MRI studies [177] [178] [179]. However, unlike for brain scans, the physiology of the chest region can vary hugely. There are studies demonstrating chest CT registration, but these are mainly for motion-correction, meaning the registration is for the same patient [180] [181]. Segmentation software has been developed to segment the mediastinum, and would be ideal for this task, however this is not publicly available [182]. To create a similar program would require a dataset of segmented chest CT scans, which we did not have access to. We therefore discounted this method.

5.4.2.2 Using the CT profile

Another option was to use the profile of the sum of the CT values slice by slice. Because the lungs contain air they have low HU values. Thus, looking at the means of the CT values along the different axes, one would expect to see a dip corresponding to the lungs. We plotted this profile along the three orthogonal axes for several patients to see if this would allow us to find the lungs by an automated minimum detection.

5.4.2.3 Automatically Segmenting the Lung

Using the same feature of the lungs, their low HU values, we tried automatically segmenting them. One would expect the lungs to be the two biggest air-filled regions within the body, so we tried to find these regions through automated thresholding. This method is based on [183], but we had to tweak some of the parameters to make it work in our case.

The CT scans were first thresholded so that any voxels with HUs greater than -400 were set to zero and any below were set to one. This identified the air-filled regions. This was followed by a two-iteration binary closing operation and a seven-iteration binary opening operation (both with a square connectivity of one). These operations remove very small air-filled regions and small nodules within the lungs. In some cases the lung regions connected to the region of air outside the patient through the trachea, meaning the segmentation did not work. Tuning of the binary operations was necessary to prevent this from happening. Often the largest regions (or sometimes largest 2-3 regions, depending on bed position) were the air regions outside the patient, so we first discounted any regions within 25 voxels of the image border.

In most cases this worked, but in some cases the lungs were connected (so finding the largest two regions also found a non-lung region). In other cases a particularly large tumour

meant that, despite the binary opening and closing operations, a lung would be split into two regions, and the second-largest region would be in the intestine. To avoid these problems we instead just used the single largest region, which in all cases was one of the lungs.

To determine the mediastinal region we took only the slices within the limits of this segmentation, giving an image for each patient containing only the lung area.

5.5 Preprocessing - Results and Discussion

5.5.1 Using the CT profile

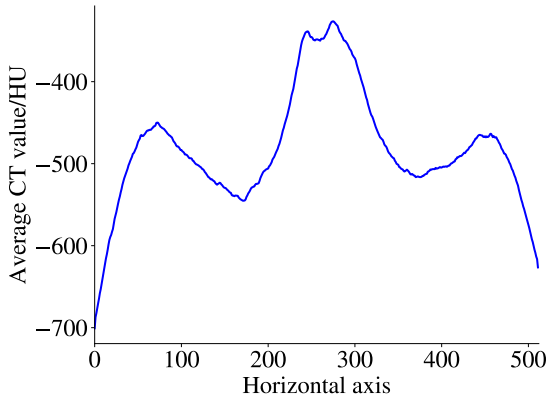
Figure 5.4 shows CT profiles in the horizontal and craniocaudal directions for cases that worked well. In these cases we see clear minima at the lung locations and could hence use these minima to isolate the mediastinal region. However, as shown in Figure 5.5, there were cases where this did not work. In these examples there are no clear minima and thus there is no clear way of identifying the mediastinal regions.

5.5.2 Segmenting the Lung

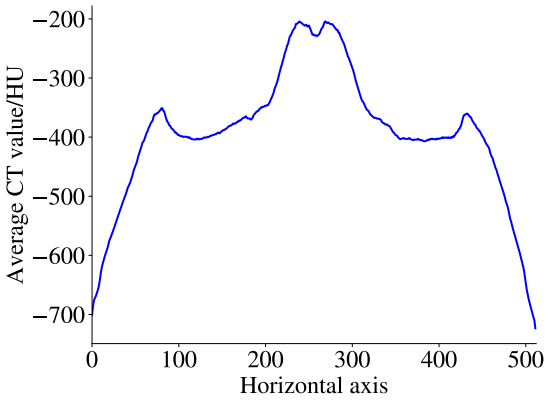
Figure 5.6 shows some of the problems encountered when segmenting the lungs. In Figure 5.6a the lungs are joined, meaning when we tried to segment the largest two regions we found a non-lung region. This figure also shows several large regions around the edge of the image. Figure 5.6b shows a patient with a large tumour in the right lung, which has resulted in it being split into several smaller regions. The large region in the abdomen was then chosen as the second-largest volume. These problems were resolved by taking only the largest region for each patient and by discarding regions at the edges of the image.

5.5.3 Discussion

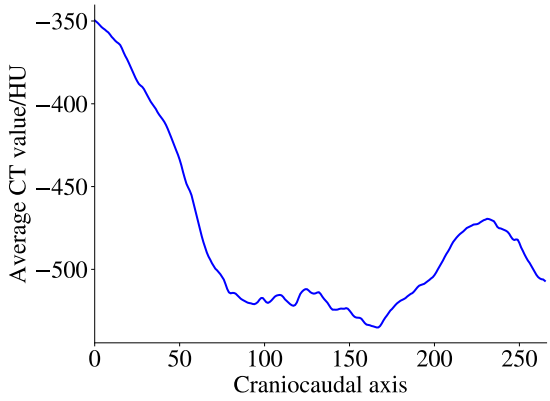
The CT profile did not work reliably enough as a method for selecting the lung region. We therefore used the lung segmentation method. There could be cases where this method does not work (for example a patient with large tumours in both lungs), but for our dataset it worked for all patients. Although this is crude (sometimes segmenting one and sometimes two lungs), it serves its purpose. If we had reliably segmented both lungs we may have been able to cut the volume of interest down further by excluding any areas that were not between the two lungs (and thus keeping only the actual mediastinal region). A future improvement could be to use a CNN to achieve this, but this was not attempted due to the added time and complication (it would require manual segmentations of the lungs, which we did not have).



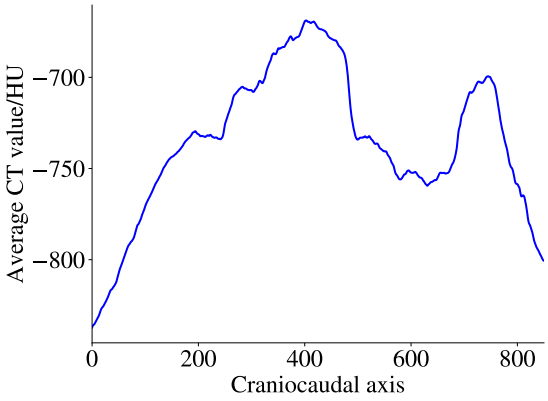
(a) Example in the horizontal direction



(b) Example in the horizontal direction

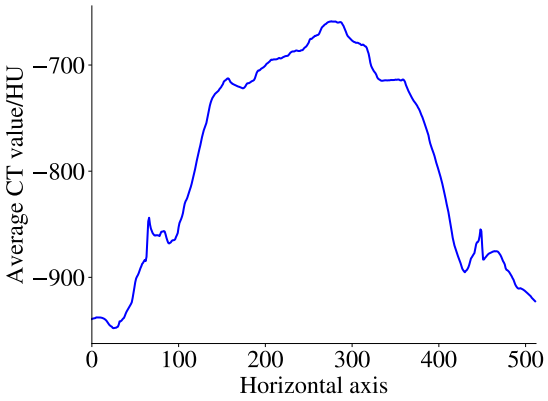


(c) Example in the craniocaudal direction

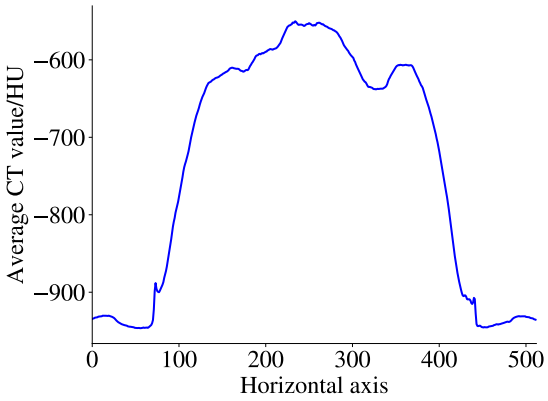


(d) Example in the craniocaudal direction

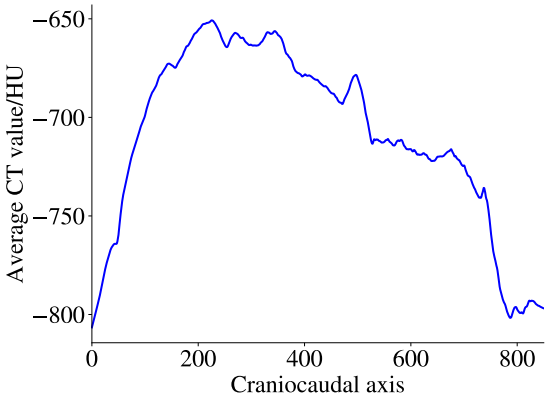
Fig. 5.4 Mean HU values along the horizontal and craniocaudal directions for example CT scans. These cases show distinct dips in the mean CT values along these axes and could thus be used to locate the lungs



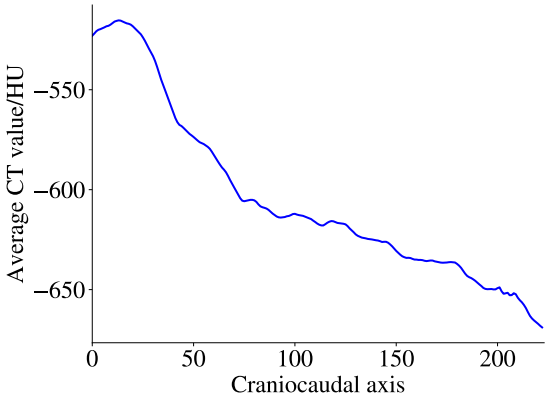
(a) Example in the horizontal direction



(b) Example in the horizontal direction

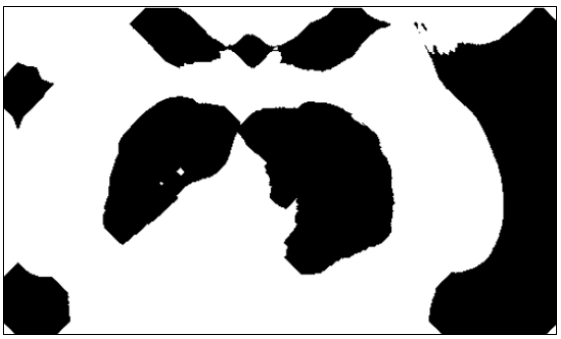


(c) Example in the craniocaudal direction



(d) Example in the craniocaudal direction

Fig. 5.5 Mean HU values along the horizontal and craniocaudal directions for example CT scans. In these cases there are no clear dips in the profiles, meaning it would be difficult to precisely locate the lungs



(a) Two problems encountered when segmenting the lungs. The two lungs are joined and there are several regions of air outside the body

(b) A patient with a large tumour in the right lung, meaning the bladder is chosen as the second-biggest air-filled region

Fig. 5.6 Problems encountered when segmenting the lungs

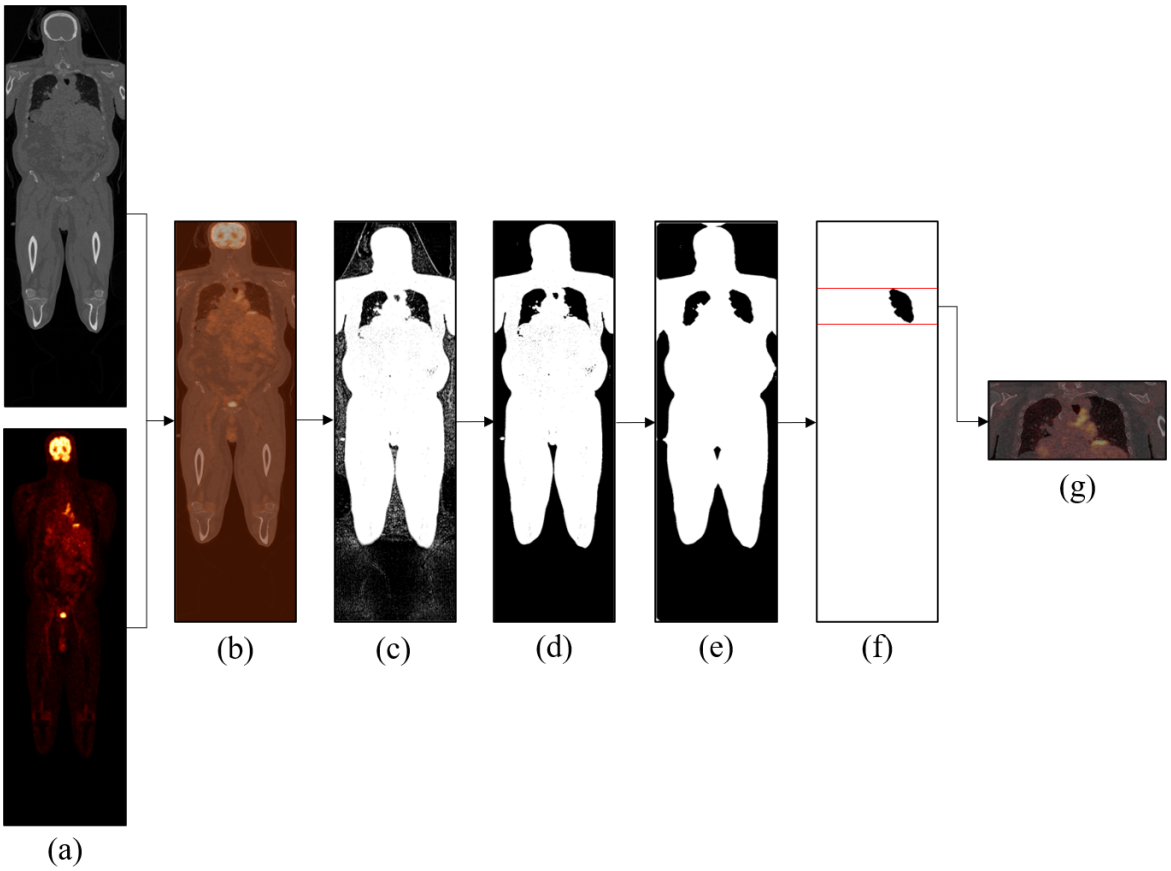


Fig. 5.7 Summary of mediastinal preprocessing steps. The input images of different resolutions (a) are aligned to the CT resolution and merged (b). The CT is thresholded to identify air-filled regions (c) followed by binary closing and opening operations (d) and (e) for the removal of small regions. The largest region, a lung, is selected (f), and the limits of this region are used to cut the relevant slices from the PET/CT image

Figure 5.7 summarises the full set of preprocessing steps taken.

5.6 One-Phase Approaches - Method

Having the whole chest as an input, with the mediastinal nodes making up only a small fraction of the volume, posed a problem. Only a very small percentage of each scan was relevant to the question we were asking. We first tried methods using just one phase of classification, i.e. going directly from the scans to a classification. We tried two methods; using PET maximum intensity projections (MIPs) and using 3D PET/CT cubes. These are illustrated in Figure 5.8.

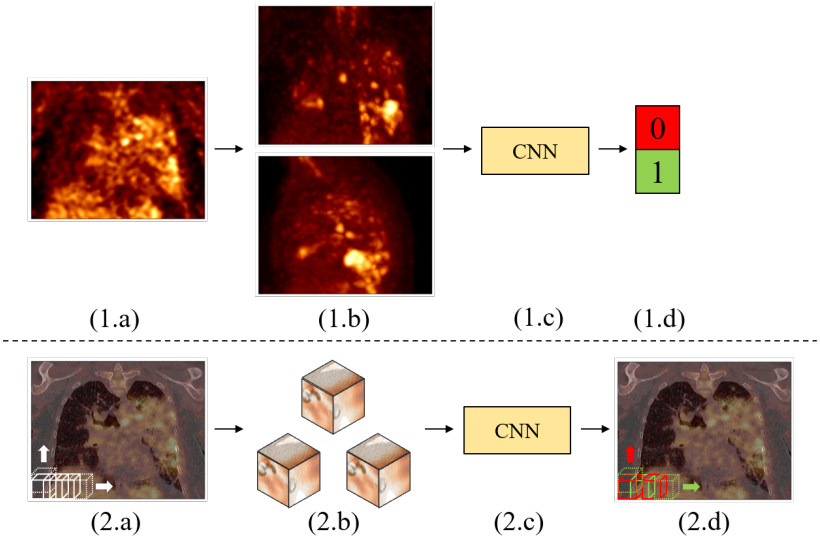


Fig. 5.8 Summary of one-phase approaches. (1) shows the PET MIP approach. The PET image (1.a) is converted into MIP images (1.b). These are input into a CNN (1.c), which outputs a classification for each input image (1.d). (2) shows the 3D cube approach. The PET/CT image is split into overlapping 3D cubes (2.b). These are input into a CNN (2.c), with the output giving the locations of predicted nodes on the scan (2.d)

5.6.1 PET Maximum Intensity Projection Approach

Introduction MIP images are projections from a 3D to a 2D space, with each pixel having the value of the voxel that has the maximum intensity along its projection axis:

$$M_{ij} = \max_k V_{ijk}$$

where M is the 2D MIP projection, V is the full 3D image, and k is the axis of projection.

MIP images have been used for MRI breast lesion classification [184], L3 CT slice detection[2] [185], and for whole-body 18F-FDG PET/CT lymphoma uptake patterns [186]. This final case in particular shows similarities to our problem, as it also involves FDG-PET/CT volumes which are large compared to the regions of interest. Mediastinal nodes have high 18F-FDG PET uptake so show as bright spots on MIP images. In a clinical setting MIP images are often used by physicians to help locate nodes [187], as looking at the MIP image is much quicker than viewing the whole 3D image. Because of this simplicity this was the first method attempted. Using 2D images means CNN learning is much faster and gives the option of transfer learning from networks trained on large 2D image databases, such as ImageNet.

[2]L3 CT slices are slices extracted at the third lumbar vertebra

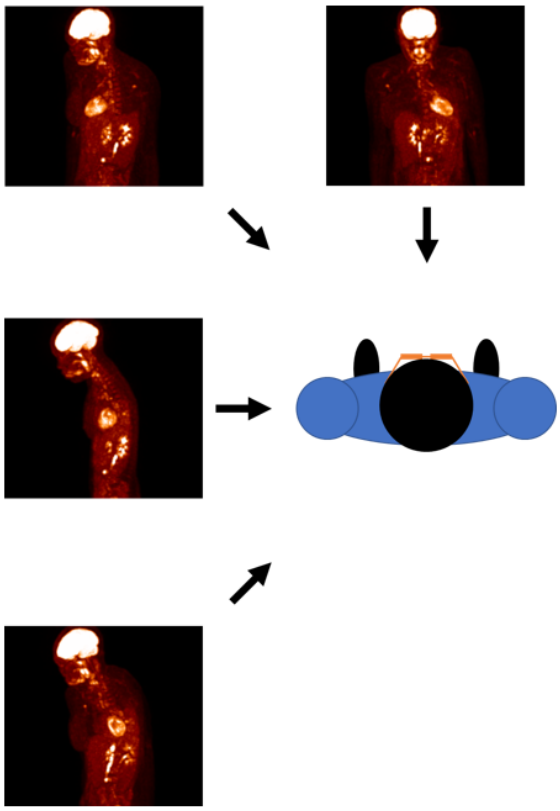


Fig. 5.9 Orientations of the four MIP directions used, showing the four resulting MIP images

Table 5.2 Parameters tested - PET MIP approach

Parameter	Values Tested
Model	MIP_1 - MIP_3
Learning Rate	10^{-3} 10^{-4}
Transfer Learning	Yes No
Number of MIP Orientations	2 (at 90°) 4 (at 45°)
Freeze TL Layers	Yes No

Note that, these experiments having taken place early in the thesis, we only had 58 patients in total. These were split into: 24 training, 6 validation, and 28 test.

Method Because this method did not use the CT, during preprocessing the PET voxels were not upsampled to match the CT voxels. The voxel resolution was therefore 4x4x4 mm^3. The 3D PET images were all cut to a size of 100x100x80 voxels by symmetrically trimming or padding the edges. This was done to reduce the large redundant areas around the body and to make the data of uniform resolution so they could be used in the CNN. Voxel values were rescaled to fall between zero and one. MIP images from four angles were created at 45° intervals in the vertical plane (see Figure 5.9). Experiments were run using several model architectures, some using all four MIP images as input, others just two orthogonal MIPs (the front and side views). Given the small dataset size, in most experiments we employed transfer learning, using an Inceptionv3 model pretrained on ImageNet. Inceptionv3 only accepts 3-channel images (as it is designed for RGB images), so each MIP image was duplicated three times and input into a separate Inceptionv3 branch of the network. The outputs of these branches were merged, then fully connected layers were added before a final classification. An example network is shown in Figure 5.10 (corresponding to the model MIP_3) and network architectures for all experiments are described in Appendix C.1. Tests were performed training the models with the Inceptionv3 weights frozen (so only the weights in the fully connected layers at the end could change) and unfrozen (so all weights could change). Each experiment was run for 50 epochs using an SGD optimiser, cross-entropy loss, a batch size of four, and weighted classes to account for the class imbalance. We tested learning rates (lrs) of 10^{-3} and 10^{-4}. Random rotations $\in [-10°, 10°]$ and translations $\in [-20$ voxels$, 20$ voxels$]$ in all three axes were used to augment the dataset by a factor of 12. The details of the parameters tested are shown in Table 5.2.

Because the inputs for this approach were entire images it did not give locations of positive nodes, only an overall classification for each image. In the future, steps could be added to localise nodes on positive scans.

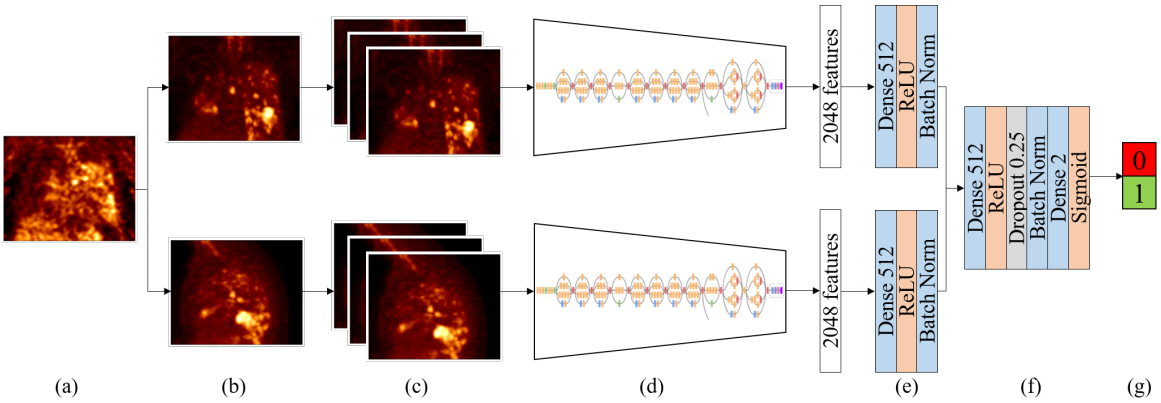


Fig. 5.10 MIP CNN architecture for MIP_3. The original PET image (a) is transformed into MIP projections (b). These are duplicated three times (for the three Inceptionv3 channels) (c) and input into the Inceptionv3 networks pretrained on ImageNet (d). The 2048 features from the final layers of each of the Inceptionv3 networks are extracted, followed by dense, ReLU, and batch normalisation layers (e). The outputs of these are then merged and succeeded by more dense, dropout, batch normalisation, and activation layers (f). Finally, the image is categorised as positive or negative (g). Note that in this diagram only two MIP orientations are shown. For networks using four MIP orientations, the same architecture was used but with four branches

5.6.2 3D Cube Approach

Introduction Secondly, we tried a fully 3D approach, utilising both the CT and the PET. The scans were far too large to be used in a CNN whole, so instead we broke them down into smaller cubes. Marking individual cubes containing a node as positive and those without as negative, by predicting the status of a cube we could determine the location of the predicted node in the scan. This is similar to methods used in the analysis of histopathological slices, which are typically of the order of several million pixels in size. The slices are therefore cut into smaller patches using a textitsliding window approach [188] [189]. Note that for these experiments we had the full dataset available.

Self-Supervised Learning Because our dataset was small, we again looked to transfer learning methods. In contrast to 2D, for 3D data there are not any large well-established pretrained models to use. We therefore pretrained models using a self-supervised method (a jigsaw-based task), as described in 4.3.3. Scans were first resampled to have isotropic voxels of size 1x1x1 mm^3. For each scan a number of cube pairs were randomly selected, at random orientations with respect to each other (i.e. above/below/left/right/front/behind). A random translation of $\leq |5|$ voxels was applied to each cube position to increase the difficulty of the task. We then trained CNNs to predict the cube pairs' orientations. We tried several network architectures, using as our starting points CNNs that had been used in

Table 5.3 Parameters tested - self-supervised learning

Parameter	**Values Tested**
Model	Jigsaw_1 - Jigsaw_{19}
Downsampling Factor	1 2 4
N_{jigsaw}	50 100
Modality	CT PET
Learning Rate	0.01 0.001
Batch Size	16 32

similar multimodal classification tasks (e.g. [190] used multi-modal brain MRI data to predict survival time, training a CNN with several branches corresponding to different modalities. [191] used a two-branch CNN to predict locoregional recurrence in head and neck squamous cell carcinoma using PET and CT). In these networks different branches were used for the different input modalities. We therefore tried similar network designs. We planned to pretrain a CNN for the PET and CT separately. One branch from each could then be cut and merged to form a network appropriate for our classification task. Figure 5.11 shows this process. For each patient, N_{jigsaw} cube pairs were created. The networks were trained on cubes from the training cohort and validated on cubes from the validation cohort. We used input cubes of side length 30 mm. (This was a compromise between too-small and too-large fields of view. Most mediastinal lymph nodes are of diameter 0-20 mm [192]). In all cases the input values were rescaled to between zero and one. For some experiments we downsampled the scans prior to training, using a cubic spline interpolation. This was to give a larger field of view, so there was more physiological information for the jigsaw network.

The parameters tested are listed in Table 5.3. Networks were trained for 50 epochs or stopped early if it was clear that the validation loss was not decreasing. In all cases an Adam optimiser and cross-entropy loss were used. The CNN architecture for Jigsaw_9 is shown in Figure 5.11. All other network architectures are detailed in Appendix C.2.

Method With a suitable model pretrained, the scans had to be prepared for the actual task of detecting mediastinal nodes. The scans were of different resolutions, so were first resampled to have isotropic voxels of size 1x1x1 mm^3 using a cubic spline interpolation. The voxel values for both PET and CT were rescaled so they fell within the range [0,1] (to match the preprocessing used during network pretraining). We then divided the scans into cubes so they could be used in the CNN. For training, cutting a cube at every voxel coordinate would result in a very large, highly unbalanced dataset. We therefore undersampled the negative cubes. At every voxel within 5 mm of a positive node a cube of side length c mm was cut centred on

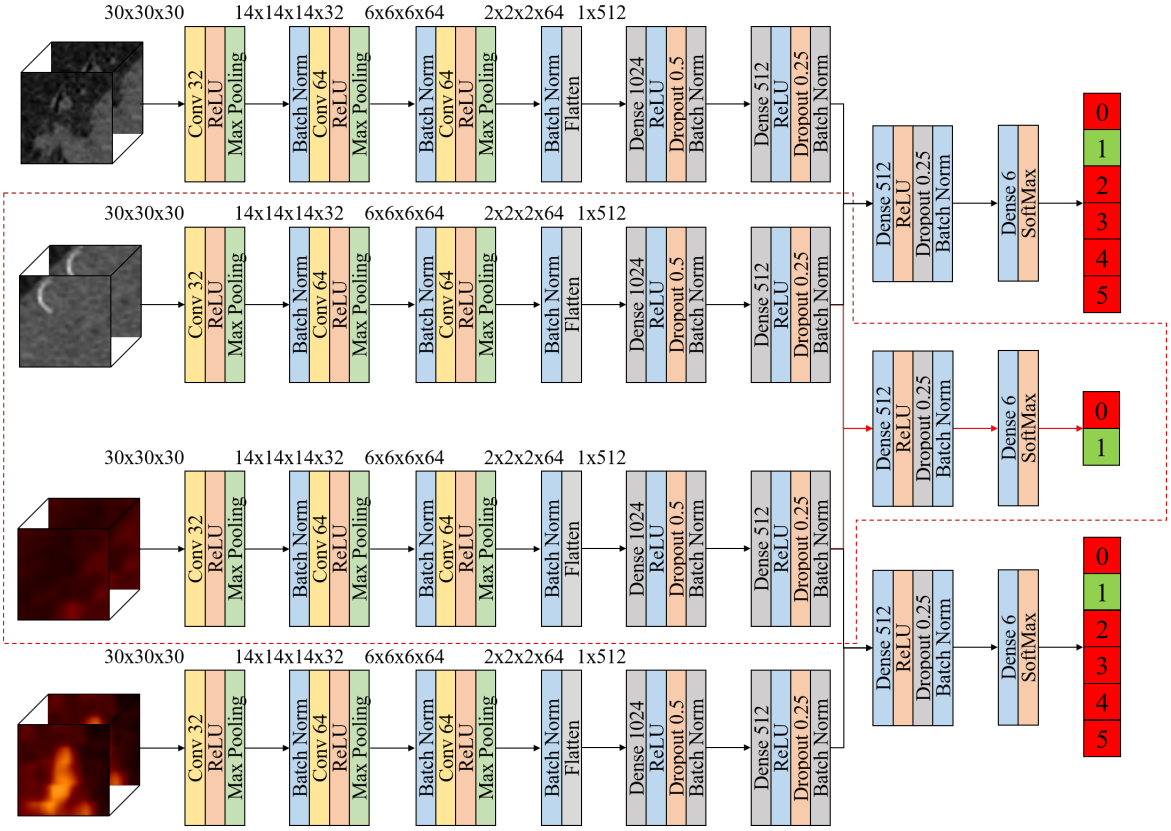


Fig. 5.11 Transfer learning for 3D cube approach. Two networks are pretrained using the jigsaw task, one with CT inputs (the upper half of the diagram) and one with PET inputs (the lower half of the diagram). Each of these has six output classes, corresponding to the six jigsaw orientations. Once trained, one branch from each of these networks is merged to give a pretrained network that can be used for our classification task (in the red box). Dense, ReLU, batch normalisation, and dropout layers are added, before a two-class output (corresponding to the cube being classified as positive or negative)

Table 5.4 Parameters tested - 3D cube approach

Parameter	Values Tested
Pretrained Model	None test 22 test 33
Model	$3D_1$ - $3D_3$
Modality	CT PET BOTH
c/mm	30 50 70
Augmentation	Flips Rotations
N_{neg}	150 250 600
Batch Size	16 32

the voxel. These were labelled as positive. N_{neg} negative cubes per patient were created by choosing random points within the images that were not within 10 voxels of positive nodes. This distance of 10 voxels was set so that we did not have confusing border cases. These cubes were used to train CNNs. CNNs with and without pretraining on the jigsaw task were tested. All CNN architectures are illustrated in Appendix C.3. Flips and rotations of 90° in all three axes were tested as augmentation. Cross-entropy loss and an Adam optimiser were used in all cases and the loss was weighted proportionally to the class imbalance. A learning rate of 10^{-3} was used. All parameters tested are shown in Table 5.4. Experiments were run for 20 epochs (by which point the validation loss had stopped decreasing).

Once the best combination of parameters was chosen, the model was tested on the full validation set images. In training we only took a sample of N_{neg} negative cubes, to reduce the class imbalance. To fully classify the validation images we input overlapping cubes sweeping across the whole image volumes, with spacings of 2 voxels. The classification of the cubes (using CNN output thresholds of 0.70, 0.90, 0.99) was used to plot predicted node locations.

We tested cubes of side length 30 mm, 50 mm, and 70 mm. In cases where the cube side length was not 30 mm (and thus the pretrained network input was not the same size), the cubes were resampled using a cubic spline interpolation to have side length 30 mm. This was deemed to be easier than retraining and reoptimising the self-supervised model (although it does mean that for larger sizes some information was lost).

5.7 One-Phase Approaches - Results and Discussion

5.7.1 PET MIP Approach

The results are shown in Table 5.5. Tests 1 and 2 were experiments without transfer learning (using two and four MIP orientations respectively), tests 3-5 used Inceptionv3 networks with

Table 5.5 Results - PET MIP approach

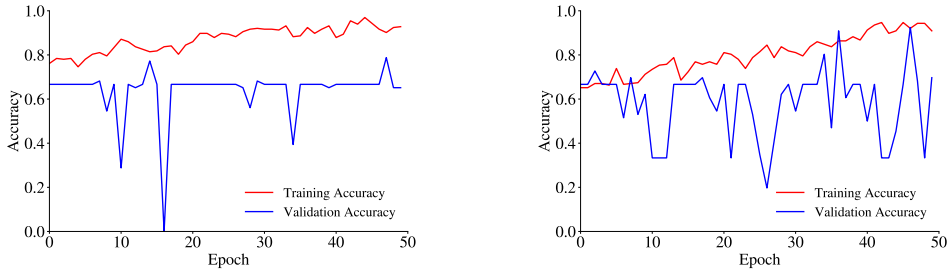
Test ID	Model	lr	TL	No. MIP Orientations	Frozen	Val Acc
1	MIP_1	10^{-3}	No	2	No	0.67
2	MIP_1	10^{-3}	No	4	No	0.67
3	MIP_2	10^{-3}	Yes	2	Yes	0.67
4	MIP_2	10^{-3}	Yes	4	Yes	0.67
5	MIP_3	10^{-3}	Yes	2	Yes	0.67
6	MIP_2	10^{-3}	Yes	2	No	0.67
7	MIP_2	10^{-3}	Yes	4	No	0.67
8	MIP_1	10^{-4}	No	2	No	0.67
9	MIP_1	10^{-4}	No	4	No	0.67
10	MIP_2	10^{-4}	Yes	2	Yes	0.67
11	MIP_2	10^{-4}	Yes	4	Yes	0.67
12	MIP_3	10^{-4}	Yes	2	Yes	0.67
13	MIP_2	10^{-4}	Yes	2	No	0.67
14	MIP_2	10^{-4}	Yes	4	No	0.67

pretrained weights frozen, and tests 6 and 7 unfroze the Inceptionv3 weights. Tests 8-14 were the same, but with a learning rate of 10^{-4} instead of 10^{-3}. None of the experiments were able to find the positive patients, instead classifying all images as negative ($2/3$ of cases in the validation set were negative). Figure 5.12 shows some example learning curves from the experiments, showing no improvement in the validation accuracy over time. While this set of experiments was in no way comprehensive (we did not try changing the batch size and only tried a few network architectures), it quickly became clear that the models were not producing anything that could give reliable conclusions. This is partly because of the small dataset; with only six patients in the validation set it was very hard to see small improvements across experiments. Not using the CT inevitably limited the effectiveness of the method, as the CT is key to physically orientating a scan. We thought that the PET MIP approach may at least be useful as a precursor step but found this to not be the case.

Because none of the validation tests were successful, we did not use the test set.

5.7.2 Self-Supervised Learning

Results are shown in Table 5.6. Note that there are six classes (corresponding to the six orientations in the jigsaw task), so the *no learning* accuracy would be 0.17. Tests 1-6 tested different learning rates and batch sizes for CT cubes with no downsampling. We found that without downsampling we were unable to train the CNN. In tests 7-20 we changed the



(a) Learning curve for PET MIP test 2 (b) Learning curve for PET MIP test 3

Fig. 5.12 Learning curves for two of the PET MIP experiments. They show the network is not learning the node locations at all, just overfitting the training set. This is typical of what we saw using this method

downsampling factor to two, testing various models with different batch sizes and learning rates. A comparison of tests 14-16 (all using the Jigsaw_4 model) showed that a learning rate of 0.01 and batch size of 32 gave the best results, so these were used for subsequent experiments. Downsampling the scans by a factor of four led to the validation accuracy increasing from 0.46 in test 17 to 0.62 in test 21. Tests 22-32 used different CNN architectures with a bigger training dataset (increasing N_{jigsaw} from 50 to 100). The highest validation accuracies attained were tests 22 and 25 (both 0.84), but test 23 and tests 27-32 all scored between 0.76 and 0.79 and could therefore be viable networks for transfer learning. Figure 5.13 shows the learning curve for test 22, showing how the training and validation accuracy improve over training epochs. Although there is some overfitting, the network has a final validation accuracy of 0.84. Finally for the CT, in test 33 we tested using the best-performing model (Jigsaw_9) without any downsampling, but this gave a low validation accuracy of 0.25.

We then tried to pretrain a network using the PET cubes. Tests 34-37 attempted to train the PET-based self-supervised CNN, however in all cases the performance was poor, even with a downsampling factor of four. We did not run any further tests on the PET cubes.

To explain the effect of downsampling, and the failure of the PET-based CNN, in Figure 5.14 we show examples of neighbouring cubes at the three downsampling factors of one, two, and four, for CT and for PET. Looking at the CT cubes, at a larger scale we have more physiological context, meaning it is easier to orientate neighbouring cubes. This explains the superior performance of the resulting self-supervised models. The concern when using a network pretrained on downsampled cubes is that the input resolutions for the jigsaw task and the actual task of finding mediastinal nodes will not be the same. The aim of pretraining the model is to improve performance on the mediastinal classification task, not to optimise for the jigsaw task itself. It is unclear how using a different resolution will affect the transfer

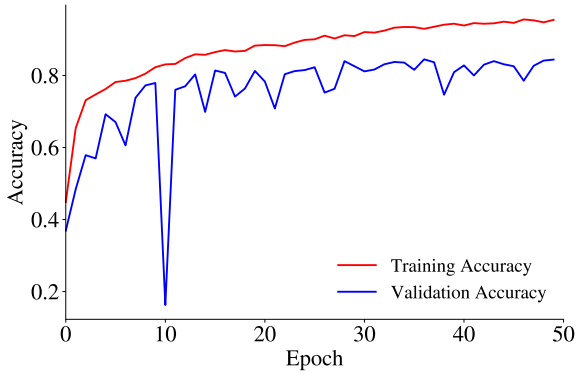
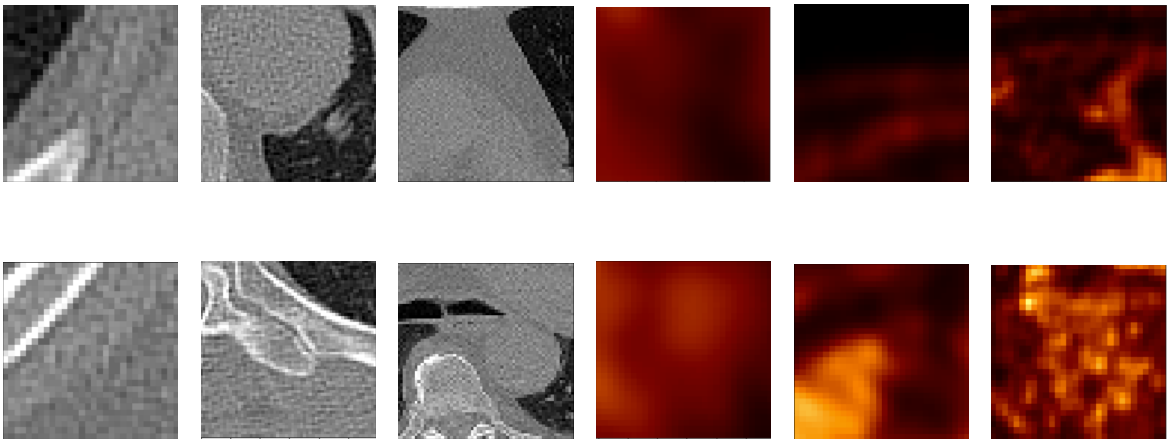


Fig. 5.13 Learning Curve for self-supervised test 22



(a) CT: SF 1 (b) CT: SF 2 (c) CT: SF 4 (d) PET: SF 1 (e) PET: SF 2 (f) PET: SF 4

Fig. 5.14 Examples of paired cubes at different downsampling scale factors (SFs) used for the self-supervised jigsaw task. (a) to (c) are paired CT cubes at scale factors of 1, 2, and 4 respectively, (d) to (e) are equivalent examples for PET cubes

of weights from one task to another. Because of this, in the next section we will test CNNs pretrained with different downsampling factors.

Examining the PET cubes in Figure 5.14, even with a downsampling factor of four it is very hard to orientate the cubes. This explains the failure of the PET-based tests. Because of this, we will use solely the CT-based pretrained CNNs for transfer learning. Rather than merging one branch from a PET-based model and one branch from a CT-based model (as in Figure 5.11), we will simply use the whole CT-based models. Although CT-based features are clearly not very similar to PET-based ones, the learnt weights may be better than if they were randomly initialised.

Table 5.6 Results - self-supervised learning. Results marked '-' were manually stopped when it was clear the CNN was not learning. The best tests are highlighted

Test ID	Model	lr	Batch Size	Downsampling	N_{jigsaw}	Modality	Val Acc
1	$Jigsaw_1$	10^{-2}	32	1	50	CT	-
2	$Jigsaw_1$	10^{-3}	32	1	50	CT	-
3	$Jigsaw_1$	10^{-2}	16	1	50	CT	-
4	$Jigsaw_2$	10^{-2}	32	1	50	CT	-
5	$Jigsaw_2$	10^{-3}	32	1	50	CT	-
6	$Jigsaw_2$	10^{-2}	16	1	50	CT	-
7	$Jigsaw_1$	10^{-2}	32	2	50	CT	0.20
8	$Jigsaw_2$	10^{-2}	32	2	50	CT	0.19
9	$Jigsaw_2$	10^{-3}	32	2	50	CT	-
10	$Jigsaw_2$	10^{-2}	16	2	50	CT	-
11	$Jigsaw_3$	10^{-2}	32	2	50	CT	-
12	$Jigsaw_3$	10^{-3}	32	2	50	CT	-
13	$Jigsaw_3$	10^{-2}	16	2	50	CT	-
14	$Jigsaw_4$	10^{-2}	32	2	50	CT	0.45
15	$Jigsaw_4$	10^{-3}	32	2	50	CT	-
16	$Jigsaw_4$	10^{-2}	16	2	50	CT	-
17	$Jigsaw_5$	10^{-2}	32	2	50	CT	0.46
18	$Jigsaw_6$	10^{-2}	32	2	50	CT	0.31
19	$Jigsaw_7$	10^{-2}	32	2	50	CT	0.27
20	$Jigsaw_8$	10^{-2}	32	2	50	CT	0.43
21	$Jigsaw_5$	10^{-2}	32	4	50	CT	0.62
22	$Jigsaw_9$	10^{-2}	32	4	100	CT	0.84
23	$Jigsaw_{10}$	10^{-2}	32	4	100	CT	0.81
24	$Jigsaw_{11}$	10^{-2}	32	4	100	CT	0.25
25	$Jigsaw_{12}$	10^{-2}	32	4	100	CT	0.84
26	$Jigsaw_{13}$	10^{-2}	32	4	100	CT	-
27	$Jigsaw_{14}$	10^{-2}	32	4	100	CT	0.76
28	$Jigsaw_{15}$	10^{-2}	32	4	100	CT	0.77
29	$Jigsaw_{16}$	10^{-2}	32	4	100	CT	0.79
30	$Jigsaw_{17}$	10^{-2}	32	4	100	CT	0.78
31	$Jigsaw_{18}$	10^{-2}	32	4	100	CT	0.76
32	$Jigsaw_{19}$	10^{-2}	32	4	100	CT	0.77
33	$Jigsaw_9$	10^{-2}	32	1	100	CT	0.25
34	$Jigsaw_9$	10^{-2}	32	4	100	PET	0.17
35	$Jigsaw_9$	10^{-3}	32	4	100	PET	-
36	$Jigsaw_9$	10^{-2}	16	4	100	PET	-
37	$Jigsaw_9$	10^{-3}	16	4	100	PET	-

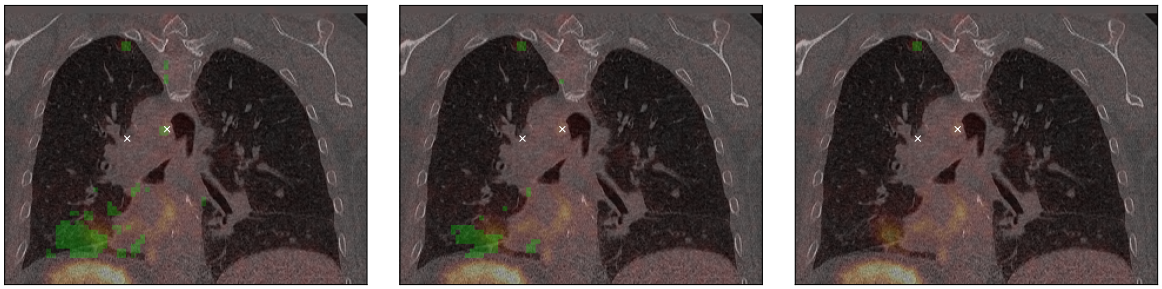
5.7.3 3D Cube Approach

Results from the different parameter tests are shown in Table 5.7. Tests 1-3 did not use transfer learning. Test 1 used both PET and CT (AUC 0.94), whereas tests 2 and 3 used only CT (AUC 0.68) and only PET (AUC 0.95) respectively. The CT performed much worse than the PET in this case, and there was no significant advantage to combining the PET with the CT. Tests 4-7 used transfer learning with weights transferred from test 22 above. These tested the number of negatives cubes N_{neg} and the augmentations types. The best performances were tests 5 and 6, both achieving AUCs of 0.96. They used 150 and 250 negative cubes per patient respectively with augmentation. Tests 8 and 9 tested input cubes of side length 50 mm and 70 mm respectively. These did not perform any better than the cubes of side length 30 mm, with AUCs of 0.95 and 0.87 respectively. Finally, test 10 used a model pretrained without any downsampling, test 33 above. The performance was the same as with no transfer learning (AUC 0.94). We did not calculate confidence intervals for these intermediate experiments, but given the size of the dataset, AUC differences of a couple of hundredths are unlikely to be significant. Hence, none of the experiments using pretrained models performed better than the CNN trained from scratch in test 1.

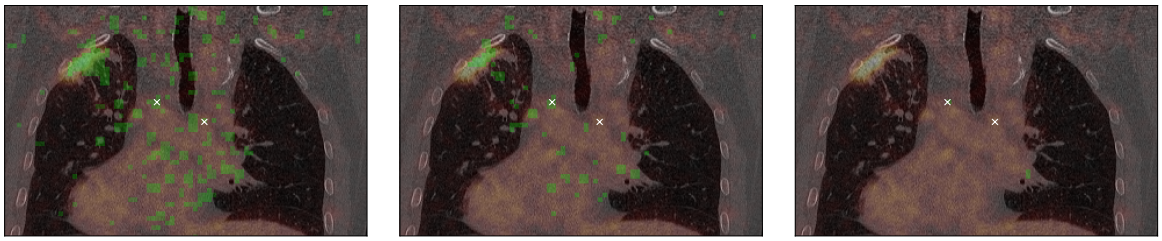
The AUC values attained here are very good, however, the validation set only contained a limited number of negative cubes per patient. To fully evaluate the effectiveness of the method, we then tested the best-performing model (we chose test 5, although there was little difference between several) on the full validation set images. Some example outputs using different thresholds are shown in Figure 5.15. We see that with a threshold of 0.7 far too many regions are predicted as positive. Even at a threshold of 0.9 we have too many false positive regions yet also miss true positives. At a threshold of 0.99 we finally reduce the number of false positives but have a lot of false negatives. Clearly the accuracy of the model is not good enough. Because the results on the validation set were disappointing, we did not use the test set.

5.7.4 Discussion

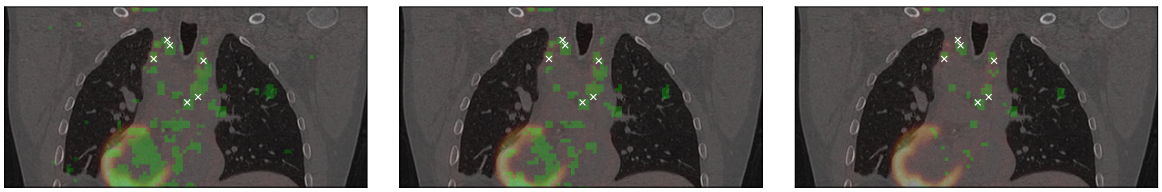
The PET MIP approach did not give satisfactory results, with the CNN not learning at all. We had fewer patients for this method, so it cannot be compared directly to the other results in this section. There were many parameters that we did not test (such as learning rate, batch size, and dropout). As is often the case in deep learning, there are endless permutations to try, but all of them take time. Having seen no evidence of the models learning anything from the MIP images, we decided to move on and try a different method.



(a) Example 1



(b) Example 2



(c) Example 3

Threshold=0.7 , Threshold=0.9 Threshold=0.99

Fig. 5.15 Predicted positive regions for three patients at three thresholds using the 3D cube approach, test 5. Predicted positive regions are shown in green and have been projected onto the 2D plane so we can see all regions in one image. White crosses represent true positive nodes

Table 5.7 Results - 3D cube approach

Test ID	Pretrained Model	Model	Modality	c/mm	Augmentation	N_{neg}	Batch Size	Val AUC
1	None	$Cube_1$	BOTH	30	Flip Rot	150	32	0.94
2	None	$Cube_2$	CT	30	Flip Rot	150	32	0.69
3	None	$Cube_2$	PET	30	Flip Rot	150	32	0.95
4	test 22	$Cube_3$	BOTH	30	None	150	16	0.91
5	test 22	$Cube_3$	BOTH	30	Flip Rot	150	16	0.96
6	test 22	$Cube_3$	BOTH	30	Flip Rot	250	16	0.96
7	test 22	$Cube_3$	BOTH	30	Flip Rot	600	32	0.93
8	test 22	$Cube_3$	BOTH	50	Flip Rot	150	32	0.95
9	test 22	$Cube_3$	BOTH	70	Flip Rot	150	32	0.87
10	test 33	$Cube_3$	BOTH	30	Flip Rot	150	32	0.94

Using the 3D cube approach the results improved; clearly the model was learning something, but were still disappointing. When classifying the full scans (with cubes at every coordinate) the model did not perform well, with far too many false positives. This is perhaps because, with only a small selection of negative cubes in the training set, there were not enough *difficult* cases for the CNN to learn. However, using all the cubes for training would have made the dataset far too large and unbalanced. We successfully trained a self-supervised network, but this did not improve the performance significantly. These results led us to try two-phase approaches, as elaborated in the next section.

5.8 Two-Phase Approaches (Phase 1) - Method

From the above experiments a one-phase classification approach was not achieving the accuracy we required. The mediastinal nodes occupied too little of the overall scans to give accurate results. We therefore decided to try a two-phase classification approach. The first phase would be a coarse sorting, which would find potentially suspicious regions. The key was to have a high sensitivity on this first phase so as not to miss any true positives. The second phase would then definitively classify these regions as positive or negative. Two-phase approaches have been used in similar tasks such as CT lung nodule identification [193] [194] and PET/CT lymphoma classification [186]. We tested three approaches in this first phase; a PET threshold, radiomic features, and a U-Net segmentation. These are detailed in Figure 5.16.

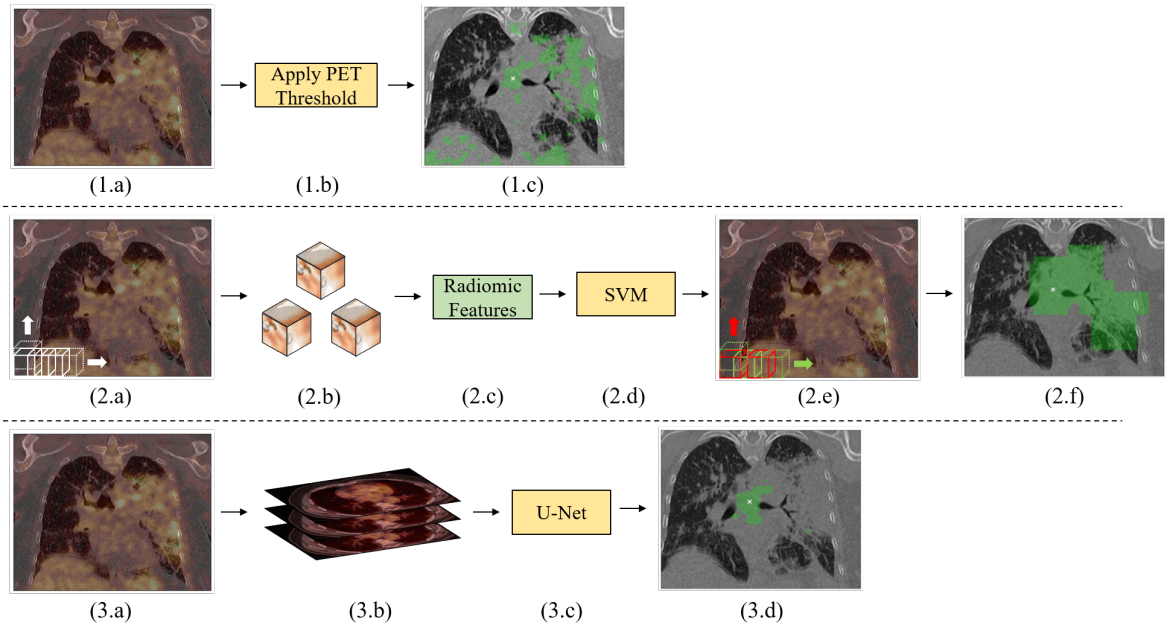


Fig. 5.16 Summary of the three approaches tested to identify suspicious regions (phase one of our two-phase approach). (1) shows the PET threshold approach. In (2), the radiomic features approach, the scans are first divided into overlapping cubes (2.b), then texture features are extracted (2.c) before they are input into an SVM (2.d). This SVM then classifies the cubes as suspicious or not (2.e). In the U-Net approach (3) the images are fed slice-by-slice into a U-Net (3.c), which predicts suspicious regions (3.d)

5.8.1 PET Threshold Approach

Introduction In a clinical setting a PET uptake of >2.5 SUV is often used as a cut-off to determine if a node is pathological or not (combined with visual inspection) and was used in [195] to identify pathological mediastinal lymph nodes with sensitivity and specificity scores of 89% and 84% respectively (though in this study the lymph nodes had already been localised). This therefore seemed a sensible way to create candidate regions and narrow down our search volume.

Method As in the one-phase 3D cube approach, we first resampled the scans to have isotropic voxels of size 1x1x1 mm^3. They were also all cropped symmetrically to be 256x256 pixels in-plane. This was to make them consistent with the U-Net method below (which, using a CNN, needed all input slices to be the same size). We tested two thresholds, 2 SUV and 2.5 SUV.

To evaluate the threshold effectiveness, we calculated the total volume of the thresholded regions across the test set. We also counted the number of nodes missed (due to the threshold being too high).

5.8.2 Radiomic Feature Approach

Introduction As explained in Chapter 2, radiomic texture features have been used in a wide range of medical imaging contexts. Given their success in similar domains, and their ease of use (compared to fine-tuning a neural network), they were a suitable choice for phase one of the process.

Method Again, we first resampled the scans to have isotropic voxels of size 1x1x1 mm^3 and cropped them to 256x256 pixels in-plane. To find the radiomic features, we cut cubes of side length 30 mm at regular intervals of 15 mm in all axes from the scans. Positive cubes were oversampled by a factor of eight. Any cubes with a maximum PET value lower than 2 SUV were removed (as these were clearly negative). This gave 91,055 cubes in the training set, 6,818 of which were positive. From these cubes we calculated 682 handcrafted features using PyRadiomics (341 for both PET and CT). First-order, GLCM, GLSZM, GLRLM, NGTDM, GLDM, and first-order features derived after applying LoG (with $\sigma = 1, 2, 3$), wavelet, exponential, and gradient filters were used (see Appendix B for full details). We reduced the number of features using a two-stage process; feature numbers were first reduced using a univariate ANOVA F-test, then any features with a correlation greater than 0.95 were removed (for two correlated features, the mean absolute correlation of each feature was

Table 5.8 Parameters tested - radiomic feature approach

Parameter	Values Tested
No. Features after ANOVA F-Test	60 80 100 120 140 160 180
SVM Kernel Type	linear RBF
SVM C-Value	0.1 0.5 1 2 5

calculated and the feature with the largest mean absolute correlation with respect to all other features was removed). Before analysis, features were scaled to have zero mean and unit variance (using the training set to fit the scaling). To ensure the SVM was optimised, we tested a number of F-test thresholds and tested linear and RBF kernels with various SVM C values. These were compared using the AUC score on the validation set. Full details of the parameters tested are in Table 5.10.

To compare this to the other approaches we calculated the total volume of candidate regions on the test set (taking into account the overlapping of the cubes) and counted the missed nodes.

5.8.3 U-Net Approach

Introduction Given U-Net's success in medical segmentation tasks, it seemed a good candidate for the first phase of finding suspicious regions. Its encoder-decoder architecture means it outputs a segmented volume, and the segmented regions can be used as candidates for phase two. A similar setup using a U-Net has been used in other two-phase network designs, such as for CT lung classification [193].

Method We did not have actual segmentations of the nodes so instead used spheres to indicate the node regions. Scans were again resampled to have isotropic voxels of size 1x1x1 mm^3 and symmetrically cropped to have in-plane dimensions of 256x256 pixels. We created binary masks of the volumes, with *True* values within 15 mm of positive nodes and *False* values everywhere else (most mediastinal lymph nodes are of diameter 0-20 mm [192]). These were used as the labels to train a 2D U-Net on a slice-by-slice basis with the training data being 256x256x2 PET/CT slices. Only slices for which the binary mask had some true values (so were within 15 mm of a positive node) were used in training. We used this whole-slice view so the network had a full physiological view of the slices and could learn to only pick candidates within the mediastinum. We trained the network for 20 epochs (by which point the validation loss had stopped decreasing) with a cross-entropy loss and an

Table 5.9 Parameters tested - U-Net approach

Parameter	Values Tested
Batch Size	2 4
Learning Rate	10^{-4} 10^{-5}

Adam optimiser. As shown in Table 5.9, a few experiments were run varying the learning rate and batch size. These were compared using the DICE score on the validation set.

Once trained, for each patient the entire 3D volume was input into the U-Net slice-by-slice, and a set of contiguous candidate regions were produced. As with the above methods, the total volume of the candidate regions across the test set was calculated and the number of missed nodes was counted.

5.9 Two-Phase Approaches (Phase 1) - Results and Discussion

5.9.1 PET Threshold Approach

Applying a PET threshold of 2.5 SUV across the whole test set left a volume of 17,261,815 voxels (out of a total volume of 1,854,911,547 voxels), meaning the volume had been reduced by 99.1%. However, four positive nodes were missed (all between 2 and 2.5 SUV). Using a PET threshold of 2 SUV left a volume of 28,913,460 voxels.

5.9.2 Radiomic Feature Approach

Table 5.10 shows the AUC scores for the different SVM optimisation tests. The choice of SVM parameters did not greatly affect the performance, with all tests achieving scores between 0.85 and 0.88. Using a model with the 140 best univariate features, a linear kernel, and $C = 1$ (test 5) on the test set left a total volume of 11,546,199 voxels (a reduction of 99.4%). Three positive nodes were missed.

5.9.3 U-Net Approach

Table 5.11 shows the results for the different U-Net optimisation tests. Using test 1 (learning rate 10^{-3} and batch size two) to classify the test set left a total volume of 3,810,241 voxels (reducing the volume by 99.8%) with one node missed.

Table 5.10 Results - radiomic feature approach

Test ID	Features after ANOVA F-Test	Kernel	C-Value	AUC
1	60	linear	1	0.85
2	80	linear	1	0.86
3	100	linear	1	0.86
4	120	linear	1	0.87
5	140	linear	1	0.88
6	160	linear	1	0.88
7	180	linear	1	0.88
8	140	linear	0.1	0.88
9	140	linear	0.5	0.88
10	140	linear	1	0.88
11	140	linear	2	0.88
12	140	linear	5	0.88
13	140	RBF	0.1	0.88
14	140	RBF	0.5	0.88
15	140	RBF	1	0.88
16	140	RBF	2	0.88
17	140	RBF	5	0.88

Table 5.11 Results - U-Net approach

Test ID	Batch Size	Learning Rate	DICE Score
1	2	10^{-4}	0.39
2	4	10^{-4}	0.02
3	2	10^{-5}	0.02
4	4	10^{-5}	0.32

Table 5.12 Results - two-phase approaches (phase one)

Method	Test Volume Remaining	Nodes Missed
PET Threshold 2.5 SUV	17,261,815	4
PET Threshold 2 SUV	28,913,460	0
Radiomic Features	11,546,199	3
U-Net	3,810,241	1

5.9.4 Discussion

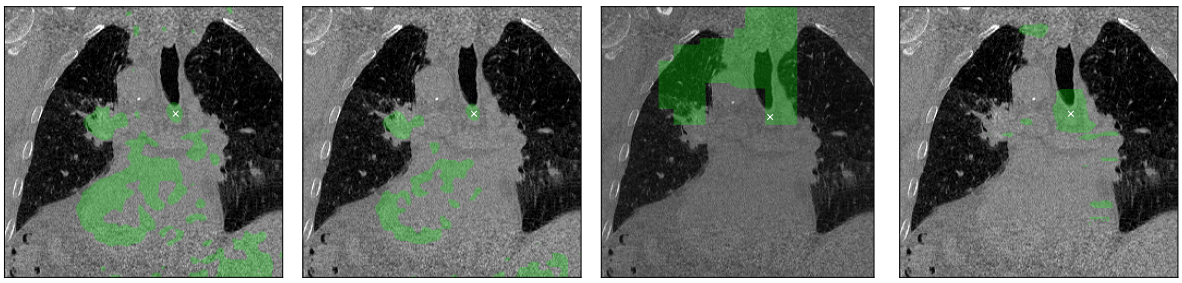
Table 5.12 summarises the results. Figure 5.17 gives some examples applied to images. Both PET thresholds left a large volume remaining compared to the other methods. Looking at the examples we see there are large regions of high uptake in the abdomen. At 2.5 SUV we also missed four positive nodes. The radiomic feature-based results were more promising, leaving a smaller volume and missing only three nodes. From the examples we see the remaining regions are more consistently in the mediastinal region, rather than in the abdmomen. However, example 2 still has a large suspicious region in the right lung (which, not being in the mediastinum we are not interested in). Finally, the U-Net method was much more parsimonious with its volume selection, leaving approximately 3x less volume than the radiomic feature method and 5x less than a PET threshold of 2.5 SUV. The examples show that the selection was much more focused, with small individual regions compared to the other methods. Additionally it only missed one node, fewer than the 2.5 SUV threshold and the radiomic feature approach.

We concluded that a U-Net based model would clearly be the best method for phase one. We did not spend much time optimising the model at this stage and thought that with further improvements we would be able to improve the performance. In the next section we systematically test different parameters to try and improve these results.

5.10 Phase One Optimisation - Method

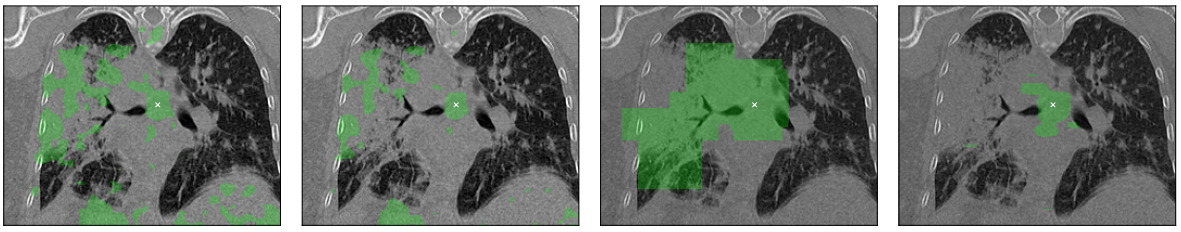
5.10.1 Introduction

The general method for phase one (using a U-Net for segmentation) has already been outlined in 5.8.3 above. However, this was far from optimised. With such a large number of parameters to optimise, and the time taken to train each CNN, it was unrealistic to try every combination of parameters. Instead we split the optimisation process into stages. Here we optimise phase



2 SUV Threshold | 2.5 SUV Threshold | Radiomic Features | U-Net

Example 1



2 SUV Threshold | 2.5 SUV Threshold | Radiomic Features | U-Net

Example 2

Fig. 5.17 Outputs of the different phase one approaches for two patients. The thresholded areas are shown in green, with white crosses marking positive nodes. The CT is shown in greyscale

one of the process, fine-tuning the U-Net and its associated parameters to reduce the volume of suspicious regions as much as possible, while not missing any true positive nodes.

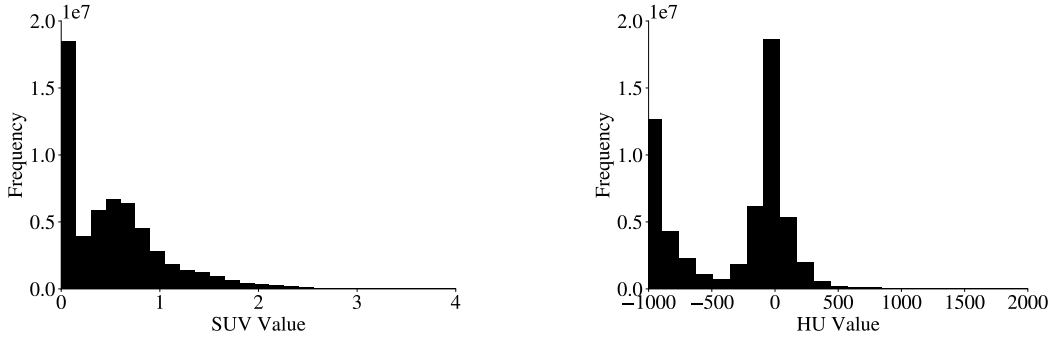
5.10.2 Parameters Optimised

The method is as in 5.8.3, but in this section we spend time optimising the parameters more carefully. Below is an explanation of all of the parameters we optimised, which are summarised in Table 5.13. In each case, as above the U-Net was trained for 20 epochs (or stopped early if it was clear that the validation loss had stopped decreasing). The experiments were evaluated on the validation set using the DICE score.

Voxel Size In the initial preprocessing we upsampled the PET to have the same resolution as the CT. However, there was no guarantee that this was the best resolution. Choosing a higher resolution than the CT makes no sense; this would add no more information, just make the images larger. Lowering the resolution, while losing some information from the CT, would make the images and therefore the CNN smaller, and would mean less interpolation was needed for the PET images. This reduction in the number of parameters could mean that the network's performance would improve. Previous work has shown that the impact of resolution reduction varies depending on the dataset [196]. We therefore trialled voxel sizes of 1 mm, 1.5 mm, and 2 mm. In each case the images were rescaled using cubic spline interpolation.

Thresholding PET The vast majority of SUV values on the scans were within the range 0-4 SUV (see Figure 5.18a for an example). However, the maximum value of the entire dataset was 71 SUV. When standardising, this can give a very skewed dataset, with most values at the lower end of the scale. A way to reduce this problem is by imposing a cutoff, whereby every value above the cutoff is set to the cutoff value. Something similar was done in [197], the best-performing model on the LUNA lung CT classification challenge. For the PET we tried cutoff values of 4, 8, and 12 SUV (denoted PET_{max} in the results).

Thresholding CT Similarly for the CT, the minimum and maximum HU values were -1024 and 2976 HU respectively (although these changed slightly when resampling). Figure 5.18b shows a typical HU distribution. We tested no lower cutoff and a lower cutoff of -1000 HU, and upper cutoff values of 400, 1000, and 2000 HU (denoted CT_{min} and CT_{max} respectively in the results).



(a) Typical SUV distribution of a PET scan

(b) Typical HU distribution of a CT scan

Fig. 5.18 Typical voxel value distributions for PET and CT scans

Voxel Value Rescaling As explained in 4.1, it is common practice to rescale image values before training CNNs. This helps the stability and convergence of a model, but there are several methods and no consensus on which is the best [198] [199]. We therefore tested rescaling to between zero and one (denoted [0,1]), and rescaling to give the dataset zero mean and unit variance (denoted $[\mu, \sigma]$).

Radius of Spheres As mentioned above we did not have actual segmentations of the nodes, instead using spheres of radii 15 mm centred on positive coordinates. Mediastinal lymph nodes are typically of size 10-30 mm, but how different radii would affect the U-Net performance was unclear. Smaller radii, while guaranteeing better localisation of the node, give a smaller positive region for the U-Net to train on. We tried three different sphere radii: 5, 10, and 15 mm (denoted r in the results).

Negative Slices In the above section, to train the U-Net we only used slices containing positive nodes. When inputting whole scans there will be many slices without any positive regions. It may therefore be beneficial to train the U-Net with some negative slices as well as the positive ones. The disadvantage of this is that it makes the classes even more unbalanced. We ran experiments with 0, 5, and 10 random negative slices per patient (denoted N_{neg} in the results).

Model U-Net being such a well-established model for segmentation, we did not spend much time trying other models or building our own [112]. Given our small dataset size, we did try a U-Net model with half the number of filter maps in each layer (denoted U-Net_{small}).

Table 5.13 Parameters tested - phase one optimisation

Parameter	**Values Tested**
Voxel Size/mm	1 1.5 2
PET Threshold (PET_{max})/SUV	4 8 12
CT Lower Threshold (CT_{min})/HU	-1000
CT Upper Threshold (CT_{max})/HU	400 1000 2000
Rescaling	[0,1] [μ, σ]
Sphere Radii (r)/mm	5 10 15
Negative slices per patient	0 5 10
Model	U-Net U-Net_{small}
Loss	CE DICE
Learning Rate	10^{-4} 10^{-5}
Augmentation	Flips Rotations Translations Zooming

Loss Function The dataset is very imbalanced, so it was important to carefully select a loss function. Studies commonly use cross-entropy [200] [111] or DICE loss [201] [202], but there are a wealth of other loss functions tailored to segmentation tasks [203] [204]. We limited ourselves to running experiments testing cross-entropy (denoted CE) and DICE loss.

Learning Rate and Augmentation Learning rates of 10^{-4} and 10^{-5} and random flips, rotations $\in [-20°, 20°]$, translations $\in [\pm 20\%]$ in both axes, and random zooms $\in [\pm 20\%]$ were all tested as in-training augmentation.

Dilation and Minimum Volume After finding the optimum parameters for the network, any candidate regions smaller than a certain number of voxels were removed. We then dilated the candidate regions to ensure that a reasonable area around the positive nodes was captured. The values for these were found by visually inspecting the training set outputs.

5.11 Phase One Optimisation - Results and Discussion

The results for all experiments are shown in Tables 5.14 to 5.17. Table 5.14 shows tests 1-36, which used different thresholds for PET and CT at learning rates of 10^{-4} and 10^{-5}. For all of these tests we used sphere radii of 15 mm, [μ, σ] rescaling, zero negatives slices per patient, a U-Net model, cross-entropy loss, no augmentation, and 1 mm voxel resolution. Several tests improved on the DICE scores in the previous section, indicating that using thresholds is advantageous. The best-performing model was test 25 with a DICE score of 0.47, but several tests achieved DICE scores greater than 0.40 (highlighted). There was no

clear pattern to these high-performing tests, so no threshold can be immediately discounted. This highlights one of the problems with optimising neural networks; optimising parameters individually will not necessarily work. They are unpredictable, and the results do not follow an obvious pattern or intuition. The reasoning in the subsequent tests was therefore not to necessarily find *the best* set of parameters, which could require testing all combinations of parameters (temporally unfeasible), but rather to find a set of parameters that gave a near-optimal performance. The tests with a learning rate of 10^{-5} on average performed better than those with a learning rate of 10^{-4}, so we limited ourselves to a learning rate of 10^{-5} in future tests. We also only used PET_{max} thresholds of 4 and 8 SUV, because using a PET_{max} of 12 SUV did not give significantly better results.

Table 5.15 details experiments testing DICE loss, rescaling methods, and sphere radius sizes. For all these tests we used a U-Net model with no negative slices per patient, a learning rate of 10^{-5}, no augmentation, and a voxel resolution of 1 mm. Tests 37-44 used [0,1] rescaling. These results are not as good as the tests using [μ, σ] rescaling above, so we discounted this option in further experiments. The results using DICE loss (tests 45-55) were very similar to those using cross-entropy loss, but we decided to use DICE loss in the future because the best performance was slightly higher (test 54). This is not to say that DICE loss is always better than cross-entropy loss (as shown above, the results are highly unpredictable and in many cases cross-entropy loss performed well) but is rather to reduce the number of subsequent tests to perform. Using spheres of radii 5 mm (tests 56-67) and 10 mm (tests 68-79) did not result in good results (although as the positive volume was smaller for these tests they cannot be directly compared to the above). This is most likely because the models had less positive volume to train on.

The tests in Table 5.16 trained the model with some negative slices and using augmentation. For all experiments we used a U-Net model with a learning rate of 10^{-5}, voxel resolution 1 mm, DICE loss, [μ,σ] rescaling, and sphere radii of 15 mm. None of the models trained using negative slices performed well (tests 80-103). This again may be because the class imbalance becomes too great. Tests 104-111 employ in-training augmentation flip, rotation, shift, and zooming operations. Test 104 (using only flips with CT thresholds of -1000 HU and 2000 HU and a PET threshold of 4 SUV) and test 111 (using flips, rotations, shifts, and zooms with CT thresholds of -1000 HU and 2000 HU and a PET threshold of 8 SUV) were the best-performing, achieving DICE scores of 0.49 and 0.42 respectively.

Finally, in Table 5.17 we tested the U-Net_{small} model (tests 111-116) and different voxel resolutions (test 117-128). In all cases the sphere radii were 15 mm, the learning rate was 10^{-5}, there were no negative slices or augmentation, and [μ,σ] rescaling was used. Some of the U-Net_{small} models performed well, with tests 114 and 116 attaining DICE scores of

0.43 and 0.44 respectively. Using voxel resolutions of 1.5 mm and 2 mm, several models achieved DICE scores greater than 0.35 (tests 119, 122, 123, 124, 127). However, there was no notable increase in performance compared to the above tests.

Our ultimate goal in this phase was to reduce the total volume of suspicious regions while maintaining a high sensitivity. Given we had several similar results, for a few of the best-performing tests (25, 49, 53, and 104) we then calculated the overall volume and number of missed nodes on the entire validation set. These results are shown in Table 5.18. All four models performed reasonably well, with test 53 leaving the least volume. It missed one node, but we deemed this a better result than test 104, which left over twice the volume. We therefore selected test 53 to use for phase one.

To briefly test the stability of the result and ensure retraining the model would not give a wildly different performance, we reran test 53 twice, using exactly the same experimental setup. The DICE scores for the repeated tests were 0.48, 0.47, and 0.48, assuaging our fears.

Finally, small volumes were removed and the remaining volumes dilated. The minimum volume containing a positive node in the validation set was 1800 voxels, but to ensure that we did not overenthusiastically dispose of relevant regions we only removed volumes smaller than 300 voxels. Because we use a sliding window approach in the next section, we wanted to keep substantial regions around the nodes. To do this we iteratively dilated the volumes three times with a squared connectivity of one. Some examples of volumes before and after these operations are shown in Figure 5.19.

5.11.1 Discussion

Although lengthy, this set of experiments is far from exhaustive. As mentioned, there are endless possible parameter combinations, and it would be impossible to test them all. Comparing the DICE score of the best-performing model (test 53, DICE 0.48) with that of the best-performing model built in 5.8.3 (test 1, DICE 0.39), we see that there is not an enormous difference. Many of the tests in this section achieved DICE scores in the range of 0.40-0.50, suggesting that this is around the best performance possible with this set of data.

Before moving onto the next phase we retrained the model, including the validation data in training. With so much parameter optimisation in this section there was a danger we would have slightly overfit on the validation data. The DICE score on the test set was 0.44, indicating that this was not the case.

Table 5.14 Results - phase one optimisation (part 1). In all cases the radius of spheres was 15 mm, there were no negative slices or augmentation, voxel resolution was 1 mm, and [μ,σ] rescaling with a standard U-Net model and cross-entropy loss were used. Results with DICE scores of 0.40 or over are highlighted in blue

Test ID	***lr***	**CT_{min}/HU**	**CT_{max}/HU**	**PET_{max}/SUV**	**DICE**
1	10^{-4}	-	400	4	0.38
2	10^{-4}	-	400	8	0.33
3	10^{-4}	-	400	12	0.45
4	10^{-4}	-	1000	4	0.31
5	10^{-4}	-	1000	8	0.38
6	10^{-4}	-	1000	12	0.43
7	10^{-4}	-	2000	4	0.31
8	10^{-4}	-	2000	8	0.02
9	10^{-4}	-	2000	12	0.02
10	10^{-4}	-1000	400	4	0.32
11	10^{-4}	-1000	400	8	0.36
12	10^{-4}	-1000	400	12	0.39
13	10^{-4}	-1000	1000	4	0.33
14	10^{-4}	-1000	1000	8	0.36
15	10^{-4}	-1000	1000	12	0.25
16	10^{-4}	-1000	2000	4	0.33
17	10^{-4}	-1000	2000	8	0.34
18	10^{-4}	-1000	2000	12	0.37
19	10^{-5}	-	400	4	0.02
20	10^{-5}	-	400	8	0.32
21	10^{-5}	-	400	12	0.33
22	10^{-5}	-	1000	4	0.35
23	10^{-5}	-	1000	8	0.31
24	10^{-5}	-	1000	12	0.29
25	10^{-5}	-	2000	4	0.47
26	10^{-5}	-	2000	8	0.43
27	10^{-5}	-	2000	12	0.40
28	10^{-5}	-1000	400	4	0.45
29	10^{-5}	-1000	400	8	0.34
30	10^{-5}	-1000	400	12	0.32
31	10^{-5}	-1000	1000	4	0.33
32	10^{-5}	-1000	1000	8	0.45
33	10^{-5}	-1000	1000	12	0.43
34	10^{-5}	-1000	2000	4	0.35
35	10^{-5}	-1000	2000	8	0.33
36	10^{-5}	-1000	2000	12	0.45

Table 5.15 Results - phase one optimisation (part 2). In all cases the learning rate was 10^{-5}, there were no negative slices or augmentation, voxel resolution was 1 mm, and a standard U-Net model were used. Results with DICE scores of 0.40 or over are highlighted in blue

Test ID	Loss	CT_{min}/HU	CT_{max}/HU	PET_{max}/SUV	Rescaling	r/mm	DICE
37	CE	-	400	4	[0,1]	15	0.20
38	CE	-	400	8	[0,1]	15	0.21
39	CE	-	2000	4	[0,1]	15	0.02
40	CE	-	2000	8	[0,1]	15	0.21
41	CE	-1000	400	4	[0,1]	15	0.21
42	CE	-1000	400	8	[0,1]	15	0.20
43	CE	-1000	2000	4	[0,1]	15	0.22
44	CE	-1000	2000	8	[0,1]	15	0.00
45	DICE	-	400	4	$[\mu,\sigma]$	15	0.04
45	DICE	-	400	8	$[\mu,\sigma]$	15	0.46
46	DICE	-	1000	4	$[\mu,\sigma]$	15	0.04
47	DICE	-	1000	8	$[\mu,\sigma]$	15	0.34
48	DICE	-	2000	4	$[\mu,\sigma]$	15	0.46
49	DICE	-	2000	8	$[\mu,\sigma]$	15	0.47
50	DICE	-1000	400	4	$[\mu,\sigma]$	15	0.03
51	DICE	-1000	400	8	$[\mu,\sigma]$	15	0.34
52	DICE	-1000	1000	4	$[\mu,\sigma]$	15	0.35
53	DICE	-1000	1000	8	$[\mu,\sigma]$	15	0.48
54	DICE	-1000	2000	4	$[\mu,\sigma]$	15	0.46
55	DICE	-1000	2000	8	$[\mu,\sigma]$	15	0.35
56	DICE	-	400	4	$[\mu,\sigma]$	5	0.00
57	DICE	-	400	8	$[\mu,\sigma]$	5	0.00
58	DICE	-	1000	4	$[\mu,\sigma]$	5	0.00
59	DICE	-	1000	8	$[\mu,\sigma]$	5	0.00
60	DICE	-	2000	4	$[\mu,\sigma]$	5	0.13
61	DICE	-	2000	8	$[\mu,\sigma]$	5	0.00
62	DICE	-1000	400	4	$[\mu,\sigma]$	5	0.00
63	DICE	-1000	400	8	$[\mu,\sigma]$	5	0.00
64	DICE	-1000	1000	4	$[\mu,\sigma]$	5	0.00
65	DICE	-1000	1000	8	$[\mu,\sigma]$	5	0.00
66	DICE	-1000	2000	4	$[\mu,\sigma]$	5	0.00
67	DICE	-1000	2000	8	$[\mu,\sigma]$	5	0.00
68	DICE	-	400	4	$[\mu,\sigma]$	10	0.00
69	DICE	-	400	8	$[\mu,\sigma]$	10	0.02
70	DICE	-	1000	4	$[\mu,\sigma]$	10	0.00
71	DICE	-	1000	8	$[\mu,\sigma]$	10	0.34
72	DICE	-	2000	4	$[\mu,\sigma]$	10	0.00
73	DICE	-	2000	8	$[\mu,\sigma]$	10	0.02
74	DICE	-1000	400	4	$[\mu,\sigma]$	10	0.00
75	DICE	-1000	400	8	$[\mu,\sigma]$	10	0.00
76	DICE	-1000	1000	4	$[\mu,\sigma]$	10	0.00
77	DICE	-1000	1000	8	$[\mu,\sigma]$	10	0.32
78	DICE	-1000	2000	4	$[\mu,\sigma]$	10	0.35
79	DICE	-1000	2000	8	$[\mu,\sigma]$	10	0.02

Table 5.16 Results - phase one optimisation (part 3). In all cases the radius of spheres was 15 mm, the learning rate was 10^{-5}, voxel resolution was 1 mm, and $[\mu,\sigma]$ rescaling with a standard U-Net model and DICE loss were used. Results with DICE scores of 0.40 or over are highlighted in blue

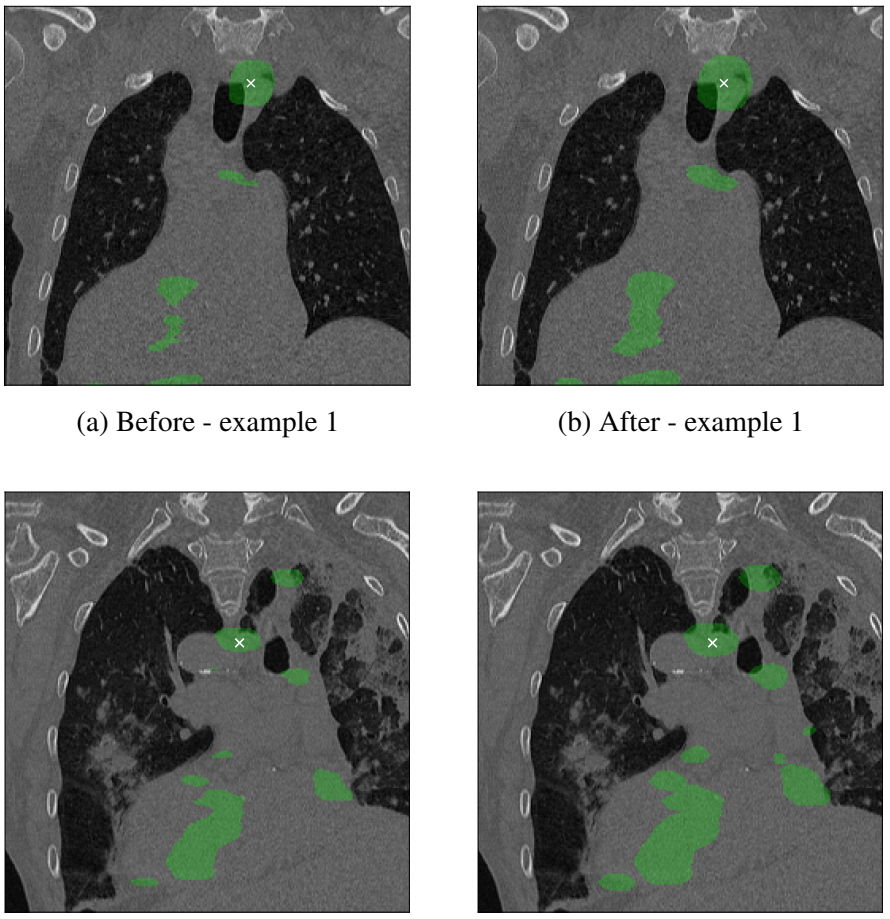
Test ID	**CT_{min}/HU**	**CT_{max}/HU**	**PET_{max}/SUV**	**N_{neg}**	**Augmentation**	**DICE**
80	-	400	4	5	None	0.00
81	-	400	8	5	None	0.03
82	-	1000	4	5	None	0.00
83	-	1000	8	5	None	0.00
84	-	2000	4	5	None	0.03
85	-	2000	8	5	None	0.03
86	-1000	400	4	5	None	0.00
87	-1000	400	8	5	None	0.00
88	-1000	1000	4	5	None	0.00
89	-1000	1000	8	5	None	0.03
90	-1000	2000	4	5	None	0.00
91	-1000	2000	8	5	None	0.00
92	-	400	4	10	None	0.03
93	-	400	8	10	None	0.07
94	-	1000	4	10	None	0.07
95	-	1000	8	10	None	0.07
96	-	2000	4	10	None	0.07
97	-	2000	8	10	None	0.03
98	-1000	400	4	10	None	0.07
99	-1000	400	8	10	None	0.07
100	-1000	1000	4	10	None	0.07
101	-1000	1000	8	10	None	0.07
102	-1000	2000	4	10	None	0.07
103	-1000	2000	8	10	None	0.00
104	-1000	2000	4	0	Flip	0.49
105	-1000	2000	8	0	Flip	0.39
106	-1000	2000	4	0	Flip Rot	0.33
107	-1000	2000	8	0	Flip Rot	0.34
108	-1000	2000	4	0	Flip Rot Trans	0.00
109	-1000	2000	8	0	Flip Rot Trans	0.00
110	-1000	2000	4	0	Flip Rot Trans Zoom	0.34
111	-1000	2000	8	0	Flip Rot Trans Zoom	0.42

Table 5.17 Results - phase one optimisation (part 4). In all cases the radius of spheres was 15 mm, the learning rate was 10^{-5}, there were no negative slices or augmentation, and [μ,σ] rescaling was used. Results with DICE scores of 0.40 or over are highlighted in blue

Test ID	CT_{min}/HU	CT_{max}/HU	PET_{max}/SUV	Model	Voxel Size	DICE
111	-1000	400	4	U-Net_{small}	1	0.04
112	-1000	400	4	U-Net_{small}	1	0.35
113	-1000	1000	4	U-Net_{small}	1	0.36
114	-1000	1000	4	U-Net_{small}	1	0.43
115	-1000	2000	4	U-Net_{small}	1	0.04
116	-1000	2000	4	U-Net_{small}	1	0.44
117	-1000	400	4	U-Net	1.5	0.05
118	-1000	400	8	U-Net	1.5	0.05
119	-1000	1000	4	U-Net	1.5	0.39
120	-1000	1000	8	U-Net	1.5	0.05
121	-1000	2000	4	U-Net	1.5	0.04
122	-1000	2000	8	U-Net	1.5	0.39
123	-1000	400	4	U-Net	2	0.36
124	-1000	400	8	U-Net	2	0.35
125	-1000	1000	4	U-Net	2	0.34
126	-1000	1000	8	U-Net	2	0.05
127	-1000	2000	4	U-Net	2	0.36
128	-1000	2000	8	U-Net	2	0.31

Table 5.18 Results - two-phase approaches (phase one). Volume remaining and nodes missed for four of the best-performing models

Test ID	Volume Remaining	Nodes Missed
25	2,657,921	1
49	2,681,895	1
53	2,191,302	1
104	5,126,446	0



(a) Before - example 1

(b) After - example 1

(c) Before - example 2

(d) After - example 2

Fig. 5.19 Examples of outputs from phase one before and after removing small volumes and dilating the remaining volumes. Regions selected by the model are in green and true positive nodes are indicated by white crosses

5.12 Phase Two Optimisation - Method

From phase one the volume that could contain pathological nodes had been reduced, leaving a set of suspicious regions. But in some cases the regions were large, meaning just predicting the entire region as pathological would not be useful; the location of the pathological node would not be precisely located. We therefore split the regions into 3D cubes, similarly to in the one-phase 3D cube approach in 5.6.2. We cut the cubes at 8 mm intervals in the x, y, and z axes, forming a regular grid. This gave 46,427 cubes for the training set (comprising 4,235 positive cubes and 42,192 negative cubes). These cubes were used to train a 3D CNN for classification. Models were trained using an Adam optimiser with cross-entropy loss for 20 epochs (or stopped early if it was clear the validation loss was not decreasing). In some instances we tried using transfer learning, using the same pretrained networks (trained on the jigsaw task) as in 5.6.2. Again, a range of parameters were optimised, detailed in Table 5.19.

5.12.1 Parameters Optimised

Phase One Approach Initially we used the radiomic feature approach in phase one, having only thought of using a U-Net approach later in our thesis. The task in this second phase, categorising 3D cubes as positive or negative, is unchanged, thus these results were still useful for optimisation. However, they cannot be directly compared with results that used a U-Net-based phase one approach. In these cases voxel values were rescaled between zero and one (rather than using $[\mu, \sigma]$ rescaling).

Model Pretraining Although the self-supervised pretraining did not confer any advantage when used in 5.6.2, we thought it was still worth trying here. For some experiments we used models pretrained on the jigsaw task in 5.6.2, freezing some layers during training. Tests here refer to the models trained in Table 5.6.

Model Several 3D CNN architectures were tested with the full details of each in Appendix C.4. We tested a range of architectures based on papers that used CNNs to tackle similar problems, as well as a 3D 50-layer ResNet [119] [190] [205]. These also built on the results from the one-phase 3D cube approach in 5.6.2.

Cube Size The cube size is clearly an important choice when optimising the network. The cubes need to be large enough to capture relevant spatial information, but cubes too large would lack the precision to localise the node. When using the texture feature approach in phase one we tested cubes of size 20 mm, 30 mm, and 50 mm. We also tested combining

models using these three cube sizes, the details of which are in Appendix D. After switching to the U-Net approach in phase one, we tested cubes of size 32 mm and 64 mm. This change was made so that the dimensions of CNN layer outputs were always divisible by two and therefore matched the max-pooling stride of two. The cube size is denoted c in the results.

Node Range The node range is the distance from the centre of the cube that a node has to be for the cube to be labelled as positive. The obvious choice is to use the dimension of the cube. However, if a node were to fall on the edge or the corner of the cube there may not be enough information to reliably classify the cube. We therefore tested node ranges of half the cube diameter and the full cube diameter.

Undersampling of Negative Nodes Despite phase one, there was still an imbalance in the positive to negative ratio of the cubes. For training we therefore undersampled the negative nodes. Aside from balancing the dataset, this was done to speed up training. The size of the input data (46,427x64x64x64x2 for cubes of side length 64 mm) meant that one training run using all the data would take several days. With many hyperparameters to optimise this was unfeasible, and undersampling the negative nodes sped up training considerably with a negligible effect on performance. For most experiments we therefore undersampled the negative nodes by a factor of eight. We ran one test undersampling by a factor of four to see the effect on the performance.

Modality We tested models trained using PET/CT, PET-only, and CT-only inputs, to see how important multi-model imaging is to the performance.

Batch Size, Learning Rate, and Augmentation Learning rates of 10^{-3}, 10^{-4} and 10^{-5} and batch sizes of 8, 16, and 32 were tested. Random flips, rotations, and zooming in all three axes were tested as in-training augmentation.

5.13 Phase Two Optimisation - Results and Discussion

Table 5.20 shows the results when using cubes derived from the radiomic feature approach in phase one. Tests 1-4 used the pretrained model from test 22, freezing different numbers of layers. None of these tests resulted in a performance above 0.55 AUC. Test 5 used the same model, without freezing any layers but with flips and rotations as augmentation. This augmentation resulted in an improved performance (AUC 0.66). Tests 6-10 were the same but with the test 23 pretrained model, and again using augmentation led to a better

Table 5.19 Parameters tested - phase two optimisation

Parameter	**Values Tested**
Phase One Approach	Radiomic Features U-Net
Pretrained Model	None $test_{22}$-$test_{32}$
Freeze Layers	4 11 12 19 25
Model	$Phase2_1$-$Phase2_4$ ResNet-50
Cube Size (c)/mm	20 30 50 32 64
Node Range/mm	16 32
Undersampling Factor	4 8
Modality	PET/CT PET CT
Batch Size	8 16 32
Learning Rate	10^{-3} 10^{-4} 10^{-5}
Augmentation	Flips Rotations Zooming

performance. In tests 11-21 we used different models (both for the pretrained model and the layers succeeding the pretrained model). Test 16 was the best-performing of these, achieving an AUC of 0.72. In tests 22 to 25 we used this model and tested different learning rates and batch sizes. Test 23 gave the highest AUC (0.73), using a batch size of 16 and a learning rate of 10^{-3}. Finally, in tests 26 and 27 we tested input cubes of side length 20 mm and 50 mm, giving AUCs of 0.67 and 0.74 respectively. Combining these three models (and thus combining information at three different length scales) did not improve the performance (see Appendix D for full details). Finally, test 28 used the same CNN architecture as test 23 but without transfer learning. The AUC was comparable (0.73 vs 0.74), indicating that pretraining the model did not confer any advantage.

Table 5.21 shows the results when using cubes derived from the U-Net described in the previous section. The first 13 tests use the same model architecture as the best-performing in Table 5.20 (without transfer learning). They test different learning rates, batch sizes, and augmentations, all with a cube size of 32 mm. The results are fairly consistent, all between 0.74 and 0.84 AUC. Using flips and rotations improved the performance, but adding zooming augmentations did not. Using a batch size of eight resulted in slightly worse performances, so in subsequent tests we only used batch sizes of 16 and 32. Tests 42-47 used a ResNet-50 architecture with cubes of size 32 mm, again testing different learning rate and batch size combinations. We see the results are slightly better than when using the $Phase2_4$ model. With an AUC of 0.88, the best-performing model was test 42, which used a learning rate of 10^{-3} and a batch size of 16. Finally, tests 48-53 used a larger cube size of 64 mm, but still with a node range of 16 mm. Tests 50 and 52 both attained AUC scores of 0.88, both using a batch

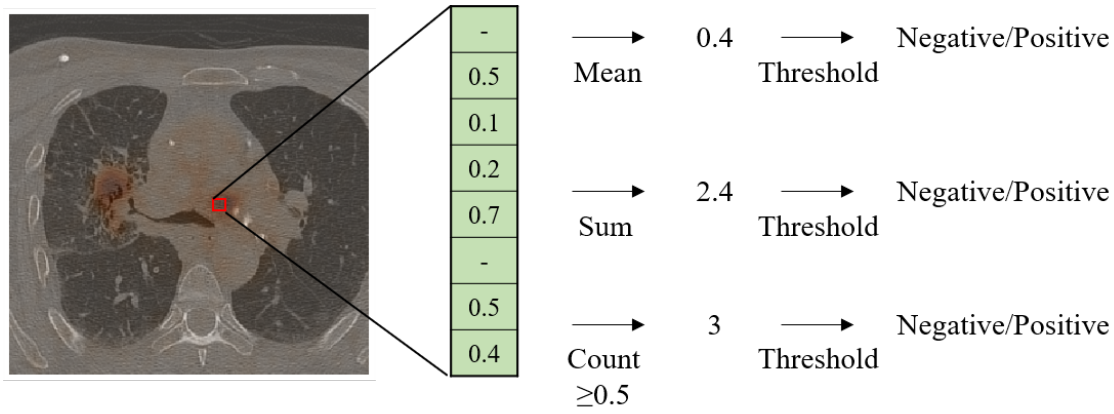


Fig. 5.20 Three thresholding methods tested to determine the pathological regions. Because the cubes overlap each coordinate has several classification predictions (in the green boxes). We tested aggregating these in three ways (mean, sum, and count)

size of 32. Test 53 used more negative cubes for training (undersampling by a factor of four instead of eight), but this did not improve the performance (AUC 0.79).

Several of these models discriminated between positive and negative cubes with a high accuracy. With several achieving AUC values of 0.88, we chose test 52 as our final model for evaluation. In the next section we use this model to give a node-level measure of the performance on the test set.

5.14 Classification Metric Choice - Method

From the classifications of each cube, classifications at coordinates within the scans could be inferred. To directly compare these with the accuracy of physicians they needed to be converted to an appropriate metric, as the accuracy of cube classification is not equivalent to the accuracy of node classification. Because the cubes overlapped there was more than one classification at each coordinate. We compared three aggregation metrics to find the most reliable at combining these classifications to give a single prediction. Details of the three metrics are given below and illustrated in Figure 5.20. These metrics were used to threshold the images. Any regions that had pathological nodes within them were classed as true positives (TPs), any without nodes were classed as false positives (FPs), and any pathological nodes that were not contained in a region were classed as false negatives (FNs). Using this measure one large region spanning the whole scan would constitute perfect accuracy despite being completely useless, so we visually checked that the regions were sufficiently small to be prognostically useful.

Table 5.20 Results - phase two optimisation (part 1). All experiments used cubes extracted using the phase one texture approach, the node range was half the cube size, and the modality was PET/CT. Experiments with an AUC greater than or equal to 0.70 are highlighted in blue

Test ID	Pretrained Model	Model	Freeze	lr	Batch Size	c/mm	Augmentation	AUC
1	test 22	$Phase2_1$	-	10^{-3}	16	30	None	0.54
2	test 22	$Phase2_1$	4	10^{-3}	16	30	None	0.52
3	test 22	$Phase2_1$	11	10^{-3}	16	30	None	0.55
4	test 22	$Phase2_1$	25	10^{-3}	16	30	None	0.50
5	test 22	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.66
6	test 23	$Phase2_1$	-	10^{-3}	16	30	None	0.52
7	test 23	$Phase2_1$	4	10^{-3}	16	30	None	0.50
8	test 23	$Phase2_1$	12	10^{-3}	16	30	None	0.50
9	test 23	$Phase2_1$	19	10^{-3}	16	30	None	0.50
10	test 23	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.67
11	test 24	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.59
12	test 25	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.65
13	test 26	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.53
14	test 27	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.67
15	test 28	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.58
16	test 29	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.72
17	test 30	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.62
18	test 31	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.59
19	test 32	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.65
20	test 29	$Phase2_2$	-	10^{-3}	16	30	Flip Rot	0.70
21	test 29	$Phase2_3$	-	10^{-3}	16	30	Flip Rot	0.62
22	test 29	$Phase2_1$	-	10^{-3}	8	30	Flip Rot	0.68
23	test 29	$Phase2_1$	-	10^{-3}	16	30	Flip Rot	0.73
24	test 29	$Phase2_1$	-	10^{-3}	32	30	Flip Rot	0.52
25	test 29	$Phase2_1$	-	10^{-2}	16	30	Flip Rot	0.64
26	test 29	$Phase2_1$	-	10^{-3}	16	20	Flip Rot	0.67
27	test 29	$Phase2_1$	-	10^{-3}	16	50	Flip Rot	0.74
28	None	$Phase2_4$	-	10^{-3}	16	30	Flip Rot	0.74

Table 5.21 Results - phase two optimisation (part 2). All experiments used cubes extracted using the U-Net approach and the modality was PET/CT. No transfer learning was used. Experiments with an AUC greater than or equal to 0.85 are highlighted in blue

Test ID	Model	lr	Batch Size	c/mm	Node Range/mm	Augmentation	Under-sampling	AUC
29	$Phase2_4$	10^{-3}	8	32	16	Flip Rot	8	0.79
30	$Phase2_4$	10^{-4}	8	32	16	Flip Rot	8	0.76
31	$Phase2_4$	10^{-5}	8	32	16	Flip Rot	8	0.75
32	$Phase2_4$	10^{-3}	16	32	16	Flip Rot	8	0.81
33	$Phase2_4$	10^{-4}	16	32	16	Flip Rot	8	0.80
34	$Phase2_4$	10^{-5}	16	32	16	Flip Rot	8	0.77
35	$Phase2_4$	10^{-3}	32	32	16	Flip Rot	8	0.81
36	$Phase2_4$	10^{-4}	32	32	16	Flip Rot	8	0.84
37	$Phase2_4$	10^{-5}	32	32	16	Flip Rot	8	0.76
38	$Phase2_4$	10^{-4}	16	32	16	None	8	0.74
39	$Phase2_4$	10^{-4}	32	32	16	None	8	0.78
40	$Phase2_4$	10^{-4}	16	32	16	Flip Rot Zoom	8	0.80
41	$Phase2_4$	10^{-4}	32	32	16	Flip Rot Zoom	8	0.82
42	ResNet-50	10^{-3}	16	32	16	Flip Rot	8	0.88
43	ResNet-50	10^{-4}	16	32	16	Flip Rot	8	0.86
44	ResNet-50	10^{-5}	16	32	16	Flip Rot	8	0.84
45	ResNet-50	10^{-3}	32	32	16	Flip Rot	8	0.71
46	ResNet-50	10^{-4}	32	32	16	Flip Rot	8	0.87
47	ResNet-50	10^{-5}	32	32	16	Flip Rot	8	0.83
48	ResNet-50	10^{-3}	16	64	16	Flip Rot	8	0.83
49	ResNet-50	10^{-4}	16	64	16	Flip Rot	8	0.83
50	ResNet-50	10^{-3}	32	64	16	Flip Rot	8	0.88
51	ResNet-50	10^{-4}	32	64	16	Flip Rot	8	-
52	ResNet-50	10^{-5}	32	64	16	Flip Rot	8	0.88
53	ResNet-50	10^{-3}	16	64	16	Flip Rot	4	0.79

Mean The first metric we tested was the mean of the classification probabilities at each coordinate. These means were then thresholded to give the pathological regions. We tested thresholds ranging from 0.3 to 0.8.

Sum The second metric tested was a sum of all the classification probabilities at each coordinate. This differs from the mean because not all coordinates are covered by the same number of cubes, some regions having been eliminated in phase one. Thresholds from 18 to 26 were tested.

Count The final metric counted the number of *true* predictions at each coordinate i.e. the number of predictions above 0.5. We thresholded the count to give the positive regions, using count thresholds ranging from 12 to 27.

After choosing the metric and threshold, we evaluated the final model on our test set. During training our *ground truth* was the labelling by an experienced physician, but as mentioned this labelling is not perfect (the sensitivity and specificity of lesion-based analyses by physicians have been shown to be 0.59 and 0.97 respectively [1]). Therefore, a more informative metric is a comparison of the variation of our algorithm with respect to a physician to the variation of one physician with respect to another. The test set was hence independently labelled by a second physician (5 years' experience in thoracic imaging). Inter-observer agreement was measured using the kappa statistic (κ). For all comparisons 95% Clopper-Pearson confidence intervals were used.

5.15 Classification Metric Choice - Results and Discussion

The results using the mean, sum, and count thresholds are shown in Tables 5.22 to 5.23. Several sum and count-based thresholds performed well. A sum threshold of 18 achieved a high performance and importantly only missed five pathological nodes, so we used this for final evaluation on the test set.

Evaluating on the test set gave 45 TPs, 7 FNs, and 12 FPs, resulting in a sensitivity of 0.87 [0.74,0.94] with 0.40 [0.21,0.68] FPs/patient. This corresponds to a substantial agreement ($\kappa = 0.77$ [0.68,0.87]) between our model and the physician's labels. The agreement between the two physicians was also substantial ($\kappa = 0.66[0.54, 0.77]$). These results show that the agreement between our model and the physician was comparable to that between the two physicians.

Table 5.22 TP, FN, and FP nodes when using different mean-based thresholds on the validation set

Threshold	TP	FN	FP
0.3	34	1	57
0.4	29	6	58
0.5	25	10	42
0.6	23	12	34
0.7	14	21	28
0.8	13	22	26

Table 5.23 TP, FN, and FP nodes when using different sum-based thresholds on the validation set

Threshold	TP	FN	FP
18	30	5	3
20	28	7	3
22	21	14	3
24	21	14	3
26	19	16	2

Table 5.24 TP, FN, and FP nodes when using different count-based thresholds on the validation set

Threshold	TP	FN	FP
12	29	6	5
15	24	11	3
18	23	12	3
21	22	13	3
24	21	14	3
27	19	16	2

5.16 Discussion

Although not explicitly calculated, in theory the number of true negatives (TNs) could be determined by subtracting the sum of TPs, FPs, and FNs from the total number of mediastinal lymph nodes per patient. However, looking at previous studies (which use biopsies to measure the true accuracy of physicians) there is no consistency in the number of nodes per patient (in the meta-analysis [1] the number of nodes per patient varies from 0.8 to 12.7). This is probably because some nodes are very hard to biopsy due to location, and consequently there is no consistency between the studies. The sensitivity and FPs/patient are therefore better measures of performance. In spite of this, because many previous studies use specificity as a benchmark, we felt compelled to estimate it for our model. The most comparable study to ours, which uses a CNN to classify mediastinal lymph nodes as positive or negative, had an average of 8.3 nodes per patient [175]. We used the same number of nodes per patient and thus determined the number of TNs on this assumption. This gave a specificity of 0.94 [0.90,0.97] on the test set.

The results show that our model identifies pathological mediastinal nodes at an accuracy comparable to that of physicians. A meta-analysis of physician performance showed sensitivity and specificity values of 0.59 and 0.97 respectively for lesion-based labelling [1]. Our model performs favourably compared to these results. However, this meta-study showed a huge range in these values (sensitivity from 0.46 to 0.90 and specificity from 0.65 to 0.98). One factor that was shown to affect the results was geography (some studies were in high tuberculosis-prevalent regions, which lead to more false positives), but other factors such as scanner type, centre protocols, and physician experience may have made a difference. This variation highlights the difficulty in comparing results across studies, so any comparison must be treated with care. Taking this into account, our results are an improvement on previous automated mediastinal detection studies, with the CNN used in [175] achieving sensitivity and specificity scores of 0.84 and 0.88 respectively. This is despite our study having as input the full FDG-PET/CT scans (compared to patches centred on the lymph nodes) and represents a significant improvement in the overall automation of the process.

Comparing our results to those between two physicians, there was a similar agreement between our algorithm and the first physician to that between the two physicians (κ=0.77 [0.68,0.87] vs κ=0.66 [0.54,0.77]). We could only find one study reporting inter-physician agreement in identifying pathological mediastinal nodes using PET/CT scans. In that, inter-observer agreement was shown to range from 0.48-0.88, depending on the type of mediastinal node [161]. These results are comparable to those of our study. As mentioned however, comparisons to other studies have to be treated with caution due to the potentially large variations.

One limitation of this study, and thus problem with the above comparisons, was the lack of comparison with a gold standard. We did not have access to direct histological comparison, as in standard clinical practice only suspicious nodes are biopsied and if a patient has several suspicious nodes not all will be biopsied. Additionally, some nodal stations are easier to access and thus more often biopsied, leading to sampling errors. These problems may explain the huge range in the number of nodes per patient seen in the meta-analysis [1]. Thus, comparison to an experienced physician may be the best benchmark to use.

Our dataset was quite small, and more data would undoubtedly improve these results. In a broader sense, this is the key advantage of an automated approach. As clinical practice adapts and datasets get bigger, a CNN can train on a broader range of nodes, allowing it to become more accurate. It could be that a certain misclassified node had no direct comparison in the training set, but with a larger training set this would rarely happen. With more data, further studies could look to identify not only mediastinal nodes but also the primary tumour and any hilar nodes, outputting a fully labelled scan with pathological node stations. This would allow a quick automated TNM-staging classification for patients. Additionally, the data used in this study are from a single scanner and centre. Inter-scanner differences are known to result in reduced performance when using machine learning models [206], and this is an important barrier to overcome to achieve large-scale deployment of these models. In Chapter 8 we address this problem by extending this work to a new dataset from a different scanner.

5.17 Conclusions and Afterthoughts

Our results show a fully automated 3D CNN-based algorithm that can detect pathological mediastinal nodes with a performance similar to that of an experienced physician. To our knowledge it is the first study of its kind to go directly from whole-body FDG-PET/CT scans to pathological mediastinal lymph node predictions and locations. This method could therefore be deployed in a clinical setting to aid physicians and save time. In the future a more advanced dataset (with more data and gold standard labels) could lead to better performance and the ability to directly compare with the ground truth.

As the first major problem tackled using deep learning, this work can also be seen as a learning curve. We tested several different approaches to the problem and found that trying to solve the problem in one step did not work. The MIP and 3D cube approaches were not successful, probably because the input scans were huge compared to the size of the mediastinal nodes. It could be that with a huge training dataset whole-body inputs could be used, but there is far too much variation in individual scans for this to work in our case.

This showed the importance of thinking carefully about the task one is trying to solve and tailoring the solution accordingly. Another demonstration of this was the switch from a radiomic feature approach to a U-Net approach in phase one. Using whole 2D slices meant that the CNN was able to accurately select only mediastinal regions, whereas when using the radiomic feature approach (which used individual cubes) often areas of high uptake in the abdomen were selected.

We also explored different transfer learning methods in 2D and 3D. For both cases, using an Inceptionv3 network pretrained on ImageNet and pretraining a 3D CNN using the jigsaw task, we did not find any benefit to transfer learning. This is of course not to say that these methods are always useless (many studies have demonstrated the value of these techniques [128] [129]), but shows that their benefits are task-dependent. This was also the case when combining cube sizes. If they do not bring significant benefit these methods should be avoided in favour of simplicity.

In each stage of the experimentation above we thoroughly tested numerous parameters relating to the training of the neural networks. To a certain degree this has helped develop our understanding of CNN training. In 5.10 we found that limiting the voxel value ranges improved the results as did using $[\mu, \sigma]$ rescaling rather than [0,1]. These results indicate that the range and distribution of values input into CNNs significantly affects their performance. Other preprocessing parameters (e.g. sphere radii, number of negative slices) dramatically affected the results, showing the importance of choosing the correct preprocessing parameters. In general the actual model architecture seemed less important (compare the U-Net and U-Net$_{\text{small}}$ results in 5.10 and different model performances in 5.6.2 and 5.12). We also learnt later into the work the strength of using prebuilt networks (e.g. ResNet). The ResNet-50 model outperformed all the models we had custom built and clearly we could have saved time by starting with this or other prebuilt and widely tested models.

Despite these insights in many cases the performance was very unpredictable, as clearly demonstrated in Table 5.11. The performance is complexly linked to the different parameters, meaning these parameters cannot be independently optimised. Given the number of parameters and time required to train networks, it is not possible to test all combinations. Compromises need to be made based on prior knowledge and results. Much work has been done studying the response of networks to parameters changes [76] [77] and discussing the best way to search this parameter space [80], but there seems no easy solution. So much of the work is empirical, with no clear theoretical explanation. It also seems problem-specific, meaning the knowledge and assumptions gained from this problem may be completely irrelevant for the next.

Finally, there are many questions we did not try to answer. In later chapters we test our model on data from a different scanner and try to interpret the model. These are crucial questions that plague current deep learning research and are particularly pressing for medical-based problems. In the following chapters we attempt to answer some of the questions that this work awoke in us.

Chapter 6

Distinguishing Prosthetic Valve Endocarditis from Non-Specific Inflammation

6.1 Problem Background

Over 300,000 prosthetic heart valves (PHVs) are implanted worldwide each year [207]. Prosthetic valve endocarditis (PVE) is a serious complication of this surgery with an incidence rate of between 0.3% and 1.2%. Despite improvements over time mortality rates for PVE remain high at 22.8% [2].

Diagnosis of PVE is challenging, requiring a multidisciplinary approach including cardiologists, cardiac imaging specialists, heart surgeons, infectious disease physicians, and microbiologists [208]. Different imaging techniques, such as transoesophageal echocardiography (sensitivity 0.86 and specificity 0.88) and transthoracic echocardiography (sensitivity 0.57 and specificity 0.63), are commonly used alongside blood culture tests [209] [208]. Recently, FDG-PET/CT scans have been proposed as a supplementary diagnostic method. In a review, the diagnostic sensitivity and specificity values of PET/CT scans were reported as 0.73 to 0.97 and 0.80 to 0.94 respectively [210], and the procedure has been added to the European Society of Cardiology guidelines for the diagnosis and management of infective endocarditis [207]. Qualitative assessment is made by identifying regions of high PET uptake around the valve and comparing the attenuation-corrected and non-attenuation-corrected images. Studies have also attempted to establish SUV-based cutoffs for quantitative analysis [207]. However, physiological inflammation due to recent surgery (non-specific inflammation (NSI)) also results in high PET uptake and can lead to false positive results (see Figure

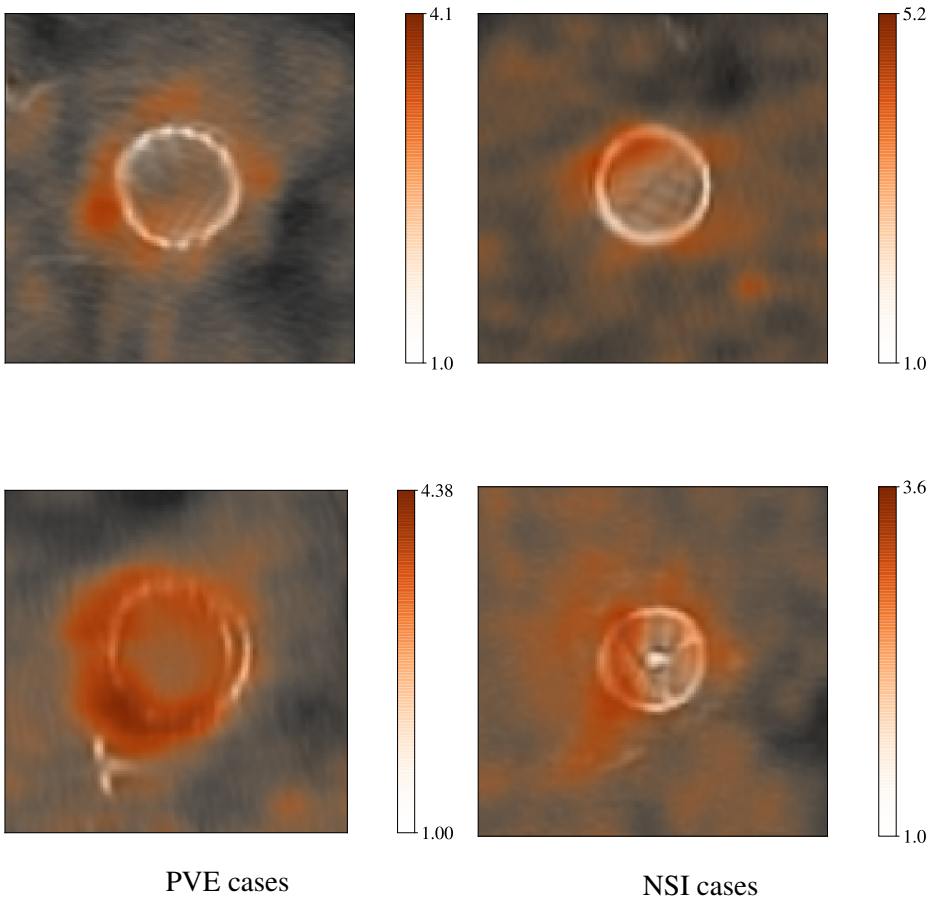


Fig. 6.1 A comparison of FDG-PET/CT scans of valves for patients with PVE and NSI. High PET uptake is in orange. The maximum value for each image is shown on the colour bar

6.1 for some example PET/CT images of PVE and NSI). One proposed workaround is to wait three months before performing the scan [207], but in some cases false positives are possible beyond three months after surgery [211]. Further work is therefore needed to better explain the characteristics of PVE and NSI and determine how they can be differentiated. In this chapter we attempt to use a CNN to differentiate between PVE and NSI cases using FDG-PET/CT images.

6.2 Previous Work

As mentioned, qualitative and quantitative methods have been proposed as means to identify PVE. [212] attempted to diagnose PVE and infection of cardiac implantable electronic devices (CIEDs) for a cohort of 41 patients. They achieved a sensitivity of 1.00 and specificity of 0.73 when combining attenuated and non-attenuated visual analysis by an expert. Using an

Table 6.1 Summary of data used split into training and validation sets

Cohort	**PVE**	**NSI**	**Total**
Total Patients	28	89	117
Excluded Patients	2	18	20
Training Set	17	48	65
Validation Set	9	23	32

SUV_{max} cutoff of 5.5 they achieved a sensitivity of 0.50 and a specificity of 0.80. Similarly, [213] found that an SUV_{max} cutoff of 3.7 identified PVE or CIED with 0.91 sensitivity and 0.79 specificity (AUC 0.89 with 95% CIs from 0.82 to 0.96) for a cohort of 92 patients. They also found that an SUV_{ratio} (calculated by dividing the SUV_{max} by the aortic blood mean SUV) cutoff of 1.69 achieved 0.91 sensitivity and 0.76 specificity (AUC 0.89 with 95% CIs from 0.82 to 0.96). [207] reviewed several studies using SUV_{max} and SUV_{ratio} cutoffs. SUV_{max} cutoffs varied from 3.3 to 6.2 and SUV_{ratio} cutoffs varied from 1.6 to 4.4. This range in cutoff values shows a lack of generalisability, as these values will vary depending on differences in cameras, imaging protocols, and reconstruction methods between centres. Finally, for a cohort of 142 patients, [214] found SUV_{max} and SUV_{ratio} were not capable of distinguishing PVE and NSI cases. They defined the *dispersion index* as the product of the number of foci on the PET scan and the relative volume of positive uptake (the volume of PET uptake divided by the volume of the valvular region). This was shown to differentiate PVE and NSI with a sensitivity and specificity of 0.97 and 0.95 respectively.

6.3 Dataset

Our dataset consisted of 117 FDG-PET/CT scans of the chest, of which 28 had been diagnosed as PVE and 89 as NSI (using followup). Images were acquired on a PET/CT (Discovery 690, General Electric Medical Systems, Buc, France) 60 minutes after injection of an FDG radiotracer and were reconstructed using the OSEM algorithm. Images were segmented semi-automatically with LIFEx using the CT image. A rough bounding box was first drawn around the valve, then a threshold of 40% of the maximum HU was used to remove the valve. The blood signal inside the valve was then segmented and manually removed by a physician.

20 cases were removed due to problems aligning the segmentation and the valve. The dataset was then split into a training and a validation set, ensuring each had a similar ratio of PVE to NSI cases. We did not use a separate test set due to the limited dataset size. Details of the dataset are shown in Table 6.1.

Table 6.2 Parameters tested. ResNet-X indicates a ResNet network with X layers

Parameter	**Values Tested**
Model	Cardiac_1 Cardiac_2 ResNet-18 ResNet-34 ResNet-50
Cube Side Length/mm	32 36 42 46
Batch Size	4 8
Learning Rate	10^{-3} 10^{-4}

6.4 Method

6.4.1 Preprocessing

The radiologist had reorientated all the scans so the ring of the prosthetic valve lay in the *xy*-plane. This meant our images were more homogeneous, which we thought would make training the CNN easier. The reorientation meant that axes had non-uniform resolutions, so we first used a cubic spline interpolation to resample all the images so the PET and CT had uniform voxel sizes of 0.5x0.5x0.5 mm^3. We found the central coordinate of each affected region using the geometrical mean of the segmentation then cut cubes around this point. Using the training set as a standard, we rescaled the values of the cubes so they lay between zero and one.

6.4.2 Deep Learning Approach

The data were used to train various CNN architectures, the full details of which can be found in Appendix C.5. Given the input's similarity to the second phase of the mediastinal problem (both are 3D PET/CT cubes) we tested similar models, including some ResNet models and some custom-made models. The mean length and diameter of the segmented regions were 36 mm and 40 mm respectively, but we did not know if more distant surrounding regions would be useful for classification. We therefore tested cubes of side length 32 mm, 36 mm, 42 mm, and 46 mm. We used random flips and translations $\in [-10, 10]$ mm in all three axes and rotations in the *xy*-plane as in-training augmentation. To address the class imbalance (17 PVE vs 48 NSI in the training set), the PVE class was oversampled by a factor of three. Batch sizes of four and eight and learning rates of 10^{-3} and 10^{-4} were tested and an Adam optimiser was used in all cases. A full list of parameters tested can be found in Table 6.2.

With a small dataset evaluation metrics can be very volatile to changes in the model (for example, a change in the classification of one patient would shift the sensitivity by 0.08). To make the classification more stable we used test-time augmentation, which has been shown to improve the robustness of testing accuracy [215] [216] [217]. The validation

set was augmented by a factor of ten, in each case transforming the data using the same random transformations as above. The mean of the ten results was taken to give the patient classification. We evaluated the results using the sensitivity, specificity, and AUC. For the sensitivity and specificity a threshold of 0.5 was used (i.e. if the aggregated result was greater than 0.5 the patient was classed as PVE). We used 95% Clopper-Pearson CIs to measure the uncertainty in the outcome.

6.4.3 Radiomic Approach

This work was done in collaboration with a colleague who used radiomic features to evaluate the dataset (see [218]). In this comparative work they extracted 40 PET-based radiomic features from the segmented regions (SUV_{max}, SUV_{mean}, one shape-based feature, five first-order features, and 32 texture features), using fixed-width bins of 0.15 SUV. Using these they performed univariate analysis and created bivariate linear regression models. For each method the experiments were repeated 50 times, each time randomly assigning 75% of patients to the training set and 25% to the validation set. The mean of these 50 experiments on the validation sets gave the final performance. When creating bivariate linear regression models, only the top ten features according to the univariate analysis (sorted by Youdon index) were considered. Sensitivity and specificity scores were found by maximising the Youdon index.

6.5 Results

The results using the deep learning approach are shown in Table 6.3. Tests 1-8 use ResNet-50 and ResNet-34 models with learning rates of 10^{-3} and 10^{-4} and input cubes of side length 46 mm and 42 mm. Tests 2, 5, and 8 all achieved AUC scores greater than 0.70 (given the small validation set size the differences are undoubtedly statistically insignificant). Because the highest scores (tests 5 and 8, both with AUC 0.75) both used a learning rate of 10^{-4}, we fixed this for future tests. As in the previous chapter, there are too many hyperparameter permutations to feasibly test them all. By only using a learning rate of 10^{-4} in subsequent tests we are not necessarily assuming that any tests with a learning rate of 10^{-3} will be worse, but rather saying that they will not give a significantly better performance.

Tests 9-13 test the same models with a batch size of four, resulting in a slightly better performance for test 10 (AUC of 0.79). In subsequent tests we therefore used a batch size of four. In the remaining tests (14-28) we tested more models (Resnet-18, $Cardiac_1$, and $Cardiac_2$) with a wider range of input cube sizes (with 36 mm and 32 mm side lengths). The

Table 6.3 Deep learning approach results. The best-performing test (test 10) is highlighted

Test ID	Model	Cube side length/mm	Batch Size	lr	AUC
1	Resnet-50	46	8	10^{-3}	0.68
2	Resnet-34	46	8	10^{-3}	0.71
3	Resnet-50	42	8	10^{-3}	0.35
4	Resnet-34	42	8	10^{-3}	0.57
5	Resnet-50	46	8	10^{-4}	0.75
6	Resnet-34	46	8	10^{-4}	0.67
7	Resnet-50	42	8	10^{-4}	0.54
8	Resnet-34	42	8	10^{-4}	0.75
9	Resnet-50	46	4	10^{-4}	0.58
10	Resnet-34	46	4	10^{-4}	0.79
11	Resnet-50	42	4	10^{-4}	0.64
12	Resnet-34	42	4	10^{-4}	0.63
13	Resnet-18	46	4	10^{-4}	0.78
14	Cardiac_1	46	4	10^{-4}	0.45
15	Cardiac_2	46	4	10^{-4}	0.68
16	Resnet-18	42	4	10^{-4}	0.78
17	Cardiac_1	42	4	10^{-4}	0.50
18	Cardiac_2	42	4	10^{-4}	0.43
19	Resnet-50	36	4	10^{-4}	0.55
20	Resnet-34	36	4	10^{-4}	0.75
21	Resnet-18	36	4	10^{-4}	0.61
22	Cardiac_1	36	4	10^{-4}	0.50
23	Cardiac_2	36	4	10^{-4}	0.52
24	Resnet-50	32	4	10^{-4}	0.75
25	Resnet-34	32	4	10^{-4}	0.72
26	Resnet-18	32	4	10^{-4}	0.72
27	Cardiac_1	32	4	10^{-4}	0.46
28	Cardiac_2	32	4	10^{-4}	0.71

best-performing model remained test 10 with an AUC of 0.79. This used a 34-layer ResNet model with a cube side length of 46mm, batch size of four, and learning rate of 10^{-4}.

95% CIs for this test gave an AUC score of 0.79 [0.61,0.94]. The sensitivity and specificity were 1 [0.66,1] and 0.39 [0.20,0.61] respectively.

Figures 6.2 and 6.3 show tables of results from the comparative radiomic study. The best univariate result used SUV_{max} (AUC 0.75 ± 0.06). Including a second feature did not significantly improve this result (several bivariate models achieved AUCs of 0.80 ± 0.06).

Paramètre	Corrélation avec SUVmax	AUC TRS	Y TRS	Se TRS	Sp TRS	AUC TES	Y TES	Se TES	Sp TES
SUVmax	1	0.82±0.03	0.56±0.05	0.88±0.05	0.68±0.05	0.75±0.06	0.49±0.13	0.8±0.13	0.69±0.08
GLZLM_HGZE	0.93	0.79±0.03	0.55±0.05	0.83±0.05	0.73±0.04	0.74±0.06	0.47±0.11	0.76±0.1	0.71±0.09
GLCM_Homogeneity	-0.77	0.75±0.03	0.51±0.06	0.71±0.07	0.81±0.07	0.72±0.07	0.44±0.14	0.65±0.13	0.78±0.12
HISTO_Energy	-0.87	0.79±0.03	0.5±0.06	0.67±0.07	0.83±0.07	0.7±0.07	0.4±0.14	0.6±0.16	0.8±0.11
HISTO_Entropy	0.90	0.8±0.03	0.5±0.05	0.67±0.11	0.83±0.11	0.69±0.07	0.37±0.13	0.56±0.15	0.81±0.14
GLZLM_LGZE	-0.81	0.75±0.03	0.5±0.06	0.78±0.05	0.71±0.04	0.71±0.07	0.42±0.13	0.75±0.1	0.68±0.09
GLRLM_HGRE	0.80	0.74±0.04	0.49±0.06	0.78±0.06	0.71±0.05	0.71±0.07	0.41±0.14	0.72±0.14	0.7±0.09
GLCM_Dissimilarity	0.80	0.75±0.03	0.49±0.06	0.74±0.08	0.75±0.08	0.69±0.07	0.39±0.13	0.64±0.15	0.75±0.13
GLRLM_SRHGE	0.83	0.74±0.04	0.49±0.06	0.78±0.05	0.7±0.05	0.7±0.07	0.4±0.13	0.72±0.13	0.68±0.09
GLCM_Energy	-0.83	0.78±0.03	0.48±0.06	0.72±0.09	0.77±0.1	0.68±0.06	0.36±0.14	0.62±0.18	0.74±0.14

Fig. 6.2 Radiomic results using univariate analysis for ten features. TRS indicates the training set and TES the test set. Histo indicates first-order features. Y=Youdon Index, Se=Sensitivity, and Sp=Specificity. Taken from [218]

Paramètre	AUC TRS	Y TRS	Se TRS	Sp TRS	AUC TES	Y TES	Se TES	Sp TES
SUVmax+ GLCM_Dissimilarity	0.81±0.03	0.46±0.11	0.7±0.06	0.77±0.13	0.8±0.06	0.46±0.14	0.69±0.1	0.77±0.15
GLZLM_HGZE+ GLRLM_SRHGE	0.8±0.05	0.46±0.1	0.71±0.06	0.75±0.1	0.79±0.06	0.43±0.15	0.71±0.1	0.72±0.17
GLZLM_HGZE+ GLZLM_LGZE	0.79±0.03	0.46±0.09	0.7±0.06	0.76±0.08	0.79±0.06	0.47±0.12	0.72±0.09	0.76±0.13
SUVmax+ GLZLM_HGZE	0.79±0.05	0.45±0.1	0.69±0.07	0.76±0.11	0.8±0.06	0.44±0.16	0.72±0.09	0.72±0.17
GLZLM_HGZE+ GLRLM_HGRE	0.79±0.05	0.44±0.09	0.72±0.06	0.71±0.1	0.78±0.07	0.41±0.13	0.73±0.1	0.67±0.14
GLZLM_HGZE+ HISTO_Energy	0.79±0.03	0.43±0.09	0.72±0.07	0.71±0.1	0.79±0.06	0.45±0.15	0.72±0.1	0.74±0.15
SUVmax+ GLCM_Homogeneity	0.8±0.04	0.43±0.1	0.69±0.07	0.75±0.11	0.8±0.07	0.43±0.19	0.7±0.1	0.73±0.19
SUVmax+ GLRLM_HGRE	0.79±0.05	0.43±0.09	0.69±0.06	0.74±0.1	0.8±0.06	0.42±0.14	0.72±0.1	0.7±0.16
SUVmax GLRLM_SRHGE	0.8±0.05	0.43±0.1	0.68±0.06	0.75±0.1	0.8±0.06	0.43±0.15	0.71±0.11	0.72±0.18
SUVmax+ GLZLM_LGZE	0.8±0.03	0.43±0.09	0.68±0.06	0.74±0.1	0.8±0.06	0.44±0.15	0.71±0.11	0.73±0.18

Fig. 6.3 Radiomic results using a bivariate linear regression model for ten feature combinations. TRS indicates the training set and TES the test set. Histo indicates first-order features. Y=Youdon Index, Se=Sensitivity, and Sp=Specificity. Taken from [218]

6.6 Discussion

Both deep learning and radiomic approaches showed comparable results, very similar to only using SUV_{max}. Our results are slightly worse than [213], which used an SUV_{max} cutoff to achieve an AUC of 0.89 [0.82,0.96]. As explained in 6.2 different studies propose different SUV_{max} cutoffs, with one study finding SUV_{max} not capable of distinguishing PVE from NSI at all. This shows that inter-cohort differences are significant, probably explaining the difference in performance between our results and other SUV_{max} based studies. [214] used the number of foci on the PET scans to give sensitivity and specificity scores of 0.97 and 0.95 respectively, better than our results. One would think that our CNN would have been able to automatically detect foci and therefore use this information, but perhaps we had too few data. This study had more patients than ours (142 vs 97) and crucially more PVE cases (70 vs 26).

It is disappointing that our models did not outperform a simple SUV_{max} threshold, especially given other studies have shown that there is potential for better performance. In future work we would like to test this foci-based method on our data, however this is not from a full paper so methodical details are scarce. Our study was not comprehensive, with many parameters such as CT and PET thresholds and different normalisation methods not tested. However, with only nine PVE patients in the validation set it was hard to draw solid conclusions and confidently differentiate between different models' performances. More data would help to improve the accuracy and the reliability of the results, allowing for further investigations. It would also allow for an independent test set and a more representative comparison between our method and previous work on the problem.

6.7 Conclusions and Afterthoughts

Our model achieved reasonable results differentiating PVE from NSI, but disappointingly they were no better than just using SUV_{max}. More data would likely improve the performance and allow for more robust comparisons between methods. Our CNN is much more complex and powerful than a univariate model, but with only 65 patients to train with this power could not be leveraged. This motivates the questions: Do we need the complexity of a CNN when using a small dataset? Would a simpler model, using a few radiomic features, give a similar performance? In the next chapter we compare radiomic and CNN-based models in depth to try to answer these questions.

This work highlighted the problems encountered when comparing results to those from other studies. Different publications present very different results even when using a simple metric (SUV_{max}). Similar problems presented themselves in the previous chapter, where we

found that the classification accuracy using SUV_{max} and the number of nodes per patient varied considerably between studies. These differences make it impossible to compare different methods, meaning we cannot determine which method is better or worse. It also dooms our models to be applicable only to small cohorts, usually from a single centre. More collaboration and sharing of datasets is necessary to transition from small research-level studies to larger-scale studies that can advance clinical practice.

Chapter 7

Comparison of Radiomic and Deep Features

7.1 Problem Background and Current Work

As described in Chapters 2 and 4, radiomics and CNNs are two ML approaches that have shown great promise in automating medical imaging analysis. The use of both has exploded over the past few years, and both have been shown to be effective at solving a wide range of medical imaging analysis problems. A comparison of the relative merits of both methods would therefore be of great use. In this chapter we compare three aspects of these methods; their *performance*, *interpretability*, and *stability*. For each we focus on tests when the dataset size is small (relevant to medical imaging) and investigate how the dataset size affects the findings.

Performance In the domain of computer vision, CNNs have long dominated image-based problems, with variants of CNNs winning the ILSVRC for the last several years [219]. However, these tasks are very different to those in the medical imaging domain. Firstly, the data themselves are very different; the 3D, irregular shapes of tumours, often with blurred boundaries, do not resemble the 2D object-based images usually used to benchmark computer vision models. Secondly, the sizes of the datasets are completely different. In 2018 the average dataset size for papers submitted to MICCAI was 250 for MRI studies and 500 for CT studies, compared to the millions of images in the ImageNet database [118]. This paucity of data places restrictions on the size and thus complexity of networks, and this limits the precision with which a network can learn features. Radiomic features, being mathematically predetermined (*handcrafted*), are independent of the dataset size. Consequently, at larger dataset sizes the adaptability of CNN-based (*deep*) features should

result in better performance. This is because with increasing data a CNN can improve and optimise its learnt features, whereas the handcrafted features stay the same. A few studies have used both radiomic and deep features and compared the two [220] [175] [221], with some combining the two types of features to give an improved performance [191] [222]. This implies that the different approaches are extracting different complementary information from the data. In Chapter 6 we compared radiomic and CNN models and found that they both gave comparable accuracies. However, these studies all use a fixed dataset size, so do not demonstrate how the performances of the two types of feature evolve as the number of data changes. Previous work has shown that CNN learning curves follow predictable power-law forms [223] [224] [225]. Using radiomic-based features, one study found a logarithmic relationship (mathematically similar to a power-law relationship) between the number of data and the performance [226]. In this study we investigate how the performance of models built using these features evolves as dataset size increases, focusing on the small-data region relevant to medical imaging problems. For three datasets we examine if and when models based on deep features start to outperform their radiomic-based counterparts, whether they always follow a predictable power-law relationship, and if there is an advantage in combining the two types of features. We also compare individual predictions between the models, to see to what extent the models agree.

Interpretability In the second part of this chapter we attempt to explain and interpret the models. A common question (and often criticism) of CNNs regards their interpretability. This is an especially pertinent question in the medical imaging field, where clinicians do not just want a binary answer, they want an explanation. The European Union's recent *General Data Protection Regulation* demands that automated individual decision-making which significantly affects users has to be explainable, further accentuating the need for more explainable models [227]. Increased interpretability would help to increase trust in automated models and could help us to understand underlying biological mechanisms. As discussed in 4.4 this is an active field of research across computer vision, and there are numerous proposed methods for interpreting CNNs. Our aim in this study is not to give a comprehensive review and analysis of these methods. Instead, we use two popular methods, Grad-CAM and occlusion sensitivity, to see how much we can practically understand about CNNs' decisions for three different medical problems.

Problems of interpretability are not limited to CNN-based studies. Although simpler radiomic features such as volume, skewness, and maximum intensity may be easy to interpret, many higher-order features are much less clear, with some arguing that radiomic models are no more interpretable than CNN models [43]. To test this, for our three datasets we find

which radiomic features are being used and whether we can use them to explain the decisions being made.

We also explore a new method of interpreting CNNs, attempting to recreate the deep features using their radiomic counterparts. This allows us to relate the two feature types and determine whether a correspondence can be established between deep features and radiomic features, i.e. determine whether deep features can be expressed as a combination of radiomic features.

Stability Next, we test the stability of deep features to model retraining. To be able to trust a model in a clinical setting it needs to be predictable. However, many studies have shown that small perturbations (*adverserial attacks*) in input data (i.e. adding noise) can affect the output of CNN-based classifiers [228] [229] [230]. This is particularly worrying in the medical imaging domain, where different protocol parameters, patient movement, and reconstruction methods can result in small distortions and noise, and has led to some questioning the scientific method and rigour of ML studies [231] [232] [233]. [234] showed that CNNs retrained after shuffling the training data (while keeping all other parameters the same) can give different outputs, despite overall AUC performances being very similar (finding an average coefficient of variation of 0.54 for an individual patient on retrained CNNs). This suggests that retrained CNNs are converging to different local minima and is concerning, as we do not want retrained models to give different outcomes for the same patient. In this section we use the above methods to investigate this problem. We test whether CNNs retrained with the same parameters give the same image-level results then use grad-CAM and occlusion maps to see if these networks are focusing on the same physical features in the images. Again, with a focus on the small-data region particular to medical imaging, we examine how the amount of data affects these results.

Effect of Segmentation Finally, we compare the performance of models with and without prior tumour segmentation. Segmentation is a key preprocessing step for both CNN and radiomic models (particularly for radiomic models, where it is almost always used [25] [26] [38]). However, the segmentation process itself is ambiguous and can lead to disagreements between different experts or segmentation-based algorithms [110]. These differences affect further analysis and decisions. In this section we evaluate the importance of precise segmentation, comparing the performance of models trained with precise segmentations to those trained with rough bounding boxes.

Table 7.1 Summary of the three datasets used in this study

Dataset	Modality	Dimensionality	Class	Data/Class
Brain	MRI	2D	Meningioma	708
			Glioma	1426
			Pituitary Tumour	930
LUNA	CT	3D	Positive	1166
			Negative	1166
Mediastinal	PET/CT	3D	Positive	6476
			Negative	6513

7.2 Datasets

Because medical imaging tasks can vary so widely, to increase the generalisability of results we ran the experiments on three datasets. We chose 2D and 3D datasets including PET, CT, and MRI data. Table 7.1 summarises the datasets used.

7.2.1 Brain Dataset

This dataset consists of 3064 T1-weighted contrast-enhanced MRI images with three types of brain tumours (meningiomas, gliomas, and pituitary tumours). The data were acquired from Nanfang Hospital, Guangzhou, China, and General Hospital, Tianjing Medical University, China, from 2005 to 2010. The dataset was originally used in [235] to automatically identify the tumour type and after made publicly available at [236]. It contains 2D images collected from MRI scans from 233 cancer patients. The tumours have been manually delineated by three experienced radiologists. Some example images are shown in Figure 7.1. In total there are 82 patients with meningiomas (708 slices), 89 with gliomas (1426 slices), and 62 with pituitary tumours (930 slices).

Several studies have been published using radiomic and CNN methods to classify brain tumours. [237] [238] and [37] all use primarily GLCM and wavelet-based features to classify brain tumours on similar datasets. On the dataset we are using, intensity, GLCM, and Bag-of-Words (BoW) features have been used to classify the images with a BoW-based model attaining a maximum accuracy of 0.88 using the segmented tumour region[1] [235]. Numerous CNN-based studies have also been benchmarked on this dataset, all achieving accuracies over 0.90 [239] [240] [241] [121]. As far as we could find [121] is currently the best result

[1] BoW features are created by taking image patches from the training data and clustering them to create an image-based feature dictionary. New images from the test set can then be split into patches and mapped to this dictionary for classification

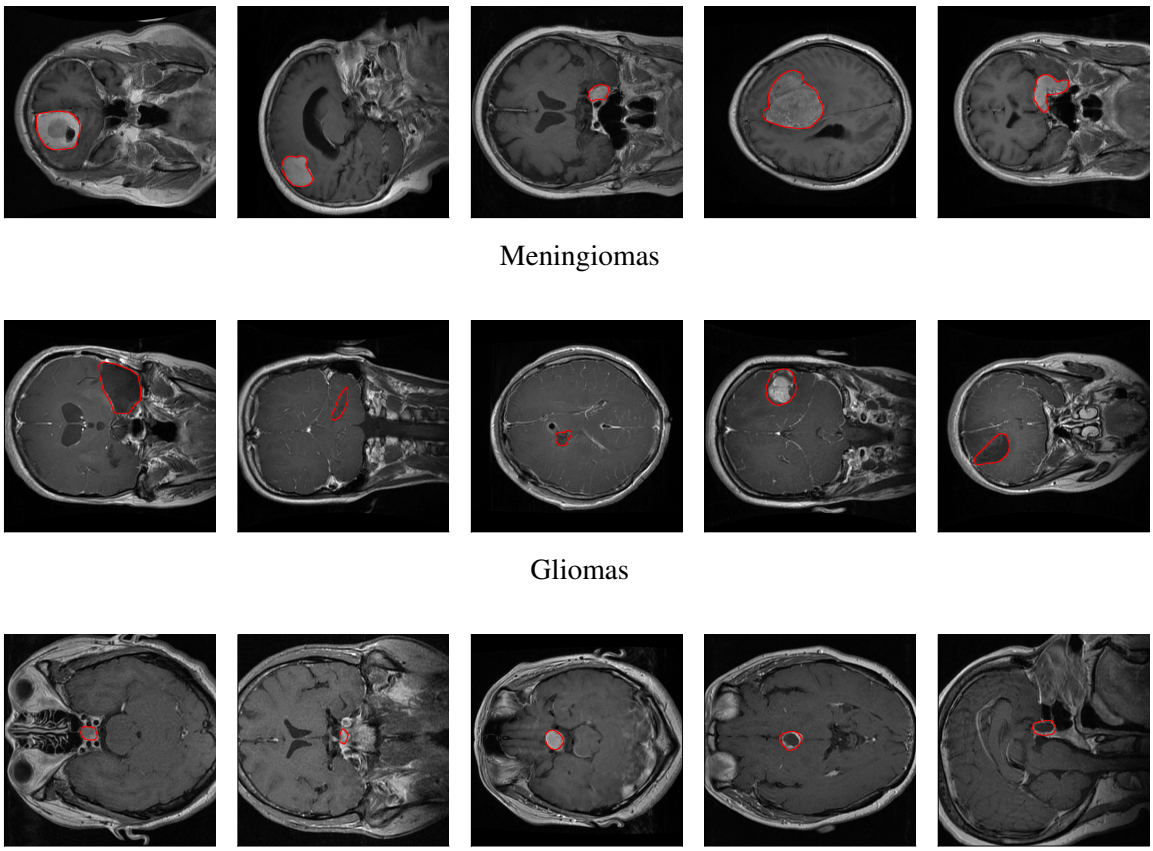
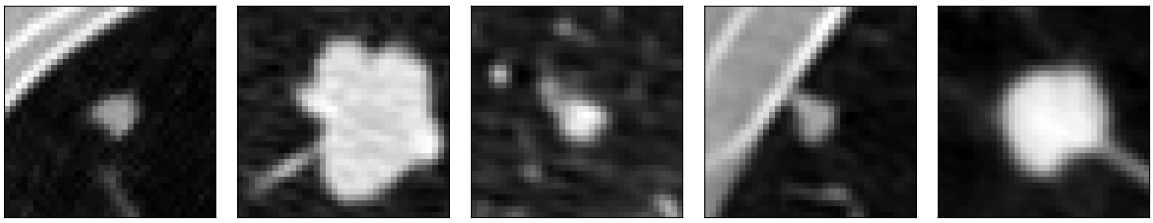
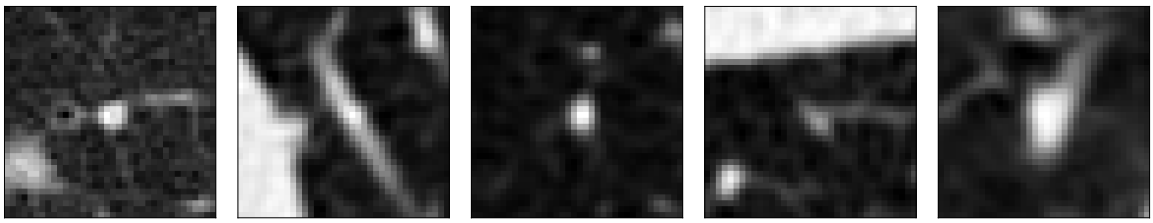


Fig. 7.1 Examples of T1-weighted contrast-enhanced MRI images from the brain dataset with the tumour segmentations in red. The three classes (meningiomas, gliomas, and pituitary tumours) are shown



Positive (Pathological) Nodes



Negative (Non-Pathological) Nodes

Fig. 7.2 Positive and negative examples of 24x24 mm^2 slices cut through the centre of nodes from the LUNA dataset

published with an accuracy of 0.97. In this study they use a 50-layer ResNet with in-training augmentation (horizontal and vertical flips, rotations, translations, zooming, zero component analysis whitening, shearing, and brightness manipulation).

7.2.2 LUNA Dataset

The 2016 Lung Nodule Analysis (LUNA) dataset consists of 888 CT scans for which pathological node candidates have been marked by a candidate detection algorithm, giving a total of 754,975 candidates [242]. The scans have been annotated by four experienced radiologists, with all node candidates $\geq$ 3 mm in their largest dimension accepted by at least three out of four radiologists marked as positive. Further details are given in [242] and [243]. This gives a total of 1,166 positive (pathological) and 753,809 negative nodes. For our experiments we randomly reduced the number of negative nodes to equal the number of positive nodes. This was done primarily to reduce the dataset size and therefore speed up training. Figure 7.2 illustrates some example images.

We could not find any studies using radiomic features directly for this task. However, several papers use radiomic features to predict nodule malignancy on the Lung Image Database Consortium (LIDC) dataset. This is the same patient dataset as the LUNA dataset, so these papers made sense as a starting point for our radiomic analysis. [64] and [35] both use intensity, texture, and wavelet features to classify nodules as benign or malignant with

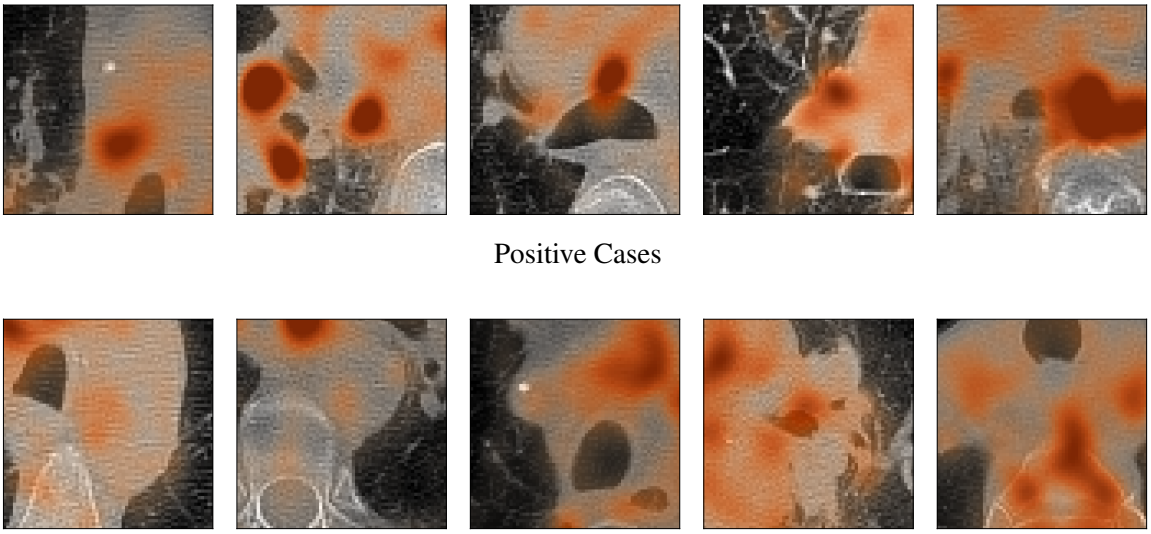


Fig. 7.3 Examples of 64x64 mm^2 slices from the mediastinal dataset. The top row shows positive cases with the node in the slice plane. The bottom row shows negative cases

accuracies of 0.83 and 0.84 respectively. Having been part of a Kaggle competition[2], there are many studies using CNNs to analyse the LUNA dataset. The best-performing model on the LUNA challenge false positive reduction leaderboard is currently [197] with a FROC score of 0.968 (using all 753,809 negative nodes). Because the preprocessing instructions are clearer, we will use the second-placed model (FROC score of 0.966) as a base for our CNN approach [244]. They use 24x24x24 mm^3 cubes centred on the node candidates to train a CNN, with flips, rotations, translations, and zooming used as in-training augmentation.

7.2.3 Mediastinal Dataset

Thirdly, we use the mediastinal database from Chapter 5. In this section we only perform phase two of the process and generated the cubes for phase one using test 104 (see 5.10 for full parameter details). This gave a dataset consisting of 12,989 3D cubes, 6,476 of which were positive and 6,513 negative. Because the cubes overlap the dataset has already been augmented. We assess the performance based on cube-level predictions, rather than worrying about node-level results. Some example cubes are shown in Figure 7.3.

[2]Kaggle is an online community of data scientists and machine learning practitioners which hosts machine learning competitions

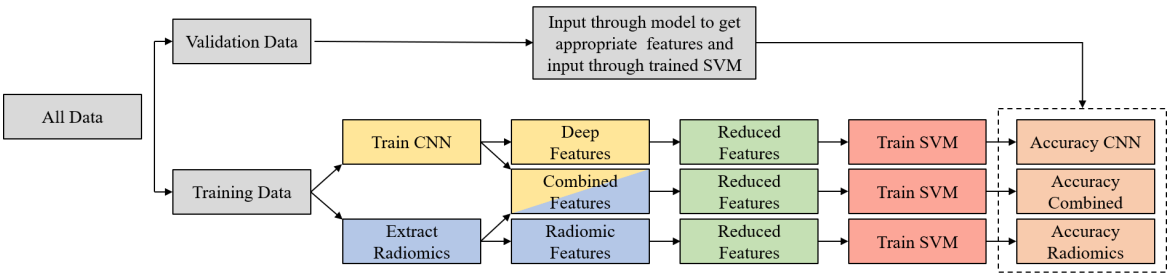


Fig. 7.4 Outline of the method for the comparison of deep and radiomic features. Data is first split into training and validation sets. The training data are used to create deep, radiomic, and combined feature sets. The numbers of features are reduced to prevent overfitting, before being used to train SVMs. The validation set is then used to evaluate the SVMs

7.3 Creation of Features

For each dataset we compared the predictive power of radiomic features to features derived from CNNs, training on a range of dataset sizes. Figure 7.4 outlines the experimental setup. For all experiments we tried to copy methods used in state-of-the-art studies, as these would ensure that we were getting the best possible performance and not biasing the study in favour of one type of feature or another. For cases where there was no state-of-the-art (such as our non-publicly available mediastinal dataset) we tested a range of parameters to optimise the experiments.

7.3.1 Deep Features

Given the time required to train a CNN, it was not feasible to optimise the hyperparameters at each dataset size. Instead, we optimised the parameters with the full dataset and used those parameters for other experiments. While this is clearly a compromise, studies have shown that optimal neural network architecture is mostly determined by the nature of the data and less affected by the sample size, suggesting that using the same CNN architecture for smaller dataset sizes is a reasonable approach [245]. We also used the same number of epochs across the range of dataset sizes. Although for smaller datasets this can lead to overfitting, we found it was not detrimental to the performance on the validation set. This was seen as a better method than using early stopping, which can lead to validation accuracy spikes being used (which do not fit the general learning trend).

7.3.1.1 Brain Dataset

We followed the method used in the best-performing publication described in 7.2.1 [121]. Images were first symmetrically cropped to 450x450 pixels (this removed background only)

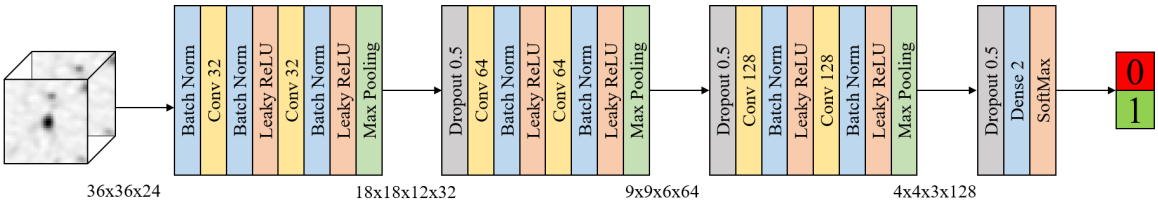


Fig. 7.5 CNN architecture used for deep learning analysis of the LUNA dataset. During convolutions padding was used so that the dimensions before and after convolution were the same. Convolution kernels were 3x3x3 voxels. Max pooling layers had kernels and strides of 2x2x2 voxels. The dimensions of the outputs after each max pooling layer are shown

and rescaled to between zero and one using the 99$^{\text{th}}$ percentile. We trained a 50-layer ResNet model with an Adam optimiser, a learning rate of 10^{-5}, and cross-entropy loss. In the original study they use a batch size of 32 and 500 epochs, but because of time and computer constraints we used a batch size of 8 and 100 epochs. We tested both combinations and found that this did not significantly affect the performance. The original study tested a wide range of augmentation techniques, but for simplicity we just used horizontal and vertical flips and rotations $\in [-20°, 20°]$. Once trained, features were obtained by taking the outputs of the final convolutional layer. This gave a total of 2048 features per image.

7.3.1.2 LUNA Dataset

Again, we used the same setup as in the paper referenced above [244]. Voxels were first resampled to have uniform spacings (x: 0.6666 mm, y: 0.6666 mm, z: 1 mm). A cutoff was used so that any HU values smaller/greater than -1000/1000 HU were set to -1000/1000 HU respectively. 36x36x24 voxel (24x24x24 mm^3) cubes were cut centred on each node coordinate. The values were finally rescaled to between zero and one.

The network architecture is shown in Figure 7.5. We used an Adam optimiser with cross-entropy loss, a learning rate of 10^{-4}, and a batch size of eight (not stated in the reference). Rotations of 90°, 180°, and 270°, shifts within two pixels in the *xy*-plane, zooms $\in [-10\%, 10\%]$, and flips in the x and y axes were randomly applied during training as augmentation (as in the reference). Models were trained for 50 epochs. Once trained, features were extracted from the final convolutional layer, giving 6144 deep features per image.

7.3.1.3 Mediastinal Dataset

We used the same CNN setup as test 52 in 5.12, extracting 2048 deep features per image from the final layer.

7.3.2 Radiomic Features

We calculated radiomic features using PyRadiomics [246], with their mathematical formulations detailed in [34]. Default settings were used unless otherwise stated. We first rescaled the datasets, then discretised the voxel values using a fixed bin width of size 25 (giving between 41 and 81 bins per dataset). Although there is no consensus on the number of bins to use, other studies have shown that bin counts in this range give good reproducibility and performance [29] [33] [32]. We used first-order, GLCM, NGTDM, GLRLM, GLSZM, GLDM, and first-order features based on gradient, exponential, LoG (with $\sigma = 1, 2, 3$), and wavelet filters (with coiflet 1 wavelets and high and low-pass filters in all dimensions). See Appendix B for full details.

7.3.2.1 Brain Dataset

The best-performing radiomic-based method for the brain dataset used BoW features, achieving an accuracy of 0.88 using the segmented tumour region as input [235] . We found that using the above radiomic features, calculated from full MRI images (without segmentation), gave a comparable accuracy. We hence used these features for simplicity (models with and without prior segmentation are compared in 7.12). Before extracting features, images were symmetrically cropped to 450x450 pixels and rescaled to between 0 and 1000 using the 99^{th} percentile. In total 229 features were extracted from each image. LoG filter-based features were not calculated, as these are not supported for 2D images in PyRadiomics.

7.3.2.2 LUNA Dataset

As with the deep features, we first used a cutoff to restrict HU values to between -1000 and 1000. Because no radiomic-based study could be found directly using the LUNA dataset, we had to optimise the preprocessing parameters ourselves. The cube size used to create the deep features may not have been the optimal size for radiomic-based features, so a number of different sizes were tested. We tested radiomic models with a voxel spacing as in the CNN models (x: 0.6666 mm, y:0.6666 mm, z:1 mm) and with a uniform voxel spacing of 1 mm. Uniform 12x12x12 voxel cubes performed best, so cubes of this size were cut centred on each node coordinate. We scaled the voxel values to between 0 and 2000 (to give 81 bins after discretisation). As mentioned above, we found several radiomic studies using the LIDC database (which is very similar to this task). The features used in these studies are available in PyRadiomics, so we again used PyRadiomics to extract all the above features, giving 355 per cube.

7.3.2.3 Mediastinal Dataset

To optimise the radiomic models for the mediastinal dataset, we again tried a range of cube sizes. Cubes of size 48x48x48 voxels were determined to be the best-performing, and the above features were thus extracted from these using PyRadiomics. Voxel values were rescaled by a factor of 200, then CT and PET features were extracted. This gave a total of 710 features per cube (355 CT and 355 PET).

7.3.3 Combined Features

As previously referenced, some research has shown benefit in combining the two feature types. To do this, for each experiment we created another feature set that combined the radiomic and deep features. This is the approach used in [191] and [222].

7.3.4 ImageNet Features

Because the brain dataset was 2D, we also generated features from a network pretrained on the ImageNet database. We used a 50-layer ResNet model but loaded weights from a model pretrained on ImageNet (using weights publicly available at [247]). The features were again obtained by taking the outputs of the final convolutional layer (with no further training). The performance of these features (which are from a model highly trained but from a non-medical domain) give a good comparison to the models trained on the actual task, but with far fewer data.

7.4 Classification

Once we had created the features for each method, we used a classifier to obtain classification performance results. There are many different classifier options, but due to time constraints we could not feasibly test all of them. Many studies have compared the performances of different classifiers [248] [249] [250]. While there is no consensus on the best choice for all cases, SVMs usually perform well and are widely used in research. SVMs are also used in several of the state-of-the-art radiomic studies for these datasets so are a logical choice. For consistency and simplicity we therefore used SVMs for all experiments. In each case we used a linear kernel with a C-value of 1. Other kernels and C-values were tested but did not significantly change the results. For the brain dataset we used a one-vs-all SVM setup. To check that there was not a wild disagreement between the SVM and another classifier, for

each dataset we ran tests at one small and one large dataset size, comparing the SVM results to those from a random forest.

It is unusual to train a CNN in this manner, splitting the feature creation and classification parts (in most cases convolutional layers would be followed by one or more fully connected layers to give a final classification, and the whole network would be trained together). We split the CNN so that we could directly study the features created by the CNN and reduce potential performance differences due to different classification methods. To check this was not significantly altering the overall performance, we compared the results using the CNN end-to-end to those using the CNN-SVM model and found no significant difference.

Using all the features to directly train an SVM could lead to overfitting, especially for experiments with few data. Therefore we first reduced the number of features. Again, there are numerous feature reduction methods available [81] [82]. We performed a two-stage feature reduction process; feature numbers were first reduced using a univariate ANOVA F-test, then any features above a correlation threshold were removed (for two correlated features the mean absolute correlation of each feature was calculated and the feature with the largest mean absolute correlation was removed). Before analysis, features were scaled to have zero mean and unit variance (using the training set to fit the scaling in each case). Because at different dataset sizes there may be different numbers of relevant features, for each experiment we optimised the F-test and correlation thresholds. The F-test threshold was varied to give the [10, 20, 40, 60, 80, 100, 120, 140] best features, and for each of these any features with absolute Pearson correlations greater than [0.95, 0.9, 0.8, 0.7, 0.6, 0.5] were removed. For all experiments we performed cross-validation with the training and validation sets randomly selected each time. This reduced the danger of overfitting during feature selection.

7.5 Performance of Radiomic and Deep Features vs Dataset Size

First, we compared the overall performance of models built using radiomic, deep, and combined features as we increased the dataset size.

7.5.1 Method

We plotted the number of data used for training (N) against the accuracy of the model. As all datasets were reasonably well balanced, accuracy was a valid indicator of performance. This

was done for models trained on radiomic, deep, combined, and (for the brain dataset only) ImageNet-based features. Power-law curves of the form

$$y = Ax^B + C \tag{7.1}$$

were fit to the radiomic and CNN results using non-linear least squares regression.

Experiments were repeated, each time randomly selecting the training and validation data. For the brain dataset each validation set comprised 40 patients and the training set ranged from 10 to 190 patients. For the LUNA dataset 400 images (200 positive and 200 negative) were used as validation, and a number of images ranging from 40 to 1800 were used for training. The mediastinal dataset was split into 34 patients for validation and between 10 and 80 patients for training. Each patient in the brain dataset did not have the same number of MRI images, and in the mediastinal dataset there was a wide range of positive nodes per patient (many patients had no positive nodes, whereas some had 4+). Splitting these datasets by patient rather than by image therefore meant that the number of data in the training sets would vary depending on the train/validation split. We used this setup, rather than splitting by image, as we did not want images from the same patient in both the training and validation sets. Multiple experiments were run to average out this effect. For the brain and LUNA datasets CNN-based experiments were repeated eight times. For the mediastinal dataset CNN-based experiments were repeated eleven times when using fewer than 60 patients and seven times when using 60 or more (because of the larger error with fewer data). Radiomic and ImageNet-based experiments were repeated 30 times. Standard error was used for the error bars. Coefficients of variation were calculated to compare the errors at low and high N.

7.5.2 Results and Discussion

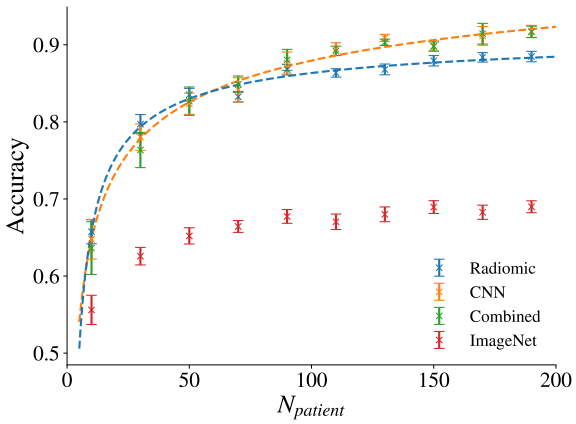
Figure 7.6 shows the learning curves for the three datasets. The deep features start to outperform the radiomic features at approximately $N_{patient} = 60$ for the brain dataset (corresponding to an average dataset size of approximately $N = 800$ images) and $N = 750$ for the LUNA dataset. At the maximum dataset size there are modest but significant differences in the performances of the two feature types (deep vs radiomic accuracies of 0.917 ± 0.008 vs 0.885 ± 0.007 ($p = 0.004$) for the brain dataset and 0.906 ± 0.007 vs 0.854 ± 0.003 ($p = 0.01$) for the LUNA dataset respectively). For the mediastinal dataset the performances of the two feature types are comparable at $N_{patient} = 80$ (0.750 ± 0.020 vs 0.759 ± 0.006 ($p = 0.65$)), but it is not possible to confidently predict trends because the error bars are much larger. This is likely because the number of positive nodes can vary a lot between patients, so each randomly selected training set could have contained a very different number of nodes.

The mediastinal dataset was more unbalanced (the positive cubes were oversampled before training using augmentation) so is harder to convert to a comparative dataset size, but at $N_{patient} = 80$ the average number of unique positive nodes in the training set was 111 and the average number of negative cubes was 4,202.

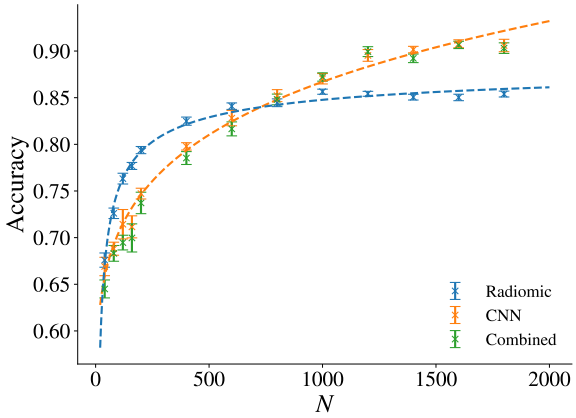
The power-law forms of the curves are in agreement with other studies [223] [224] [225] and show that the increase in performance behaves predictably even at low N (although at low N there are larger errors in the results). This implies that the underlying mechanism is predictable across the different modalities and dimensionalities. The key difference between the two curves is the gradient. For all datasets, at higher N the accuracy of the radiomic models improves more slowly and the curves start to level off sooner. This is probably because the deep features can keep changing and improving and are thus more powerful at higher N. Even when using the deep features there appears to be an upper limit to the performance, suggesting a limit to the amount of relevant information that they can extract from the images. In [223] (a non-medical study) this is called the *irreducible error region*, and it is suggested that it is due to mislabelling of data. In a medical setting it is more likely that the images fundamentally lack the information necessary to achieve perfect accuracy. There is disagreement even between experienced physicians for these tasks, implying that correct diagnosis is not always possible from the scans alone [251] [1].

The differences in curve shapes (i.e. the parameters in Equation 7.1) between the different problems are most likely due to differences in the complexity of the problems. [252] attribute the complexity of a classification problem to a combination of three main factors: the ambiguity of the classes, the sparsity and dimensionality of the data, and the complexity of the boundary separating the classes. Some work has been done trying to quantify problem complexity (see [253]), but these methods focus on feature-based rather than image-based problems. Radiomic features could be extracted from images and used as a proxy to measure complexity, but it is unclear whether this would be a reliable measure. More research into complexity for image-based problems could help quantify and explain differences in performance between problems.

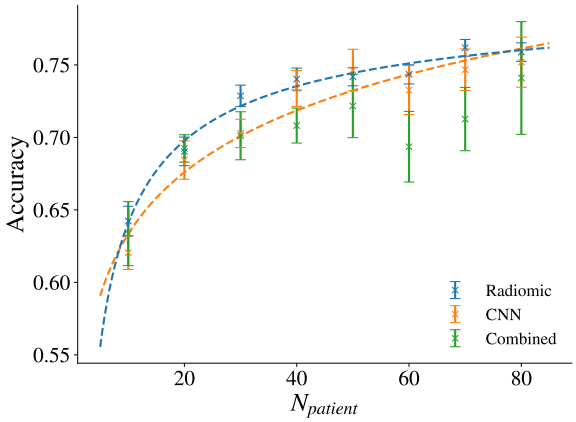
Combining the features did not improve the performance with the combined results closely mirroring the deep results for the brain and LUNA datasets (and to a lesser degree the mediastinal dataset). This is contrary to other studies, which have shown a gain in performance [191] [222]. It suggests that during feature selection the combined model is selecting mostly deep features, even at low N where the radiomic features perform better. This could be a shortcoming of the feature selection model, as at low N effective feature selection will be hindered by lack of data, meaning the best features are not always selected. The potential gain from combining the two feature types could also be task-dependent.



(a) Brain Dataset



(b) LUNA Dataset



(c) Mediastinal Dataset

Fig. 7.6 Learning curves for the brain, LUNA, and mediastinal datasets showing the performance of radiomic, deep, combined, and (for the brain dataset) ImageNet features. Power-law curves have been fit to the radiomic and CNN data

Table 7.2 Coefficients of variation of accuracy at low and high N ($N_{patient}$ for the brain and mediastinal datasets) for CNN and radiomic models

Dataset	***N***	**Coefficient of Variation**	
		CNN	Radiomic
Brain	30	0.11	0.13
	190	0.02	0.04
LUNA	80	0.04	0.06
	1800	0.02	0.02
Mediastinal	10	0.06	0.09
	80	0.06	0.05

For the brain dataset the ImageNet features did not perform well and were worse than the other methods for all N, achieving a maximum accuracy of 0.690 ± 0.008. This suggests that, while these features are highly optimised for ImageNet-type images, they are very poor at differentiating tumour types. This is perhaps not surprising given the two types of image are so different.

The errors for all experiments were higher at lower N, even having repeated these experiments more times. This is confirmed by the coefficients of variation of accuracy shown in Table 7.2. This is expected; at lower N there is more uncertainty in feature selection and thus in the accuracy. Because deep features themselves change, we thought the coefficients of variation may be larger than for the radiomic models, but this was not found to be the case.

For all three tasks tumours/nodes were not segmented, so the high accuracy of the radiomic models is perplexing. It is very unusual to train radiomic-based classification models without prior segmentation [26] [38]. This finding is explored further in 7.12.

These results show that CNN and radiomic models follow broadly the same patterns as dataset size increases, even in very different applications. CNNs are much harder to train and optimise than radiomic models, requiring more time and having more parameters to tune. Below a few hundred images a radiomic model is therefore a better option. However, when datasets are larger than a few hundred images CNNs benefit from their greater adaptability and clearly result in better performance. We saw no improvement in combining features for any of the datasets, contrary to what has been found in some other studies. This could imply that radiomic and deep features are extracting similar information from the images, and thus combining them does not give an increased performance. However, it could also be that we have too many features (especially at low N) to effectively perform feature selection so cannot exploit the extra information.

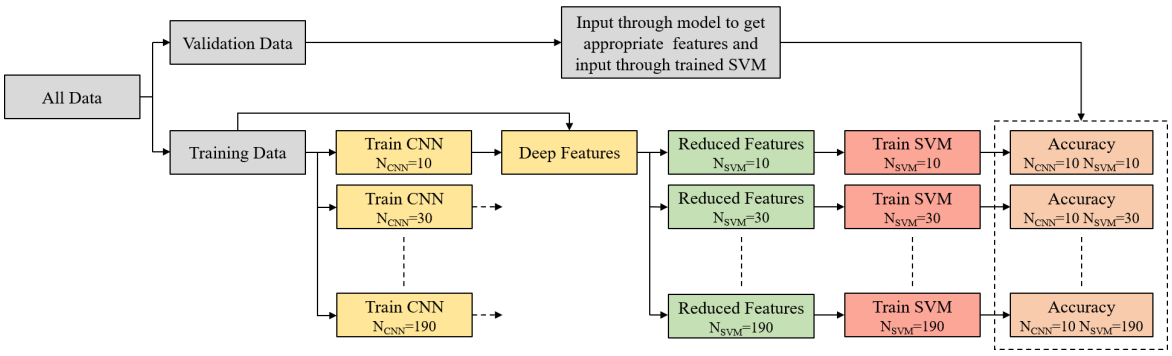


Fig. 7.7 Process diagram for separate evaluation of CNN and SVM learning. For each CNN trained with N_{CNN} data, features are extracted from all the training data. Using these features several SVMs are trained with N_{SVM} data. These are then evaluated using the validation set

7.6 Separate Evaluation of Feature Creation and Selection vs Dataset Size

The above results clearly show the power of deep features as datasets become larger. In this section we break these learning curves down into their separate CNN and SVM components. This allows us to gain a deeper understanding of how each part contributes to overall performance gain without the confounding factor of the other.

7.6.1 Method

Figure 7.7 details the process. The setup was identical to the above experiments, with features extracted from the final layer of each CNN and used to train an SVM. However, for each CNN trained with N data (which we denote N_{CNN}) we trained several SVMs with different numbers of data (N_{SVM}). Experiments were repeated, each time randomly selecting the training and validation data (for a particular CNN trained with N_{CNN} data, several SVMs were trained with N_{SVM} data. The validation set for these was the same, and the training data were randomly chosen from the non-validation data). This ensured that none of the data used to train the CNNs were then used for validation.

To analyse the results we plotted accuracy v N_{CNN} (at different values of N_{SVM}) and accuracy v N_{SVM} (at different values of N_{CNN}). The mean of repeated experiments was used to measure the performance. Experiments were repeated as previously and standard error was again used. Note that for the brain and mediastinal datasets N_{CNN} and N_{SVM} implicitly refer to the number of patients.

7.6.2 Results and Discussion

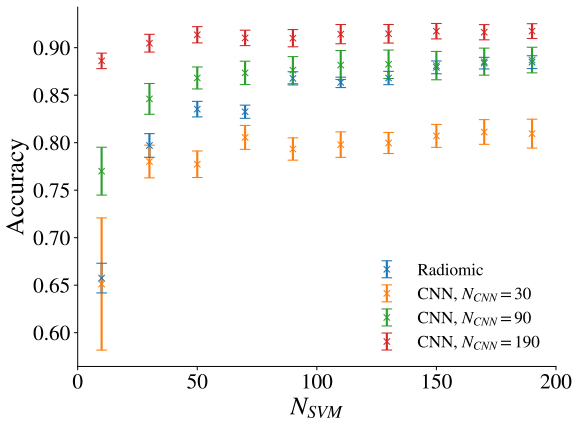
Figure 7.8 shows the learning curves of the three datasets in relation to N_{SVM}, the number of data used to train the SVM. Curves for four feature sets are shown; radiomic features and features derived from three CNN models (small, medium, and large N_{CNN}). For the brain dataset we see that above approximately $N_{\mathrm{SVM}} = 70$ there is no significant improvement in the accuracy for any of the curves, and thus we have reached the maximum accuracy possible with these features. Similar plateaus are seen for the LUNA dataset at around $N_{\mathrm{SVM}} = 700$ and the mediastinal dataset at around $N_{\mathrm{SVM}} = 30$. These show the limit of radiomic features. This is the only mechanism by which the radiomic performance can improve and it quickly plateaus. In contrast, we see that at higher N_{CNN} the CNN accuracy plateaus higher. As the CNNs are trained with more data the features become more discriminative and thus have a higher performance potential.

Figure 7.9 shows how the accuracy improves with N_{CNN}, the number of data used to train the CNNs. Curves are shown for small, medium, and large N_{SVM}. In contrast to the above graphs, which plateau very quickly, here we see the performances improve significantly up to approximately $N_{\mathrm{CNN}} = 170$ for the brain dataset, $N_{\mathrm{CNN}} = 1500$ for the LUNA dataset, and $N_{\mathrm{CNN}} = 50$ for the mediastinal dataset (although errors here are much larger). This demonstrates that CNN-based models can continue to improve more significantly at higher N compared to radiomic-based models.

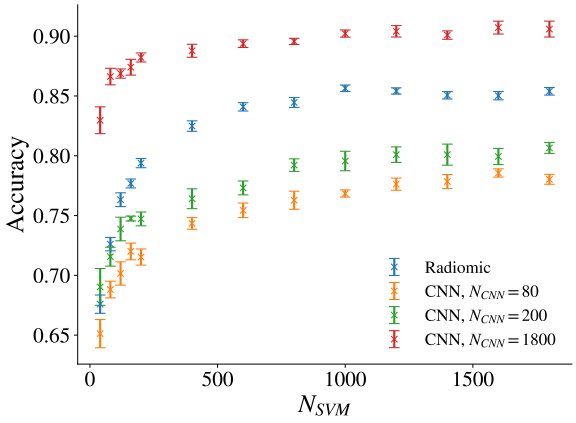
These results suggest that the factor limiting performance is the number of data available to derive optimal features for the task and not the number of data available to build the classification models based on the features. For CNN models this is not a problem, because with more data the features improve. For radiomic models it implies that efforts should be dedicated to identifying new features more suited to the tasks. Some studies have done this. [254] extracted tortuosity and dilation features from arteries and veins to grade retinopathy of prematurity from retinal images. [255] explored using orientational distributions of tumour cells to grade prostate cancer. More problem-specific features could help to improve radiomic models performance while maintaining their relatively simple (compared to CNNs) designs.

7.7 Comparison of Image-Level Predictions

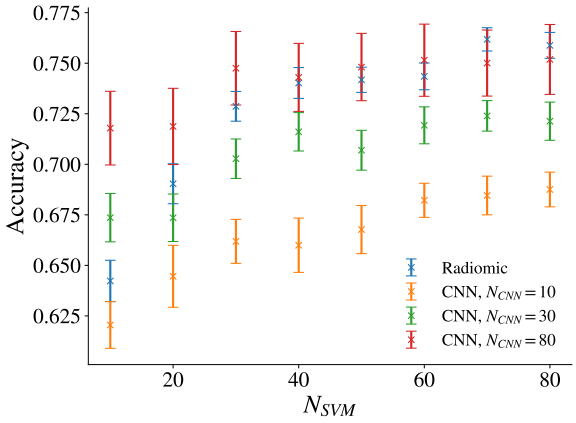
Moving on from measures of overall performance, we wanted to see to what degree the models' individual predictions agreed. To do this we compared image-by-image results for the radiomic, CNN, and combined models to see whether their predictions strongly overlapped. This gave an indication of whether radiomic and CNN models were extracting the same information from the images.



(a) Brain Dataset

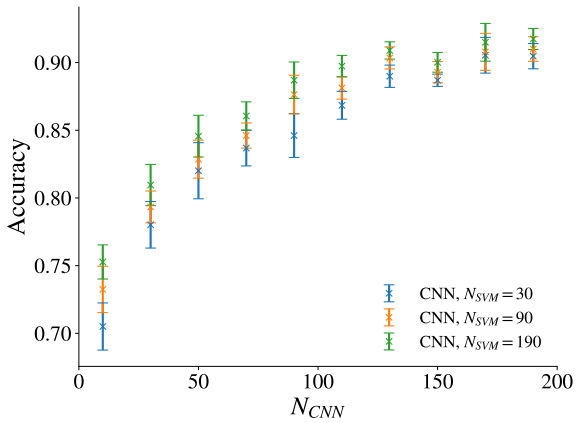


(b) LUNA Dataset

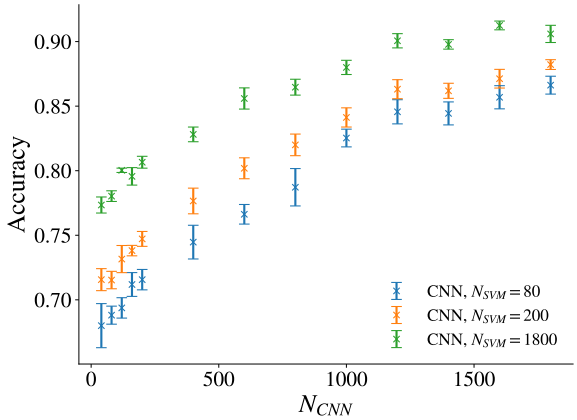


(c) Mediastinal Dataset

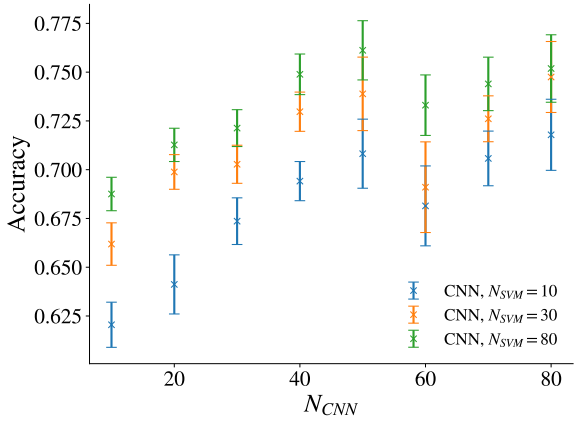
Fig. 7.8 Learning curves for the three datasets showing how the performance changes as the number of data used to train the SVM increases (N_{SVM}). The performances of radiomic features and deep features from CNNs trained with small, medium, and large datasets are shown in each case



(a) Brain Dataset



(b) LUNA Dataset



(c) Mediastinal Dataset

Fig. 7.9 Learning curves for the three datasets showing how the performance changes as the number of data used to train the CNN increases (N_{CNN}). The performances of deep features with SVMs trained with small, medium, and large datasets are shown in each case

7.7.1 Method

The analysis was performed at small, medium, and large N ($N_{\text{patient}} = 30$, 90, and 190 for the brain dataset, $N = 80$, 200, and 1800 for the LUNA dataset, and $N_{\text{patient}} = 10$, 40, and 80 for the mediastinal dataset). In each case we used the same training and validation sets and plotted Venn diagrams to show the overlap of results on the validation set.

7.7.2 Results and Discussion

Figure 7.10 shows Venn diagrams of the overlap of predictions for the CNN, radiomic, and combined models trained on the three datasets for three values of N (small, medium, and large). At high N there is a high level of agreement between all three methods i.e. in the vast majority of cases they are predicting the same result for each image (agreement in 90%, 85%, and 81% of cases for the brain, LUNA, and mediastinal datasets respectively). Because of this it should not be surprising that we do not gain anything when using the combined models. The agreement is lower at low N (82%, 60%, and 50% respectively). This is probably because with fewer data feature reduction and classification are more volatile, leading to differences in prediction. Additionally, at low N the CNN will be extracting sub-optimal features. One would think that this would result in an improved performance when combining the features, however we saw above that this is not the case. Instead, the combined predictions mostly agree with the CNN predictions. This could be because at low N there are not enough datapoints to perform effective feature selection, so the models are not able to exploit the extra information.

7.8 Interpreting CNNs

As mentioned in the introduction, being able to interpret models is key to trusting them. Additionally, as automated models began to outperform humans we would like to understand how they make their decisions so we can gain a deeper biological understanding of the data. If a complicated model is not interpretable a simpler model may be preferable, even if the performance is worse. In this section we use one gradient-based method (Grad-CAM) and one perturbation-based method (occlusion sensitivity) to try to understand which biological features the CNNs are using to make their decisions for the three datasets [143] [100]. We interpret the models *globally*, i.e. explain which general characteristics the models are using and *locally*, i.e. explain the predictions for individual cases.

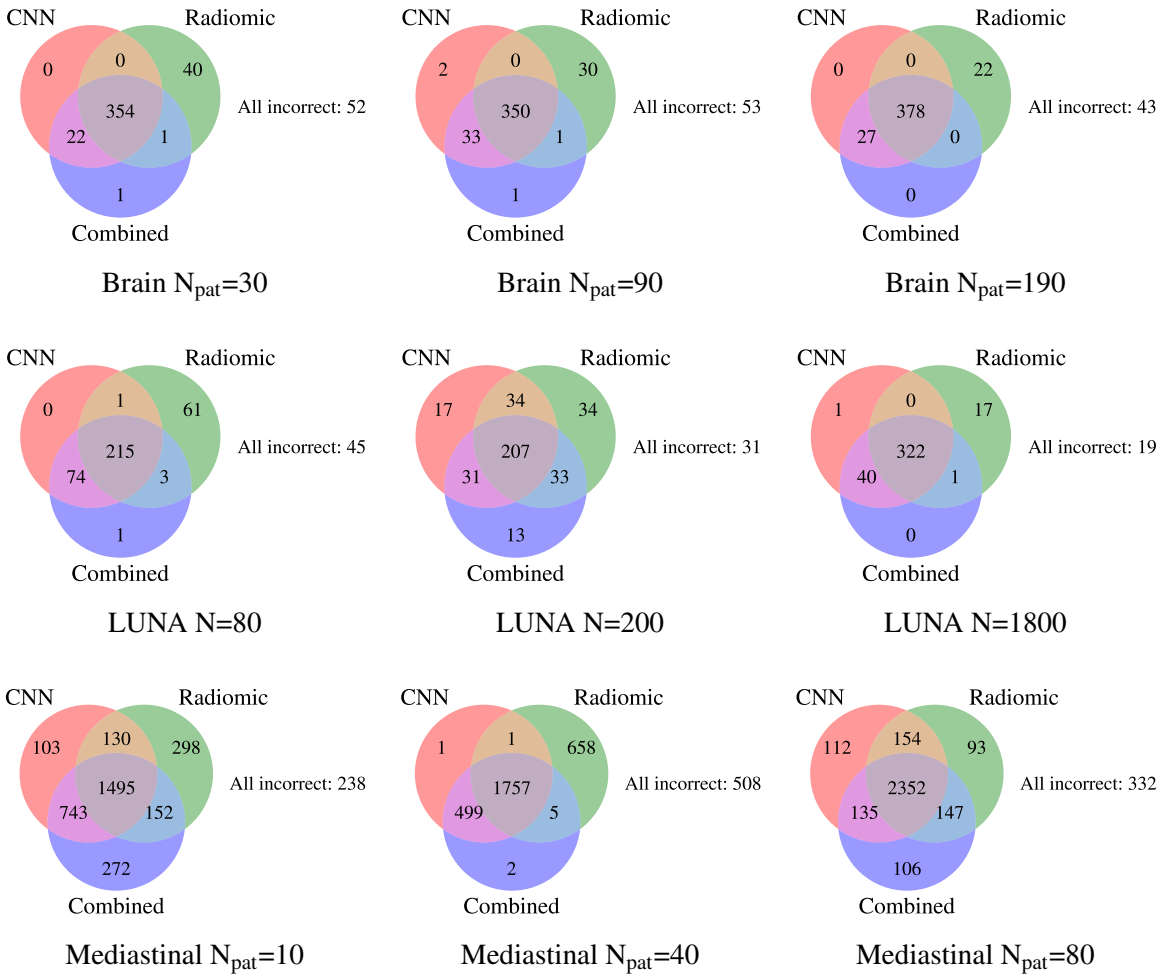


Fig. 7.10 Venn diagrams showing the overlap of image-level results at small, medium, and large N for all three datasets. $N_{pat} \equiv N_{patient}$

7.8.1 Method

For each dataset we attempted to interpret one of the trained CNNs at large N ($N_{\text{patient}} = 190$ for the brain dataset, $N = 1800$ for the LUNA dataset, and $N_{\text{patient}} = 80$ for the mediastinal dataset). Grad-CAM maps were calculated using keras-vis [256], in all cases using the gradients of the predicted class in the final convolutional layer. Grad-CAM maps were not calculated for the mediastinal dataset as the final convolutional layer was too small to give any meaningful insight. Output maps were rescaled to between zero and one. Examples used are all from the validation data.

We generated occlusion maps as described in 4.4. The sizes of the occluding regions were chosen to make the maps as informative as possible. For the brain dataset the occluding squares had side length 100 pixels and the stride between occlusions was 25 pixels. The corresponding parameters for the LUNA dataset were 6 voxels and 3 voxels, and for the mediastinal dataset 16 voxels and 8 voxels. We plotted the maps using a relative scale, as we found this produced more informative occlusion maps than an absolute scale from zero to one.

To be useful, saliency maps need to be specific. If the high-attention region of a saliency map covers most of the image it is not informative. To quantify this, for each dataset we calculated the average size of the high-attention regions (defined as >0.8 for Grad-CAM maps and within 20% of the lowest certainty prediction for occlusion maps).

7.8.2 Results and Discussion

Figures 7.11, 7.12, and 7.13 show the Grad-CAM and occlusion maps for a selection of cases for the brain, LUNA, and mediastinal datasets respectively. The actual and predicted classes are shown below each case. For the brain dataset, Figures 7.11a to 7.11c are correctly predicted as class 0 (meningiomas). The Grad-CAM and occlusion maps for these cases show that the CNN has located and is focusing on the tumour regions, which is reassuring. The Grad-CAM map in Figure 7.11a also has high intensity regions around the skull, which are harder to explain. This is concerning but could be due to a similar shaped skull in the training set. Figures 7.11d to 7.11f are correctly predicted class 1 (glioma) cases. Here the saliency maps are less directed, covering large areas of the brain and not necessarily the tumour area. The saliency maps in Figure 7.11e are particularly wild, with the Grad-CAM and occlusion maps disagreeing. Similarly, Figures 7.11g to 7.11i show correctly predicted class 2 (pituitary tumour) images, and again we see the saliency maps show the CNN using large areas of the brain to make its decisions. These maps suggest that some feature of the textures in these regions is aiding the CNN. Finally, Figures 7.11j to 7.11l show some

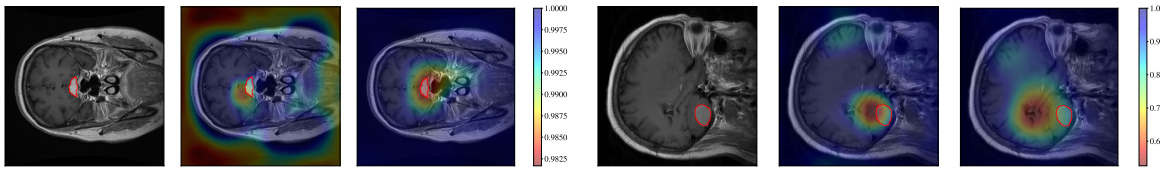
incorrectly predicted images. In Figures 7.11j and 7.11k the CNN has completely failed to focus on the brain. In Figure 7.11l we see that the CNN has successfully located the tumour but has not classified the image correctly. For these incorrect cases the occlusion maps are not useful, as the prediction is incorrect even with none of the image occluded.

For the LUNA dataset, Figures 7.12a to 7.12c show correctly predicted negative cases. These maps are unilluminating, and this was the case for most negative cases. It seems the CNN uses the absence of a node to classify an image as negative, rather than anything specific from the images. Figures 7.12d to 7.12f illustrate correctly predicted positive cases. In each case the CNN is focusing on the hyperdense regions (i.e. the nodes). The occlusion maps in Figures 7.12d and 7.12f show that the image is incorrectly classified if the central node is occluded. This makes sense. Figure 7.12g shows a false positive, again with the network focusing on the large hyperdense region. Clearly this large shape was misdiagnosed as a pathological node. Figure 7.12h shows a false negative, but from the saliency maps it is unclear why this prediction was made.

Turning to the mediastinal dataset, Figures 7.13a to 7.13c show correctly identified negative cubes. The saliency maps are not too enlightening, but we see in Figure 7.13a that when a certain area is occluded we predict a false positive. For the positive cases in Figures 7.13d to 7.13f we see that when the high uptake PET region (the positive node) is occluded the prediction is lower. This reassures us that the CNN is using the nodes to make its decisions. The final three figures show incorrectly predicted examples. The occlusion maps in Figures 7.13g and 7.13i are not particularly useful, however in Figure 7.13h we see that the prediction becomes correct if the high uptake PET region to the left of the image is occluded. This suggests that this region is being misidentified as a positive mediastinal node (when in fact it is not in the mediastinum).

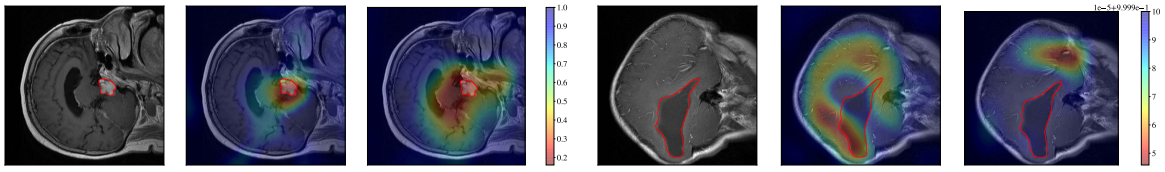
The average percentage area of the brain Grad-CAM maps with a score above 0.8 was 4% and the average percentage area within 20% of the lowest certainty prediction on the occlusion maps was 14%. For the LUNA dataset the corresponding percentages were 1% and 4%. We did not calculate these for the mediastinal dataset because calculating saliency maps for the 2000+ cubes in the validation set was unfeasible. Although these figures are quite low, suggesting the maps are specific, from the examples we can see that the size of high-attention regions varies a lot. In many cases they are too large to give precise information about which areas of the image are being used.

These results show the messiness and imprecision of interpreting CNNs; it is not an easy task. The maps are aesthetically pleasing, but it is not clear how much they can actually contribute to a deeper understanding of the problems. Even when they focus on the pathological region, the spatial resolution is far too low to understand in detail what led to



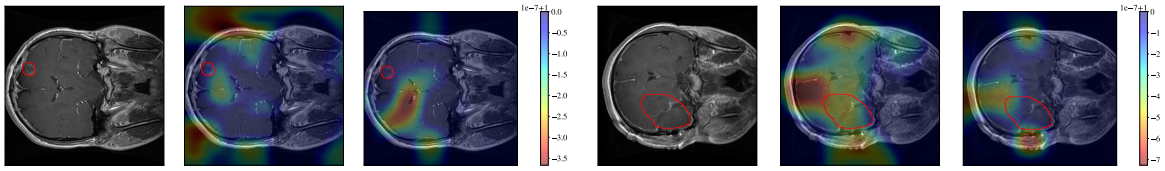
(a) Actual 0 Pred 0

(b) Actual 0 Pred 0



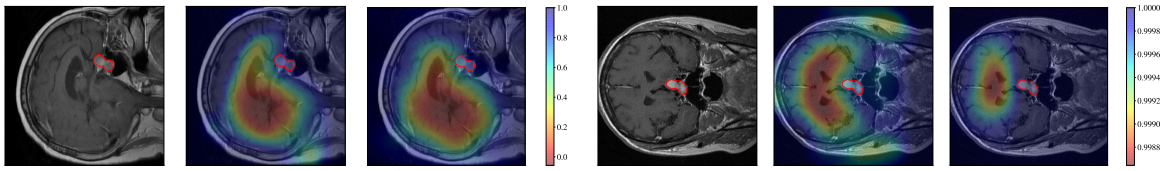
(c) Actual 0 Pred 0

(d) Actual 1 Pred 1



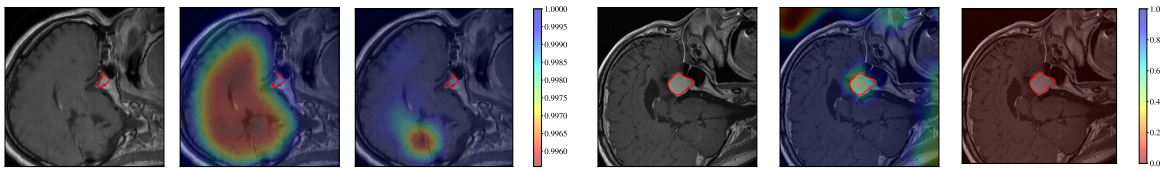
(e) Actual 1 Pred 1

(f) Actual 1 Pred 1



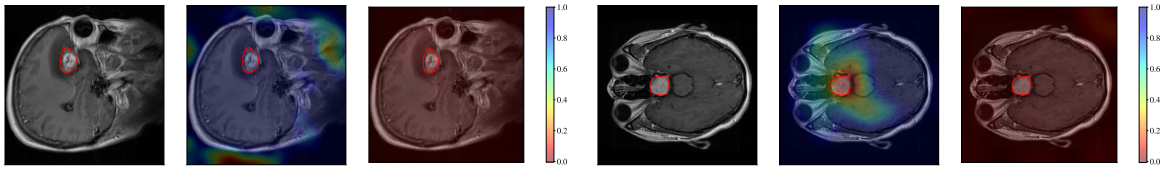
(g) Actual 2 Pred 2

(h) Actual 2 Pred 2



(i) Actual 2 Pred 2

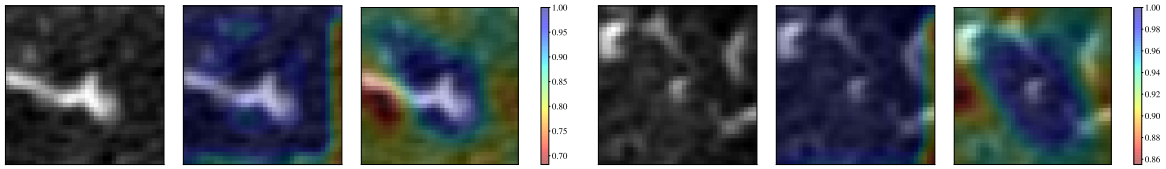
(j) Actual 0 Pred 2



(k) Actual 0 Pred 1

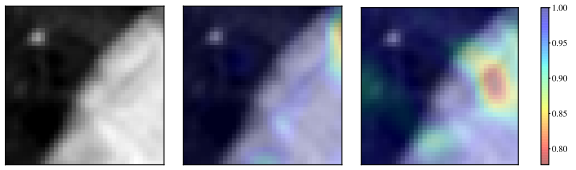
(l) Actual 2 Pred 1

Fig. 7.11 Each set of images shows the input MRI image (left), the Grad-CAM map (centre), and the occlusion sensitivity map (right) for an example from the brain dataset. The red outline shows the tumour delineation. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area was important for the classification) and blue indicates the inverse. The scale on each occlusion map is different so scales are shown. Class 0 are meningiomas, class 1 are gliomas, and class 2 are pituitary tumours

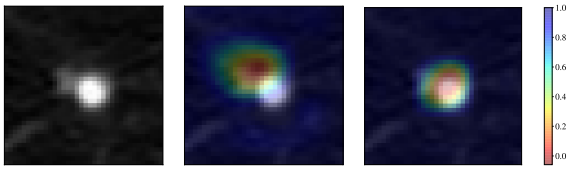


(a) Actual 0 Pred 0

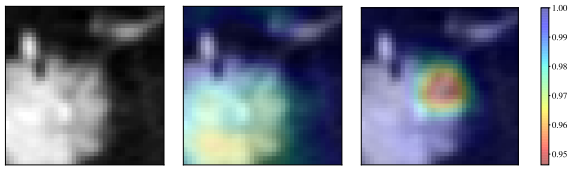
(b) Actual 0 Pred 0



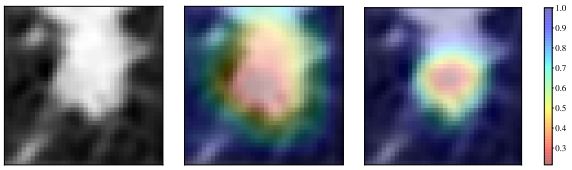
(c) Actual 0 Pred 0



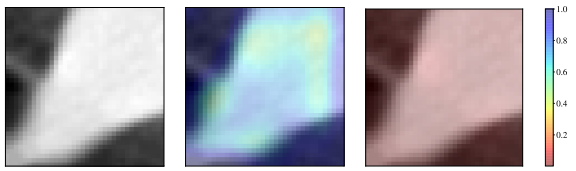
(d) Actual 1 Pred 1



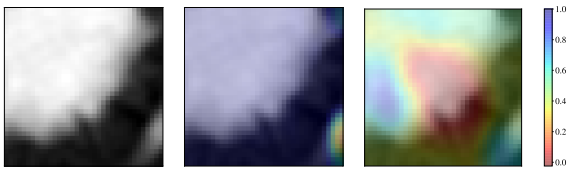
(e) Actual 1 Pred 1



(f) Actual 1 Pred 1

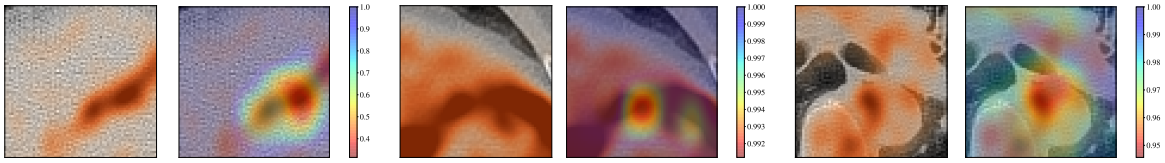


(g) Actual 0 Pred 1

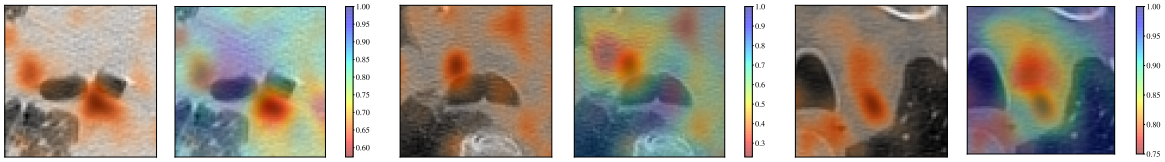


(h) Actual 1 Pred 0

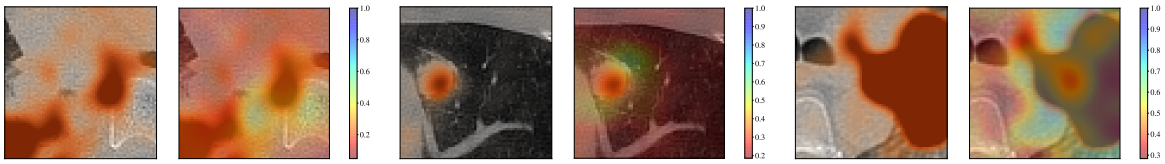
Fig. 7.12 Each set of images shows central slices from the input CT image (left), the Grad-CAM map (centre), and the occlusion sensitivity map (right). Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area was important for the classification), and blue indicates the inverse. The scale on each occlusion map is different so scales are shown. Class 0 are negative nodes and class 1 are positive nodes



(a) Actual 0 Pred 0 (b) Actual 0 Pred 0 (c) Actual 0 Pred 0



(d) Actual 1 Pred 1 (e) Actual 1 Pred 1 (f) Actual 1 Pred 1



(g) Actual 1 Pred 0 (h) Actual 0 Pred 1 (i) Actual 1 Pred 0

Fig. 7.13 Each set of images shows a 2D slice form the input PET/CT image (left) and the occlusion sensitivity map (right). Red areas on the occlusion maps indicate a drop in the prediction when that area was occluded (i.e. that area was important for the classification), and blue indicates the inverse. The scale on each occlusion map is different so scales are shown. Class 0 are negative cubes and class 1 are positive cubes

the correct or incorrect classification. Other interpretation techniques (e.g. guided-backprop [103] and SHAP [257]) have higher spatial resolution, so may have given more fine-level details.

We can perhaps infer some global trends- for gliomas in the brain dataset the CNN focuses mainly on the tumour region, whereas for meningiomas and pituitary tumours it uses a wider area around the tumour. This is in line with other studies showing that information is contained in the area around the tumour [220] [235]. For the LUNA dataset we see that for positive predictions the CNN is focusing on the hyperdense regions, but it is hard to be more precise as the maps are very coarse-grained. Locally these techniques allow us to diagnose some incorrect predictions, such as in Figures 7.12g and 7.13h, though in many other cases they do not help. The Grad-CAM and occlusion maps broadly agree, which reassures us that the maps are not completely random. However, some of the maps are very strange and hard to explain (e.g. in Figures 7.11a, 7.11e, and 7.12a), and these cast doubt on the reliability of our other findings. It is hard to know when we can and cannot trust them.

Studies have shown Grad-CAM to be one of the more reliable saliency map analysis methods [153] [154], but as detailed in 4.4 there are a plethora of alternatives. Using different methods could give more insights, but this is of course more time-consuming. In addition, some work has thrown into question the value and reliability of all of these methods. [150] evaluated various interpretation metrics and found that most failed when tested on their *utility*, *repeatability*, and *reproducibility*. They concluded that given the maps' lack of robustness they should not be used to draw conclusions in a high-risk domain such as medical imaging. [151] and [152] highlighted similar problems. The results in this section, and any methods used to interpret CNNs, should therefore be treated with a degree of caution. More work needs to be done to systematically test these methods to determine what we can learn from them and in which cases they are strong and weak. From this we could more intelligently deploy an interpretation technique specific to our needs and trust its findings.

We tried to give a quantitative assessment of the usefulness of the interpretations by using the size of the attention region, but this is clearly quite crude. Several metrics have been developed to evaluate saliency maps, but these usually rely on a reference human attention map [258] [259]. One of the motivations for using CNNs in medicine is to improve on human performance, potentially by finding biological indicators that experts miss. These metrics are therefore not appropriate. Other measures such as *stability* (the degree to which similar explanations are given for similar samples of the same class) and *consistency* (the degree to which different models trained on the same task give the same explanations) were not investigated but are also important when assessing the trustworthiness of an interpretation [153].

7.9 Interpreting Radiomic Models

Although from a radiomic model we can easily establish which features are the most important, it is often hard to determine what more complicated features actually mean biologically. In this section we examine the degree to which the decisions of our radiomic models can be understood. Again, we try to interpret the models globally and locally.

7.9.1 Method

For each dataset we chose one radiomic model at large N ($N_{\text{patient}} = 190$ for the brain dataset, $N = 1800$ for the LUNA dataset, and $N_{\text{patient}} = 80$ for the mediastinal dataset) and found the ten most important radiomic features using the coefficients of the SVM models (any more than ten features becomes difficult to interpret). Looking at how these features differed between classes we could determine the motivation behind the classifications. When features were hard to interpret (e.g. wavelet-based features), we then attempted to exchange them for more interpretable features that were highly correlated with them. We plotted distributions of both the original and interpretable features to assess how the replacements affected the features' ability to distinguish the classes.

To test how these replacements affected the performance, for each dataset we built new SVM models to compare the performance of the original radiomic model, to the performance of a model built with the top ten features, to the performance of a model built with the interpretable alternatives to the top ten features. The aim of these tests was to see if we could simplify the radiomics model while retaining performance.

A second approach to interpretability is not to replace the original model, but to instead build a simpler model to try and replicate its results. To do this, for each dataset we built one SVM using the top ten features and one using their interpretable alternatives, but with the predictions of the original model as a ground truth. The accuracies of these models showed how capable the reduced sets of features were of replicating the original model's results.

7.9.2 Results and Discussion

Figure 7.14 illustrates how the top ten radiomic features for the brain dataset vary between the three classes. We can see for the three examples that no one feature clearly determines the class, highlighting the importance of using a multi-feature model for this task. Globally, we see that higher Wavelet-HL Interquartile Range and lower GLDM Small Dependence Low Grey Level Emphasis values are indicative of class 2 (pituitary tumours), whereas higher GLDM Small Dependence Low Grey Level Emphasis values are indicative of class 1

(gliomas). Locally, for an individual case (□) we see that its low GLCM Inverse Variance and high GLDM Small Dependence Low Grey Level Emphasis values are probably resulting in it being misclassified as class 1.

However, most of these features are hard to interpret. Figure 7.16 shows some example wavelet images. Wavelet-LH, wavelet-HL, and wavelet-HH are identifying edges but relating these to biological indicators does not seem possible. We therefore replaced these features with more interpretable alternatives (see Table 7.3). In all cases it was possible to find an alternative feature with $|r| > 0.7$. From these we can deduce that the spread of voxel values is important (Variance, Mean, 10^{th} Percentile) as is the homogeneity of the image (GLCM Inverse Difference). Figure 7.15 shows how these replacements distinguish the classes compared to the original features. We see that while some replacements largely retain the original distributions (e.g. Wavelet-LH Robust Mean Absolute Deviation and GLSZM High Grey Level Zone Emphasis), others do not (e.g. GLDM Dependence Variance and GLRLM Run Entropy), and this is not necessarily linked to the strength of correlation between the two features.

Comparing the performances of the models, the original model achieved an accuracy of 0.85 using 51 radiomic features. Using only the top ten features gave an accuracy of 0.81, but replacing them with the interpretable alternatives reduced the accuracy to 0.70. Using the original model's predictions as a ground truth, the top ten features achieved an accuracy of 0.89, indicating a high fidelity to the original model. Using the interpretable alternatives we could only reproduce the original model's results with an accuracy of 0.71.

Moving on to the LUNA dataset (Figure 7.17), we see Skewness and Median, two easily interpretable features, are the most important. These indicate that the range of voxel values and the proportion of high to low voxel values are key to the model's decisions. GLCM Autocorrelation, which is related to the coarseness of features, is also important. As with the brain dataset we see that no one feature definitively separates the two classes, so multi-feature analysis is necessary. Taking an individual example (□), we can explain its misclassification as negative by noting that it has a lower 90^{th} percentile than most positive nodes (its feature values are very similar to △, which is a negative case). Other features such as LoG-2mm 10^{th} Percentile and Wavelet-LLL Range are harder to interpret but can be correlated with less opaque features, as detailed in Table 7.4. Note that for LoG-3mm Median we could not find an interpretable alternative. These alternatives again indicate that first-order features such as Range and Skewness are important to the classification. Comparing these features in Figure 7.18, we see a good agreement for most, but again some replacements have not worked as well (e.g. LoG-2mm 10^{th} Percentile and LoG-1mm Skewness).

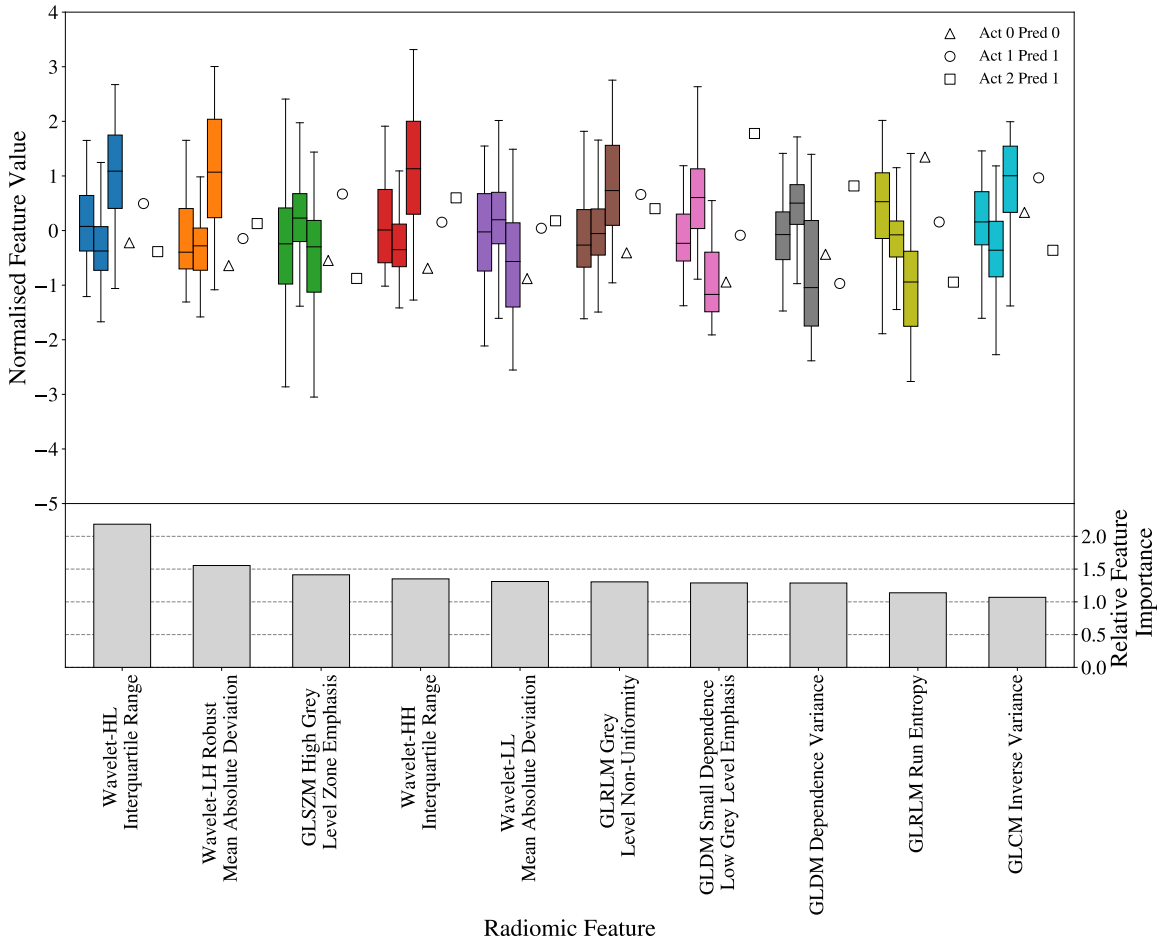


Fig. 7.14 The boxplots show how the three classes from the brain dataset are differentiated by each of the ten most important radiomic features. □, ○, and △ are three example cases. By comparing their feature values to the boxplots we can deduce which features are important to their classification. Below, the bars represent the feature importance (defined as the average SVM coefficient) of each of the ten features. Class 0 are meningiomas, class 1 are gliomas, and class 2 are pituitary tumours

Table 7.3 More interpretable alternatives to the original radiomic features used in the radiomic model for the brain dataset with their corresponding correlations. Cases where the original feature was interpretable are left blank

Original Feature	**Interpretable Alternative**	***r***
Wavelet-HL Interquartile Range	GLCM Inverse Difference	-0.95
Wavelet-LH Robust Mean Absolute Deviation	GLSZM Zone Percentage	0.92
GLSZM High Grey Level Zone Emphasis	Variance	0.85
Wavelet-HH Interquartile Range	GLRLM Run Length Non-Uniformity	0.83
Wavelet-LL Mean Absolute Deviation	Mean Absolute Deviation	1.00
GLRLM Grey Level Non-Uniformity	-	-
GLDM Small Dependence Low Grey Level Emphasis	Mean	-0.84
GLDM Dependence Variance	10^{th}Percentile	-0.83
GLRLM Run Entropy	GLCM Informational Measure of Correlation 1	-0.82
GLCM Inverse Variance	-	-

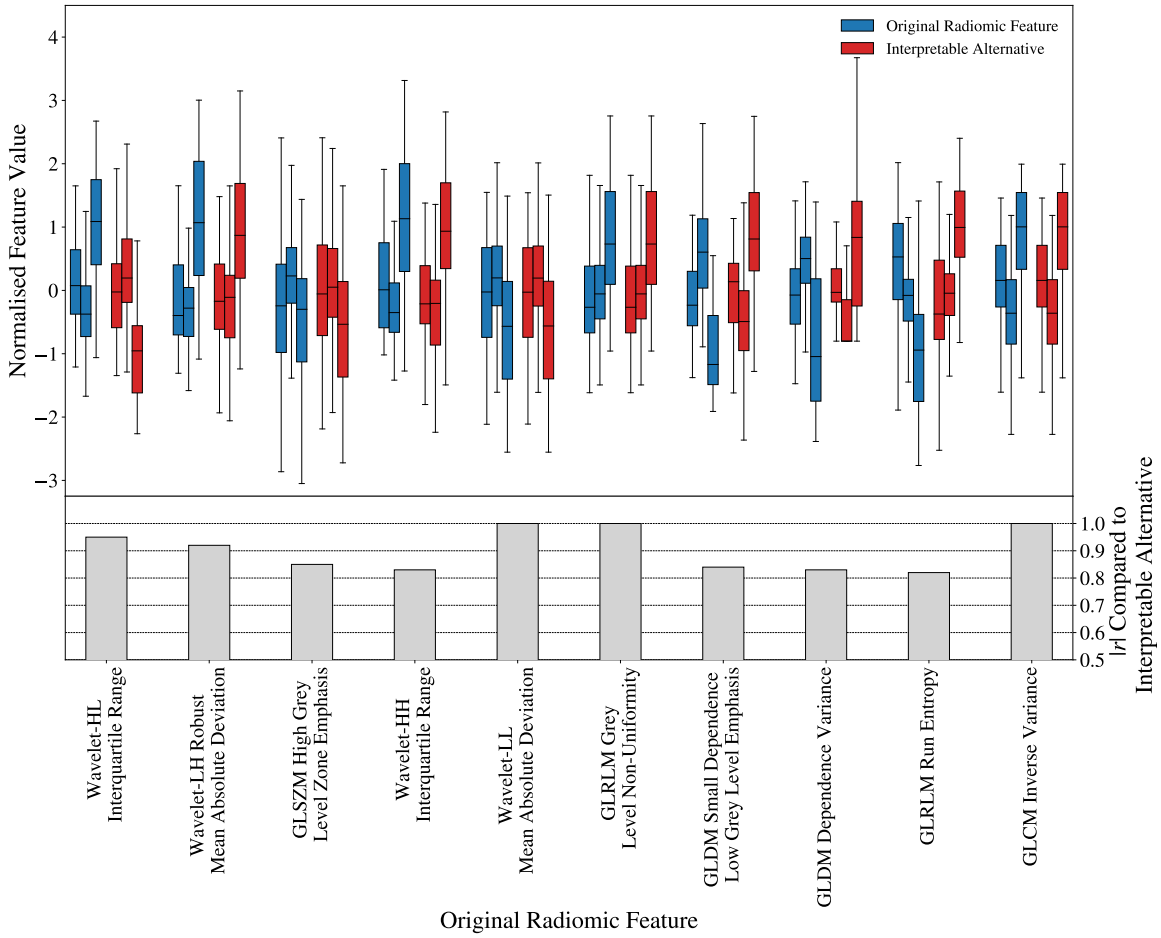
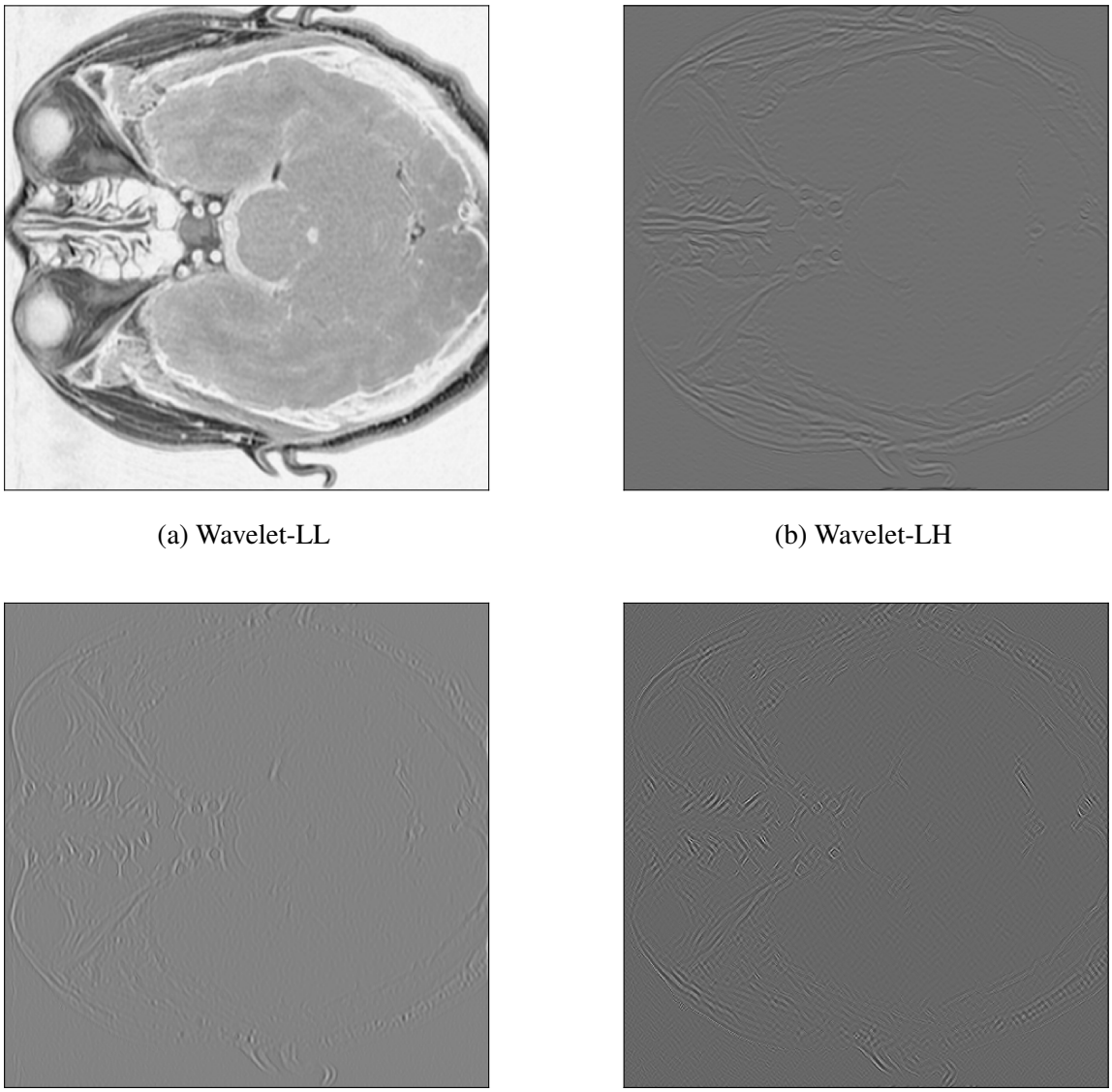


Fig. 7.15 The boxplots show how the original radiomic features and their more interpretable alternatives compare for the brain dataset, separately for the three classes. Below, the bars show the magnitude of correlation between the features ($|r|$)



(a) Wavelet-LL

(b) Wavelet-LH

(c) Wavelet-HL

(d) Wavelet-HH

Fig. 7.16 Wavelet-transformed images for an example brain MRI image

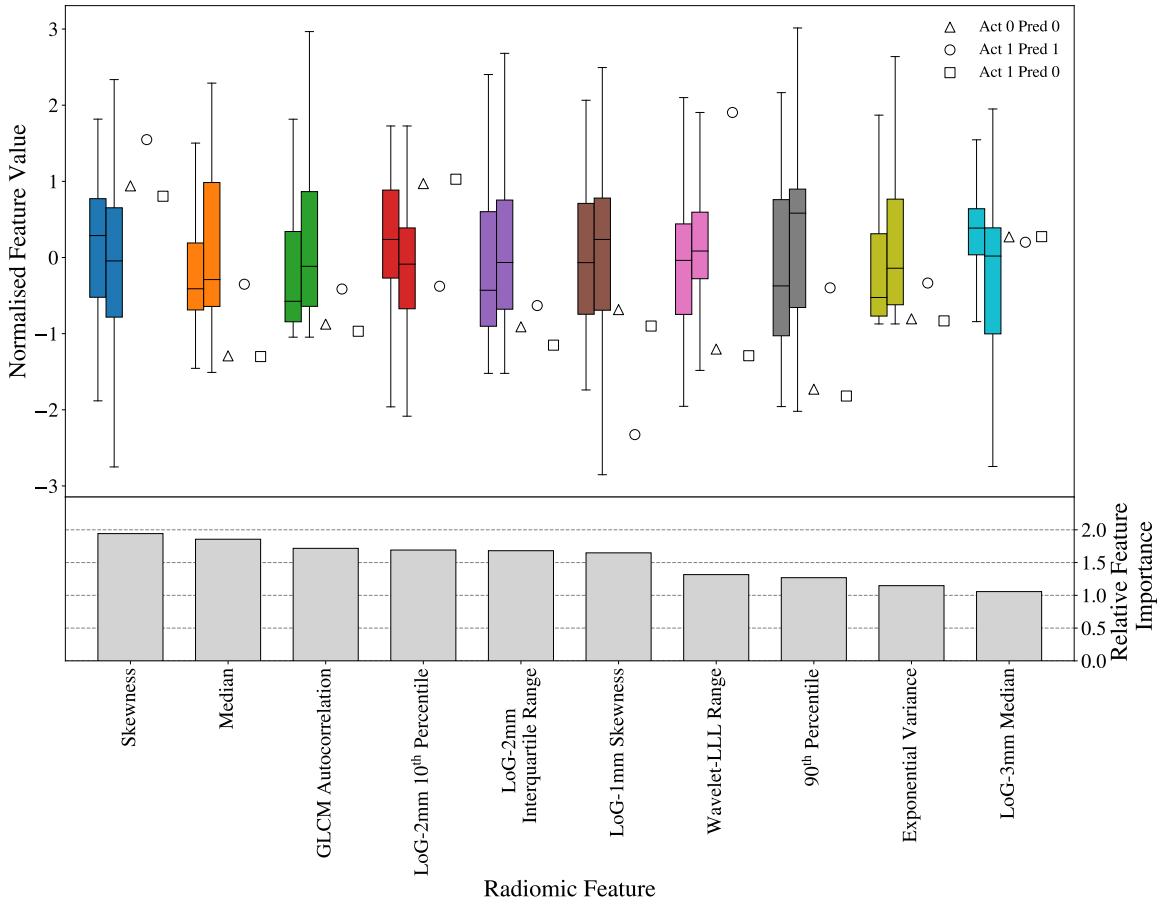


Fig. 7.17 The boxplots show how the two classes are differentiated by each of the ten most important radiomic features for the LUNA dataset. □, ○, and △ are three example cases. By comparing their feature values to the boxplots we can deduce which features are important to their classification. Below, the bars represent the feature importance (defined as the SVM coefficient) of each of the ten features. Class 0 are negative cases and class 1 are positive cases

Quantifying the performances, the original model achieved an accuracy of 0.85 using 76 radiomic features. Using only the top ten features reduced the accuracy considerably, to 0.75. Replacing these features with interpretable alternatives reduced the accuracy further, to 0.69. Using the original model's predictions as ground truth, the top ten features achieved an accuracy of 0.76, whereas the interpretable features gave an accuracy of 0.70.

Finally, interpreting the mediastinal radiomic model (Figure 7.19) we see that the majority of the top ten features are LoG-based and not easy to understand. Most of the features are CT-based, with only three PET-based features. Again, no single feature can distinguish the two classes. The □ case was misclassified as negative. Several of its features are indicative of the negative class (e.g. high PET LoG-3mm Skewness and low CT Wavelet-LLL Variance), explaining this result.

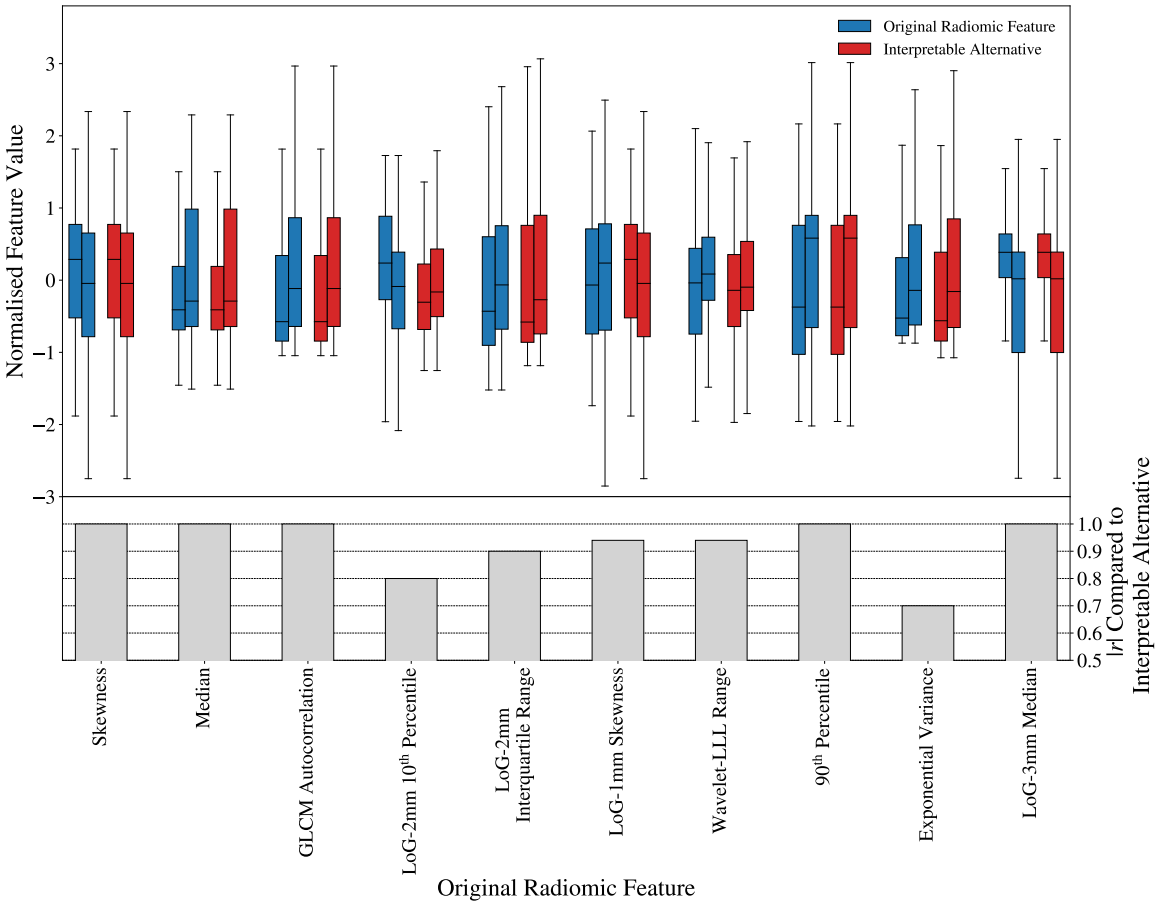


Fig. 7.18 The boxplots show how the original radiomic features and their more interpretable alternatives compare for the LUNA dataset, separately for the two classes. Below, the bars show the magnitude of correlation between the features ($|r|$)

Table 7.4 More interpretable alternatives to the original radiomic features used in the radiomic model for the LUNA dataset with their corresponding correlations. Cases where the original feature was interpretable are left blank. Cases where there was no interpretable feature with $|r| > 0.7$ are marked as NA

Original Feature	**Interpretable Alternative**	***r***
Skewness	-	-
Median	-	-
GLCM Autocorrelation	-	
LoG-2mm 10^{th} Percentile	GLCM Contrast	-0.80
LoG-2mm Interquartile Range	Interquartile Range	0.90
LoG-1mm Skewness	Skewness	-0.94
Wavelet-LLL Range	Range	0.94
90^{th} Percentile	-	-
Exponential Variance	High Grey Level Emphasis	0.70
LoG-3mm Median	NA	NA

Correlating the features with simpler alternatives sheds a bit more light on their meaning (see Table 7.5). We see several features highly correlated with CT Variance, indicating that a higher variance within the cubes is indicative of a negative prediction. PET Kurtosis and PET Skewness relate to overall PET intensities and thus could be useful for detecting positive nodes (which have high SUV uptakes). Again, plotting the original features against the interpretable alternatives (Figure 7.20) we see that some do not match up (e.g. CT LoG-3mm Uniformity and PET LoG-3mm Skewness).

The original model (with 71 features) achieved an accuracy of 0.80. With ten features the accuracy was 0.72, and when using the interpretable alternatives the accuracy was 0.70. Using the original model's predictions as a ground truth gave an accuracy of 0.80 when training a model with the original ten features and 0.67 when using the interpretable features.

These experiments achieved a degree of success in interpreting the radiomic models. Simpler first-order features (such as those used in the LUNA model) can give clear indications of how classes are being distinguished, and individual cases can be analysed to explain their classification. However, many of the top features (particularly for the brain and mediastinal datasets) were difficult or impossible to interpret. Some could be correlated to simpler features, but this weakened their interpretive power (e.g. reducing the accuracy from 0.81 to 0.70 for the brain dataset). We could not find any studies that try to directly explain higher-order texture features, wavelet features, or LoG features. Many of these features were not designed for medical images (GLCM features were first used to classify

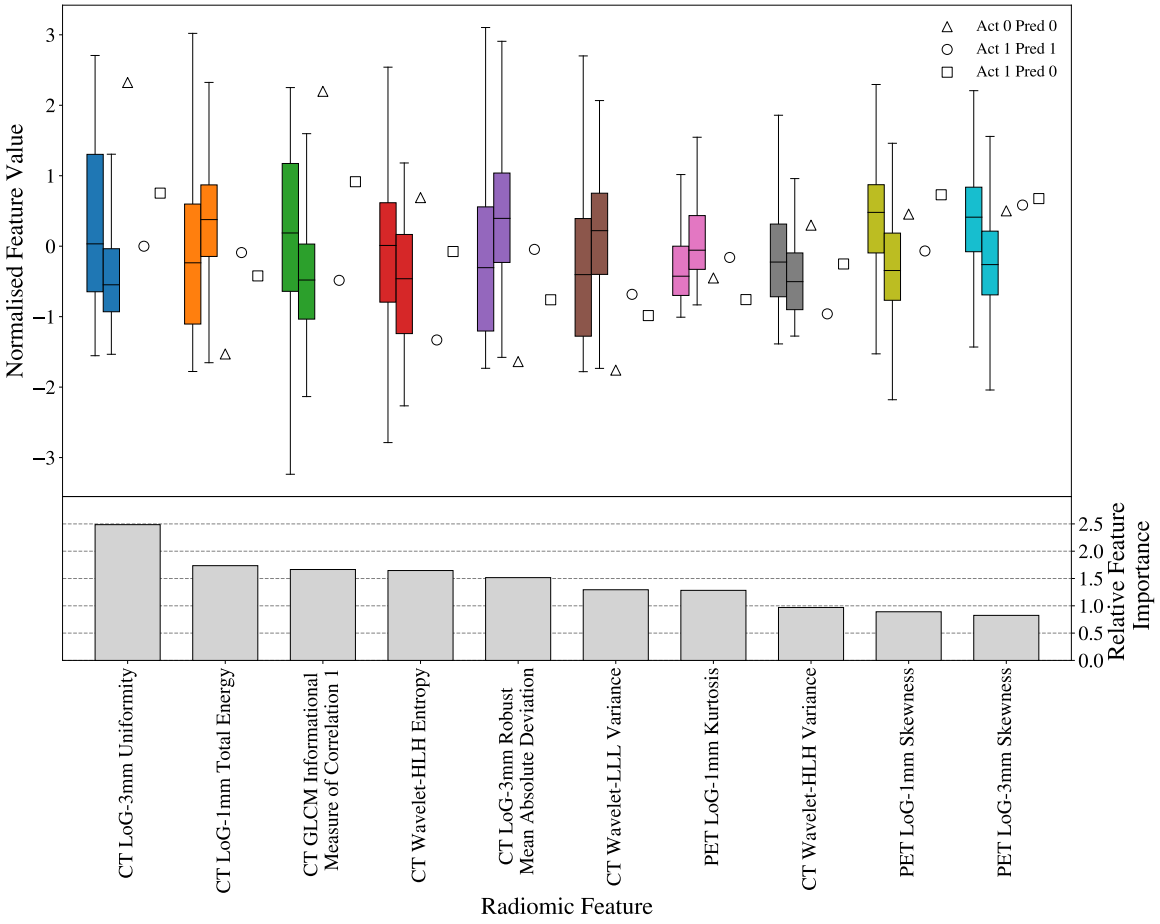


Fig. 7.19 The boxplot shows how the two classes are differentiated by each of the ten most important radiomic features for the mediastinal dataset. □, ○, and △ are three example cases. By comparing their feature values to the boxplots we can deduce which features are important to their classification. Below, the bars represent the feature importance (defined as the SVM coefficient) of each of the ten features. Class 0 are negative cases and class 1 are positive cases

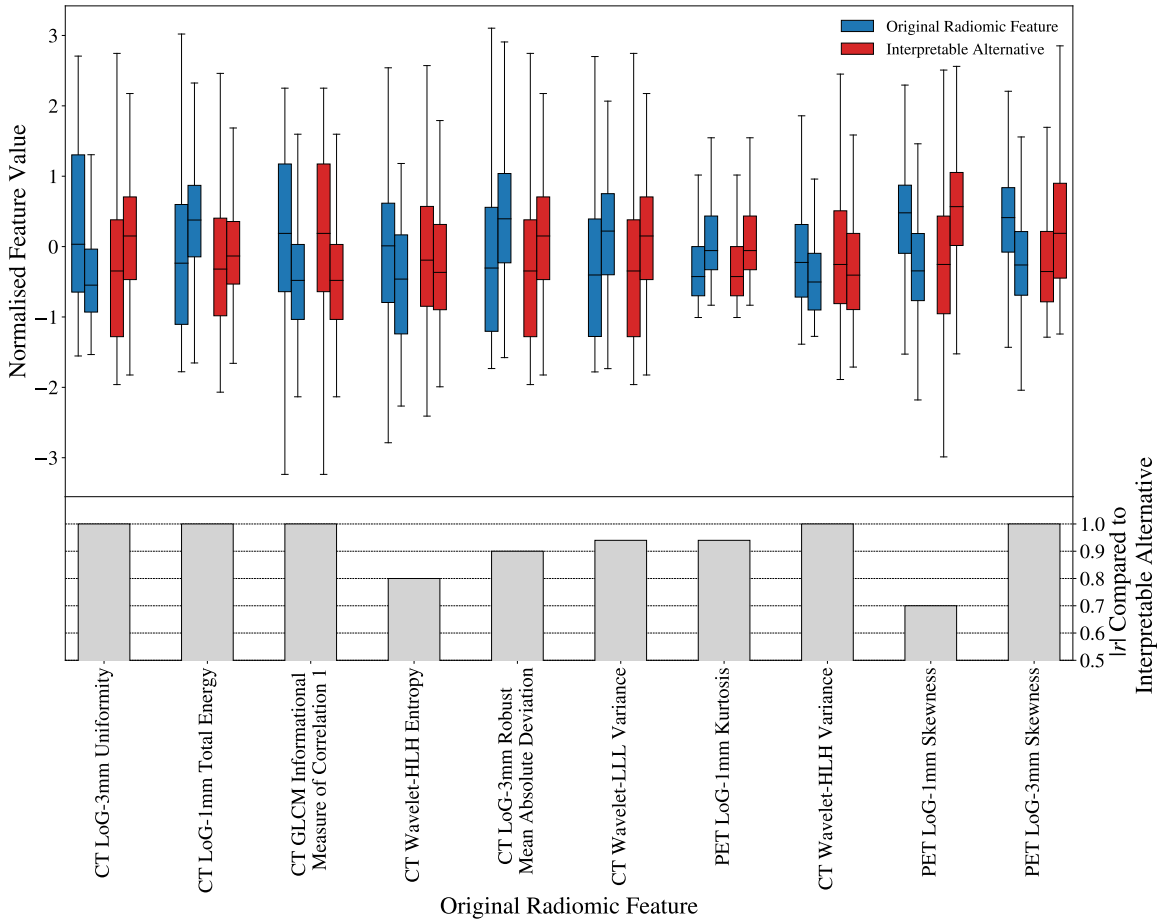


Fig. 7.20 The boxplots show how the original radiomic features and their more interpretable alternatives compare for the mediastinal dataset, separately for the two classes. Below, the bars show the magnitude of correlation between the features ($|r|$)

Table 7.5 More interpretable alternatives to the original radiomic features used in the radiomic model for the mediastinal dataset with their corresponding correlations. Cases where the original feature was interpretable are left blank. Cases where there was no interpretable feature with $|r| > 0.7$ are marked as NA

Original Feature	**Interpretable Alternative**	***r***
CT LoG-3mm Uniformity	CT Variance	-0.9
CT LoG-1mm Total Energy	CT GLCM Difference Variance	0.83
CT GLCM Informational Measure of Correlation 1	-	-
CT Wavelet-HLH Entropy	CT GLCM Difference Average	0.81
CT LoG-3mm Robust Mean Absolute Deviation	CT Variance	0.89
CT Wavelet-LLL Variance	CT Variance	0.99
PET LoG-1mm Kurtosis	NA	NA
CT Wavelet-HLH Variance	GLSZM Zone Percentage	0.83
PET LoG-1mm Skewness	PET Skewness	0.72
PET LoG-3mm Skewness	PET Kurtosis	0.78

types of sandstone [260]). Creating features specifically to identify biological functions, in collaboration with physicians, would help interpretability and, as explained in 7.5, could improve the performance. Another method of interpretation is to link the radiomic features to semantic or histological features, such as in [261] [262] [263] [44]. This can improve the confidence and utility of a radiomic feature, but this kind of insight is rare in publications. More effort needs to be made to try and understand these features so that they can be more widely used in a clinical setting.

Clearly less complex models are easier to understand. Our original radiomic models all had over 50 features, which is probably too many to easily interpret. We reduced this to the ten most important features in each case. For the brain dataset this did not greatly impact the performance (0.85 vs 0.81), but the accuracies of the LUNA and mediastinal models dropped considerably (0.85 vs 0.75 and 0.80 vs 0.72). This could indicate that these problems are more complex and thus require more features, but reducing their number would be prudent if possible. We used a crude feature reduction technique, simply choosing the top features by SVM coefficient, but using a more careful approach could reduce the number of features without impacting the performance. Perhaps using a different classifier (e.g. an SVM with a non-linear kernel) could achieve this. This highlights the importance of careful feature reduction, not just to improve performance, but also to improve interpretability.

It is hard to quantifiably compare the interpretability of the CNN-based and radiomic-based models. The mathematical simplicity of our radiomic-based linear SVM models is inherently more trustworthy than the heuristic interpretations given by the CNN saliency maps. However, more work needs to be done to make sense of the radiomic features themselves and bridge the gap between these abstract features and the underlying biological mechanisms they are identifying.

7.10 Using Radiomic Features to Create and Interpret Deep Features

In other domains studies have tried to explain the inner workings of CNNs in terms of user-defined semantic concepts. [146] linked specific filters within a CNN to body parts in natural images. [148] trained a second *explainer network* to explain the rationale behind CNN decisions in terms of understandable image features such as facial expressions. [147] tested how linear combinations of different layer outputs can be used to represent semantic features such as stripiness or colour. They then applied this to the problem of predicting diabetic retinopathy from retinal fundus images and succeeded in relating some of the CNN's outputs to diagnostic concepts.

In this section we test a novel method of interpreting our CNN models by trying to directly recreate them using radiomic features, allowing us to translate the more opaque deep features into more understandable handcrafted radiomic features. We then evaluate these models to see how faithfully they can reproduce the original CNN models' results.

7.10.1 Method

For a CNN at large N for each dataset ($N_{\text{patient}} = 190$ for the brain dataset, $N = 1800$ for the LUNA dataset, and $N_{\text{patient}} = 80$ for the mediastinal dataset) we selected the five most important deep features after feature reduction using the SVM coefficients (five features was seen as a balance between under and overkill of data). We then trained RFRs to predict these deep features using the radiomic features. To train the RFRs a similar method was used as when training the SVMs in the above experiments. The radiomic features were rescaled to have zero mean and unit variance. They were then reduced using univariate analysis (using R^2 as it is a regression problem), before strongly correlated features were removed. Again, to ensure the best-performing parameters were chosen, in each case we created models selecting the [10, 20, 40, 60, 80, 100, 120, 140] best features at the univariate stage and for each of these removed features with $|r|$ greater than [0.95, 0.9, 0.8, 0.7, 0.6, 0.5]. 100 trees were used

in each forest, MSE was used as the splitting criterion, and no limit was put on the depth of the trees (other parameter combinations were tested). After training the RFRs on the training set, they were used to predict the deep features from the validation set. The results were plotted against the actual deep feature values and evaluated using the Pearson correlation r.

We then wanted to see if we could replace the deep features with these radiomic features. We trained SVMs replacing each of the five deep features with the top one, three, five, and ten radiomic features from the corresponding RFR model (using the feature importance metric to rank the features). Note that some features were repeated, meaning that the resulting models did not always have ($5 \times$ No. radiomic replacements) features. Features were rescaled to have zero mean and unit variance and we used a linear SVM kernel with $C = 1$. We also evaluated how closely these sets of replacement features could mimic the results of the original deep features by training SVMs with the original deep feature predictions as the ground truth.

7.10.2 Results and Discussion

The actual vs predicted feature values for the top five deep features for each dataset are plotted in Figure 7.21. We see that for the brain dataset we were able to accurately recreate the deep features using radiomic features, with $|r|$ scores between 0.84 and 0.90. We were not able to create the LUNA deep features as accurately, with $|r|$ values ranging from 0.71 to 0.85. The predictions of the mediastinal deep features were very poor, with $|r|$ scores between 0.44 and 0.64.

In Figure 7.22 we plot the actual and predicted features for the three classes for the brain dataset. We can see that the predicted features closely match the original deep features, retaining the distributions. Table 7.6 lists the performances of the different models. They show that using only the top five deep features we can achieve a performance very close to the full model (0.84 vs 0.86) and can accurately match the original model's results (0.95). Replacing the deep features with an unlimited number of radiomic features, we achieve an accuracy of 0.86 against the ground truth and 0.90 against the original model. However, this is a lot of features (each RFR used between 23 and 59 radiomic features), so to reduce the complexity we then limited the number of radiomic features per deep feature. Using only one radiomic feature per deep feature we achieved a reasonable accuracy and faithfulness to the original model (0.71 and 0.77 respectively). Increasing the number of radiomic features per deep feature improved the performance as expected, and with five radiomic features per deep feature we could reproduce the original results with an accuracy of 0.88.

Figure 7.23 compares the original to the recreated deep features for the LUNA dataset. The correlations are not as strong, but the distributions of the classes are still maintained

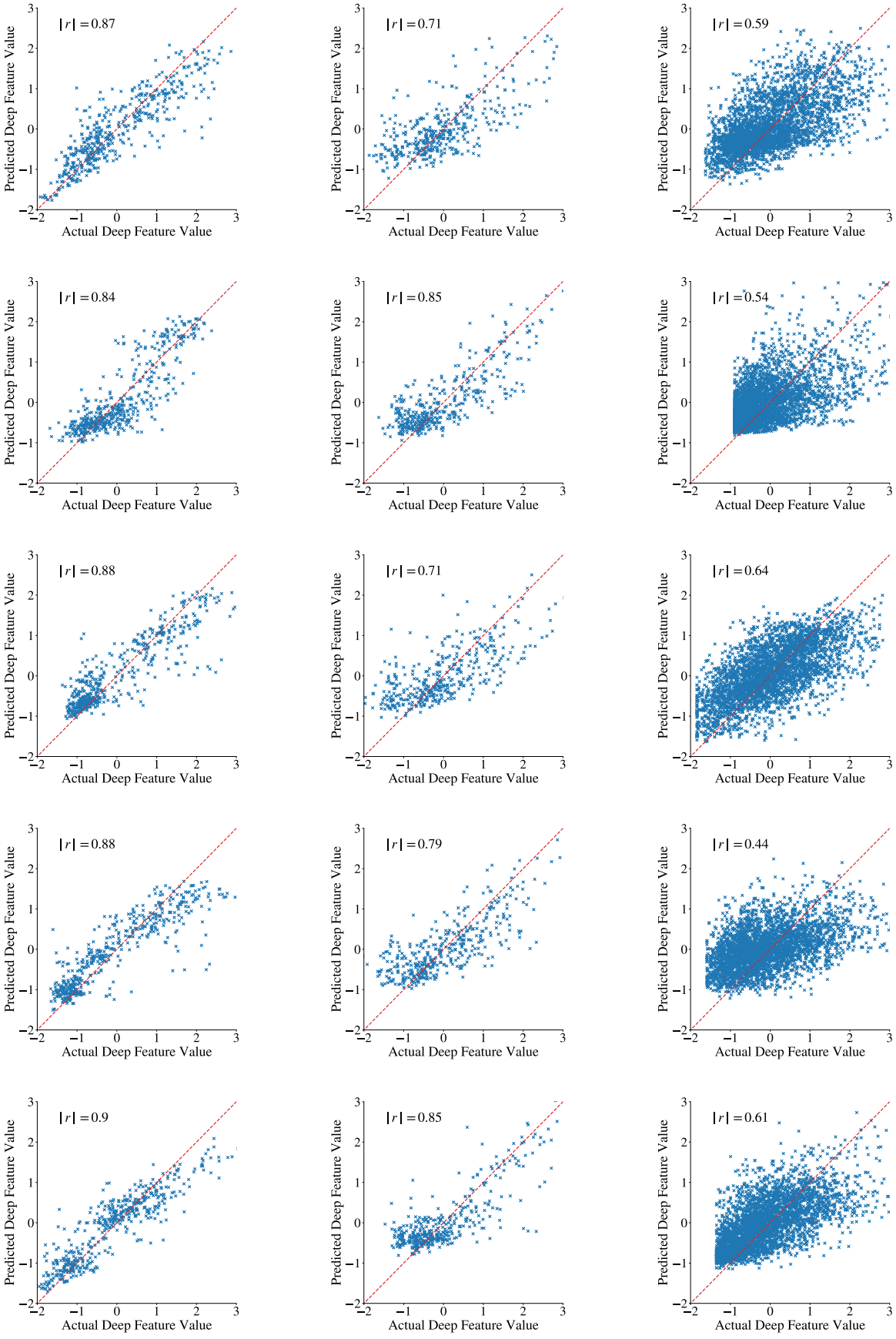


Fig. 7.21 Actual values of the top five deep features vs values predicted using an RFR trained on the radiomic features for the three datasets

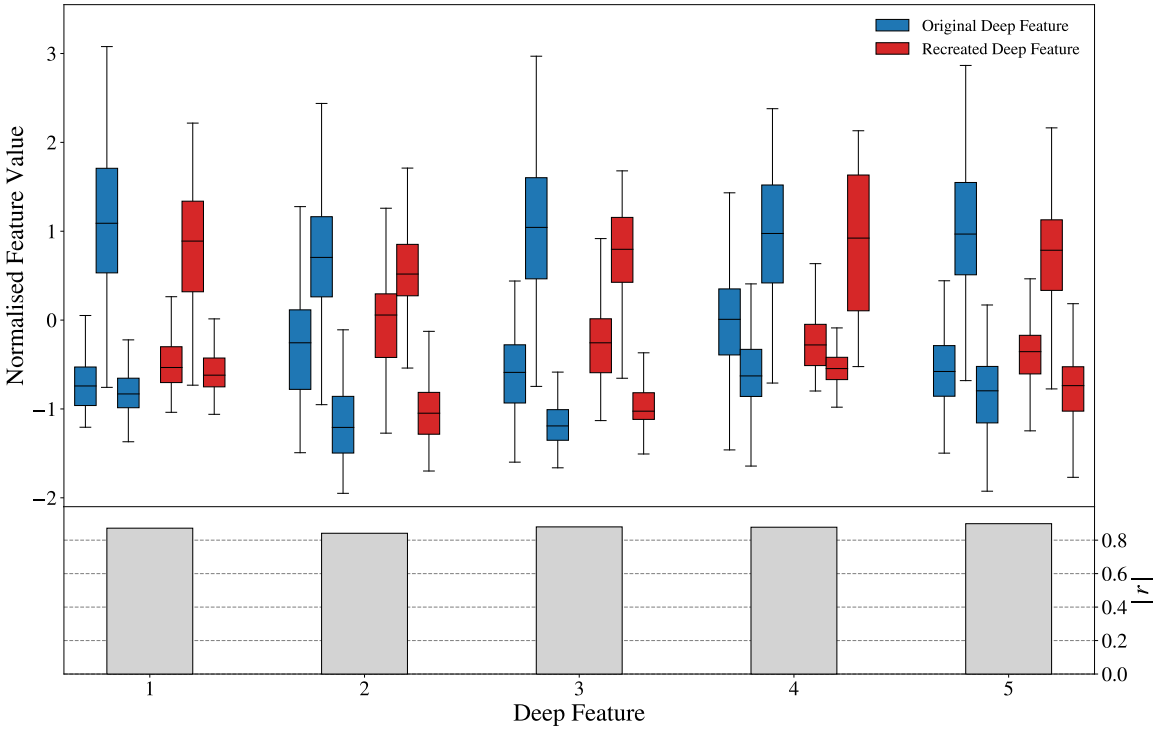


Fig. 7.22 The boxplots show a comparison of the distributions of the top five deep features and their counterparts created using an RFR for the brain dataset. Below, the bars show the correlation between the deep and recreated features

Table 7.6 Accuracy of the original CNN-based model, a model using only the top five deep features, and models replacing these deep features with a number of radiomic features, for the brain dataset. Each model was separately trained and evaluated using the ground truth labels and the predictions of the original CNN-based model

Model	**Accuracy (Ground Truth Labels)**	**Accuracy (Original Model Predictions)**
CNN-based model	0.86	-
Model with five deep features	0.84	0.95
Unlimited radiomic features/deep feature	0.86	0.90
One radiomic features/deep feature	0.71	0.77
Three radiomic features/deep feature	0.81	0.84
Five radiomic features/deep feature	0.81	0.88
Ten radiomic features/deep feature	0.85	0.88

reasonably well. Table 7.7 lists the performances of the different models. Contrary to the brain dataset, limiting the model to five deep features reduces the performance considerably (0.81 vs 0.91). Replacing each of the deep features with an unlimited number of radiomic features gives an accuracy almost as good as the original model (0.87), but again this involves a lot of features so is not that interpretable. Limiting ourselves to only three radiomic features per deep feature gives an accuracy of 0.72 relative to the ground truth and 0.77 relative to the original model's predictions. This level of agreement would allow us to gain some meaning from the deep features.

Finally, Figure 7.24 compares the top five original and recreated deep features for the mediastinal dataset. The correlations were low for these features, showing that it was not possible to accurately recreate the deep features using radiomic features. Table 7.8 shows the performances of the different trained models. We see that, as with the LUNA dataset, limiting ourselves to five deep features cut the performance drastically, from 0.87 to 0.65. Despite this, when replacing these deep features with radiomic features we achieved reasonable performances, with an accuracy of 0.77 vs the ground truth labels and 0.77 vs the original model's predictions when replacing each with three radiomic features. This shows that although the radiomic features were not successful at recreating the deep features, they were more successful at recreating the overall model's outputs.

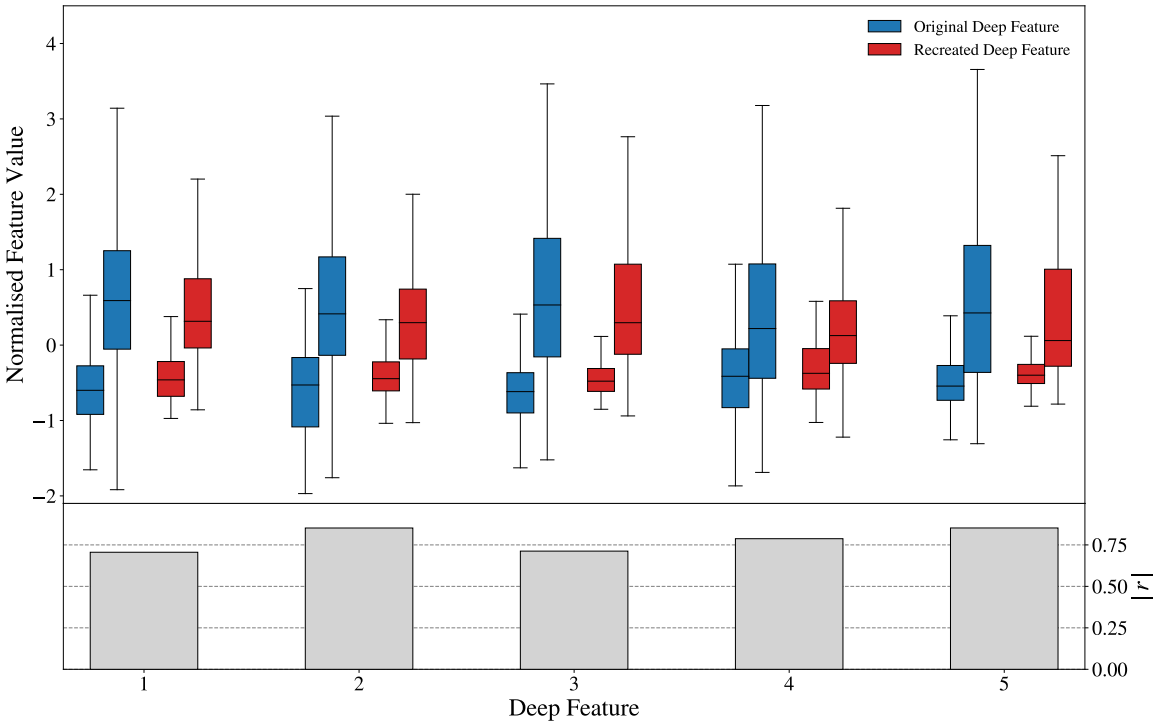


Fig. 7.23 The boxplots show a comparison of the distributions of the top five deep features and their counterparts created using an RFR for the LUNA dataset. Below, the bars show the correlation between the deep and recreated features

Table 7.7 Accuracy of the original CNN-based model, a model using only the top five deep features, and models replacing these deep features with a number of radiomic features, for the LUNA dataset. Each model was separately trained and evaluated using the ground truth labels and the predictions of the original CNN-based model

Model	Accuracy (Ground Truth Labels)	Accuracy (Original Model Predictions)
CNN-based model	0.91	-
Model with five deep features	0.81	0.86
Unlimited radiomic features/deep feature	0.87	0.85
One radiomic features/deep feature	0.72	0.70
Three radiomic features/deep feature	0.72	0.77
Five radiomic features/deep feature	0.78	0.80
Ten radiomic features/deep feature	0.84	0.85

Table 7.8 Accuracy of the original CNN-based model, a model using only the top five deep features, and various models replacing these deep features with a number of radiomic features, for the mediastinal dataset. Each model was separately trained and evaluated using the ground truth labels and the predictions of the original CNN-based model

Model	Accuracy (Ground Truth Labels)	Accuracy (Original Model Predictions)
CNN-based model	0.87	-
Model with five deep features	0.65	0.66
Unlimited radiomic features/deep feature	0.80	0.84
One radiomic features/deep feature	0.73	0.74
Three radiomic features/deep feature	0.77	0.77
Five radiomic features/deep feature	0.76	0.78
Ten radiomic features/deep feature	0.80	0.82

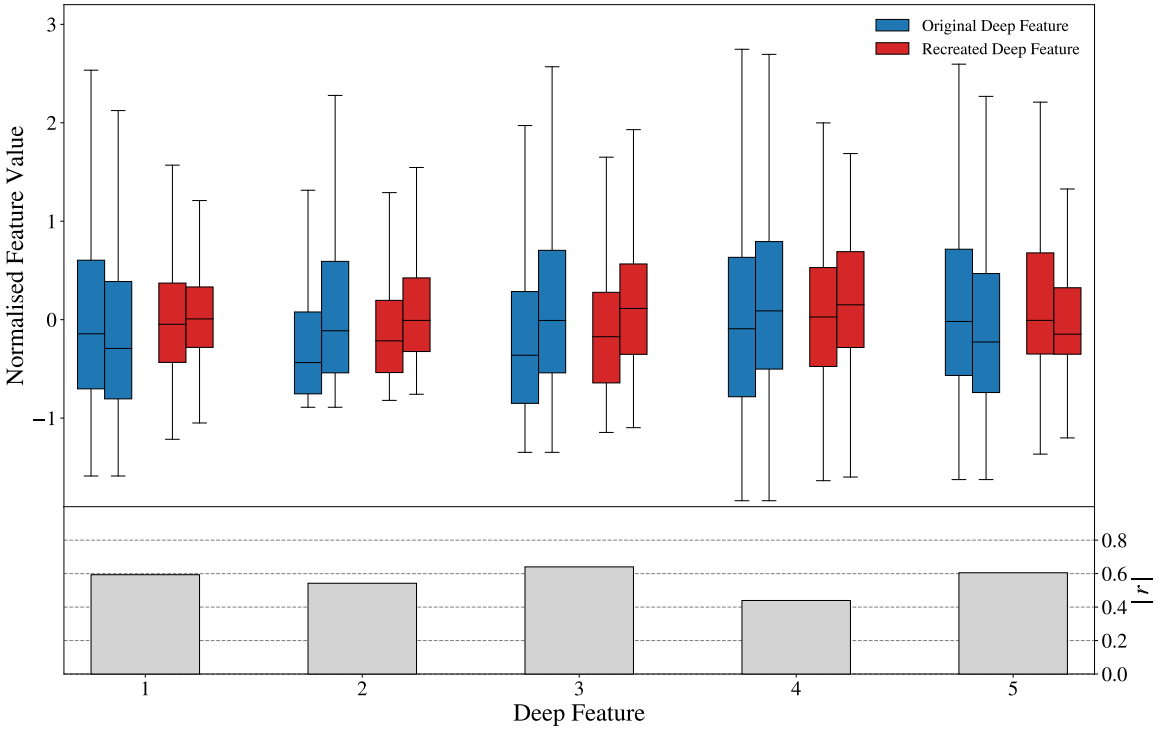


Fig. 7.24 The boxplots show a comparison of the distributions of the top five deep features and their counterparts created using an RFR for the mediastinal dataset. Below, the bars show the correlation between the deep and recreated features

These results show that the ability of radiomic features to mimic deep features significantly depends on the dataset, with radiomic features capable of recreating deep features to a high accuracy for the brain dataset, but not at all for the mediastinal dataset. This is the mirror of the work in [264], which showed that some 2D and 3D radiomic features could be accurately created using CNNs whereas others could not. Replacing each of the top five deep features with three radiomic features, we achieved accuracies of >0.75 for all three datasets (vs the original model). This demonstrates that this replacement does allow us to gain an understanding of the underlying CNN mechanism. Using more radiomic features gave higher accuracies, but at the cost of more complicated models that would be harder to interpret. As in the previous section we used SVM feature importance to select these features, but a more sophisticated selection method could improve the results. Further work is needed to understand what makes deep features from different models or datasets amenable (or not) to replacement by radiomic features. By doing this we could better understand what each deep feature is representing in the image.

7.11 Are CNNs Stable to Retraining?

Previous studies have shown CNNs to have a concerning lack of stability to retraining [234] [265]. In this section, we examine to what degree this uncertainty leads to changes in patient classification upon retraining. We use Grad-CAM and occlusion maps to see whether the focus of the CNNs is changing upon retraining. To see whether this problem is more acute with less data, we run experiments at various values of N.

7.11.1 Method

For each dataset we trained three CNNs with identical parameters and with the same training and validation sets. The only differences between the retrained CNNs were the random initialisation of weights (we used the glorot initialisation [266]) and the order of the training data (the data were randomly shuffled). We first compared the overall accuracies and characterised the accuracy when using a majority vote. We then compared the image-by-image results, plotting Venn diagrams showing the overlap of predictions as in 7.7. Finally, we created Grad-CAM and occlusion maps (only occlusion maps for the mediastinal dataset) to see whether the CNNs were using the same parts of the images (and therefore the same information) to reach decisions. To ensure direct comparison was valid, the gradients of the actual class (rather than the predicted class) were used for Grad-CAM calculations. We performed these experiments for small, medium, and large N ($N_{patient}$=30, 90, and 190 for

the brain dataset, N=80, 200, 1800 for the LUNA dataset, and $N_{patient}$=10, 40, and 80 for the mediastinal dataset).

To quantitatively compare two Grad-CAM maps we used two methods; DICE score and Spearman's rank correlation coefficient (SRCC) [267], as used in other comparisons [268] [269]. To use the DICE score we thresholded each map at 0.5 and calculated the DICE score between the two images. The threshold itself was arbitrary; we were interested in comparative rather than absolute results. To calculate the SRCC each pixel or voxel in a heatmap was ranked by its importance, and this ranking was correlated with the equivalent ranking from the second heatmap. This correlation gives a measure of how similar two heatmaps are.

7.11.2 Results

Figure 7.25 shows the overlap of image-by-image results for three retrained CNNs for the three datasets, for small, medium, and large N. Table 7.9 shows the individual and majority voting accuracies of each of the retrained CNNs in all cases. From the table we see that the overall performances of the retrained models are very stable, being slightly more stable at high N. However, this belies deeper disagreements at the patient level as the Venn diagrams reveal. At high N there was disagreement in 7.4% of cases for the brain dataset, 17.0% of cases for the LUNA dataset, and 2.2% of cases for the mediastinal dataset. At low N the differences were far greater, for 10.8% of cases for the brain dataset, 22.8% for the LUNA dataset, and 43.0% of cases for the mediastinal dataset.

Figures 7.26 to 7.28 show Grad-CAM and occlusion maps for three brain images. In each case the maps for the three retrained CNNs at small, medium, and large N are shown. We see that even when the predictions are all correct (as in Figures 7.26 and 7.27), the saliency maps can differ considerably. For $N_{patient} = 190$ in Figure 7.26 both saliency maps for CNN_3 are completely different to those from CNN_1 and CNN_2. Similar examples can be seen for all three images at low and high N. This could be a shortcoming of the saliency maps (as discussed in 7.8), however, given the image-by-image results in Figure 7.25, this could also be because of deeper differences in the learned weights of the CNNs. Several of the Grad-CAM maps in Figure 7.28 are similar, despite differing predictions. This again shows a limitation of the interpretation.

Figures 7.29 to 7.31 illustrate three examples from the LUNA dataset. These are 2D slices of 3D saliency maps so do not give the full picture, but we tried to use examples that were as informative as possible and representative of the overall trends. The CNNs trained with $N = 200$ and $N = 1800$ in Figure 7.29 give the same predictions and broadly similar saliency maps, which is reassuring. At $N = 80$ the predictions are different, probably because there

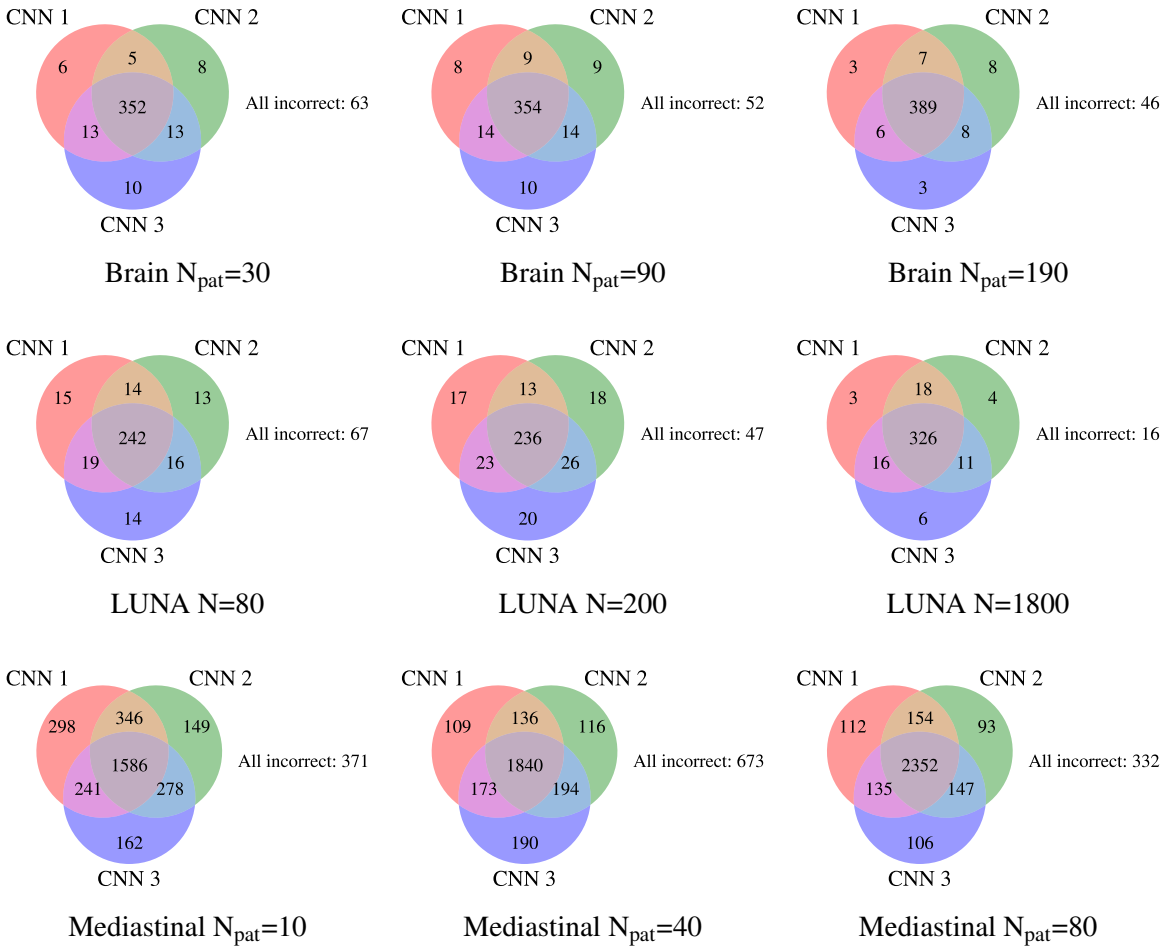


Fig. 7.25 Venn diagrams showing the overlap of image-level results for retrained CNNs at three different dataset sizes for all three datasets. $N_{pat} \equiv N_{patient}$

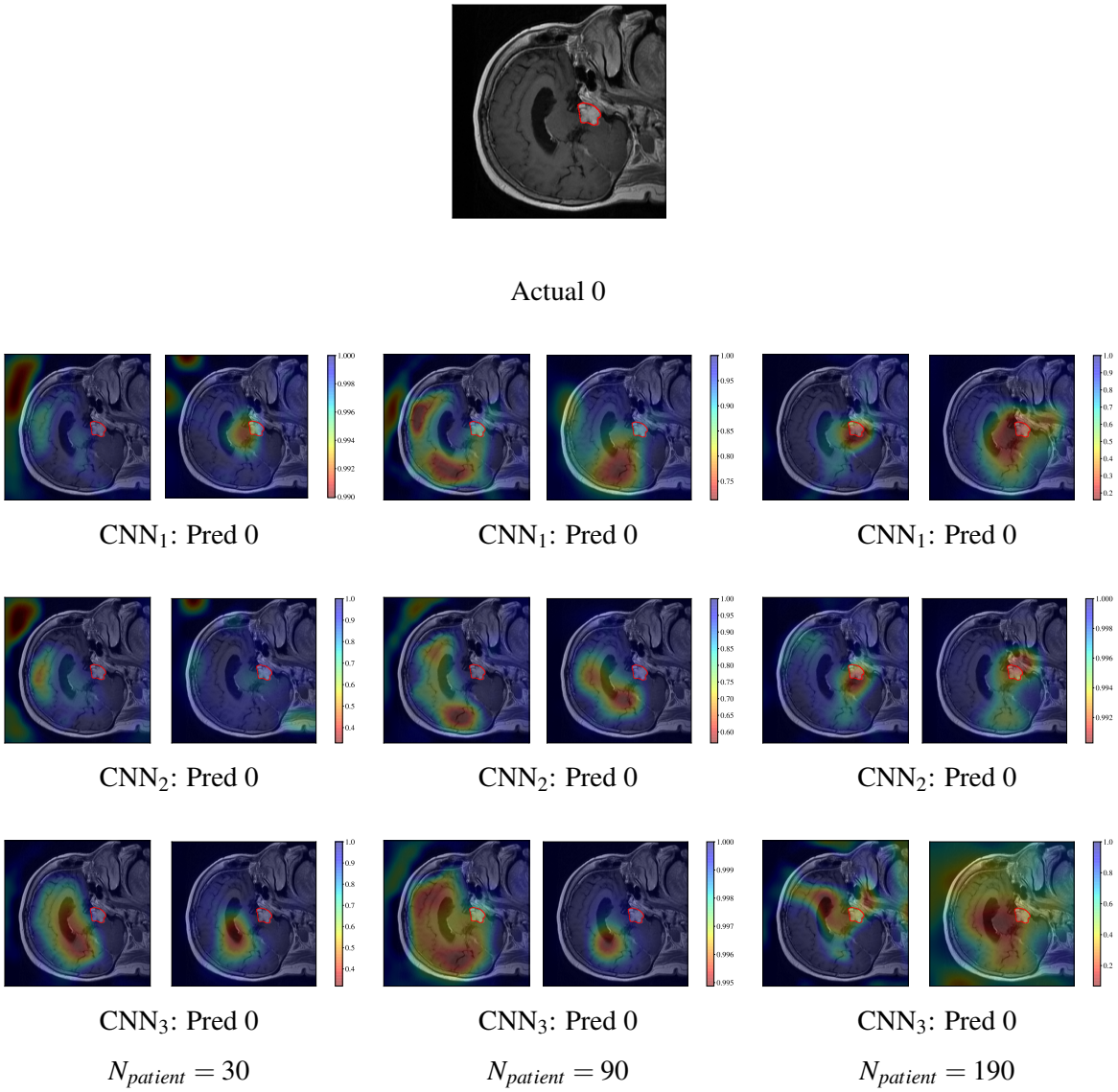
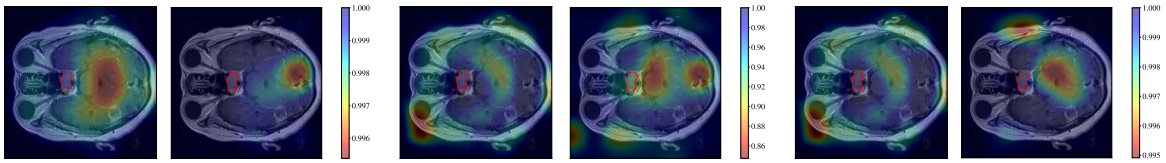
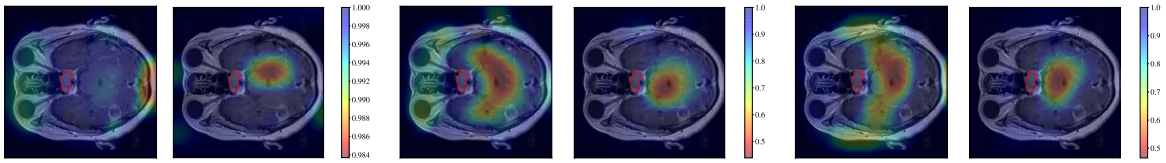


Fig. 7.26 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large $N_{patient}$, for the brain dataset. The red outline shows the tumour delineation. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown. Class 0 are meningiomas, class 1 are gliomas, and class 2 are pituitary tumours

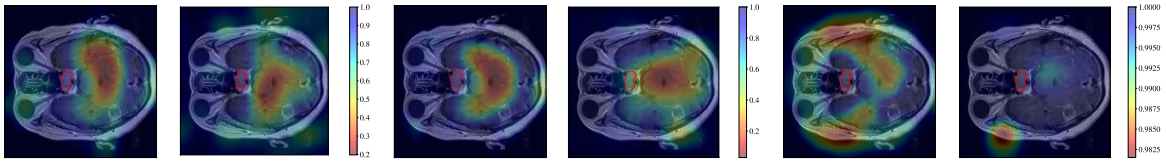
Actual 2



CNN_1: Pred 2 CNN_1: Pred 2 CNN_1: Pred 2



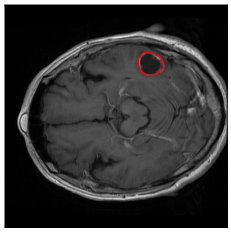
CNN_2: Pred 2 CNN_2: Pred 2 CNN_2: Pred 2



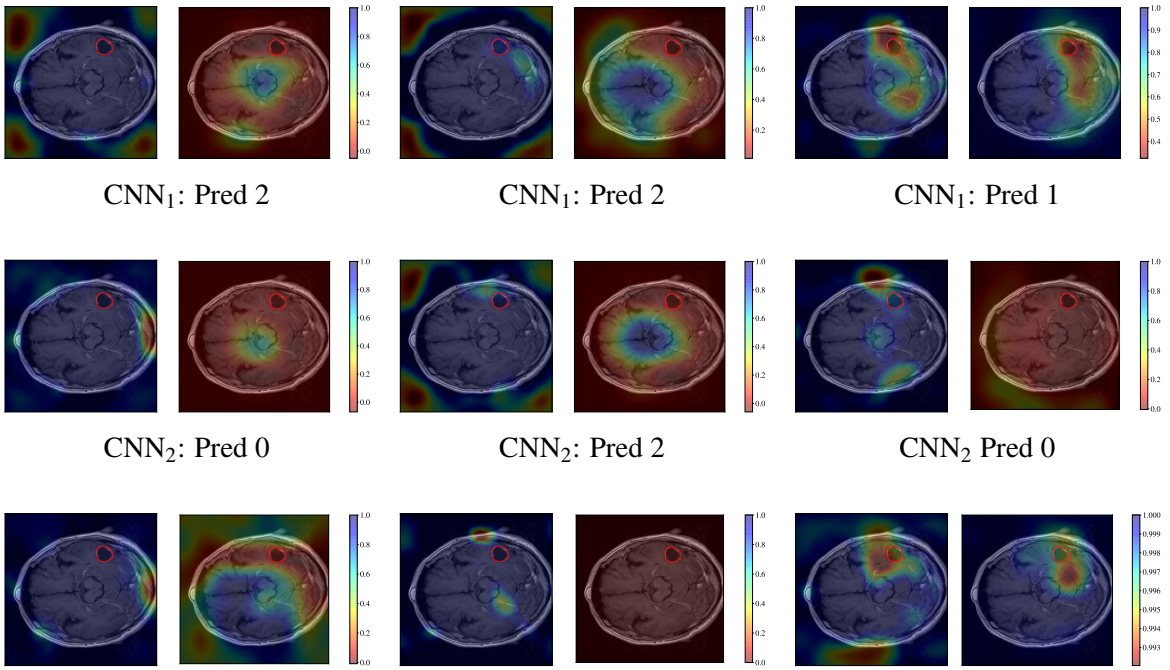
CNN_3: Pred 2 CNN_3: Pred 2 CNN_3: Pred 2

$N_{patient} = 30$ $N_{patient} = 90$ $N_{patient} = 190$

Fig. 7.27 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large $N_{patient}$, for the brain dataset. The red outline shows the tumour delineation. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown. Class 0 are meningiomas, class 1 are gliomas, and class 2 are pituitary tumours



Actual 1



CNN_1: Pred 2 CNN_1: Pred 2 CNN_1: Pred 1

CNN_2: Pred 0 CNN_2: Pred 2 CNN_2 Pred 0

CNN_3: Pred 0 CNN_3: Pred 0 CNN_3: Pred 1

$N_{patient} = 30$ $N_{patient} = 90$ $N_{patient} = 190$

Fig. 7.28 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large $N_{patient}$, for the brain dataset. The red outline shows the tumour delineation. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown. Class 0 are meningiomas, class 1 are gliomas, and class 2 are pituitary tumours

Table 7.9 Overall performances for retrained CNNs for the three datasets at small, medium, and large N ($N_{patient}$ for the brain and mediastinal datasets). The performances using majority voting are also shown

Dataset	***N***	**Accuracy**			
		CNN_1	CNN_2	CNN_3	Majority Vote
Brain	30	0.80	0.80	0.83	0.81
	90	0.82	0.82	0.83	0.83
	190	0.86	0.88	0.86	0.87
LUNA	80	0.73	0.71	0.73	0.73
	200	0.72	0.73	0.76	0.75
	1800	0.91	0.90	0.90	0.93
Mediastinal	10	0.72	0.69	0.60	0.71
	40	0.66	0.67	0.70	0.68
	80	0.80	0.80	0.81	0.81

was not enough training data. In Figure 7.30 at $N = 200$, CNN_2 is focusing on a different area of the image, which seems to have lead to an incorrect prediction. Similarly, in Figure 7.31 at $N = 1800$, CNN_3 is using a completely different area compared to the other CNNs.

Finally, Figures 7.32 to 7.34 show three examples from the mediastinal dataset. Without the Grad-CAM maps we do not get as much insight, but in Figure 7.32 at $N_{patient} = 80$ we see that the occlusion map from CNN_3 is different to those from CNN_1 and CNN_2. In Figure 7.34 for $N_{patient} = 80$ we see that all three occlusion maps are similar despite a different prediction for CNN_3.

We quantified the differences in Grad-CAM maps across the whole validation sets using the DICE and SRCC scores. Results are shown in Table 7.10. To contextualise these results Figure 7.35 gives some example images and their corresponding DICE and SRCC scores. The metrics are reassuringly consistent, both showing the same trends. They show that there are significant differences between the Grad-CAM maps of retrained CNNs even at large N, confirming our findings for the individual results. Average agreement is slightly higher when N is larger for both datasets. It could be that with much larger datasets (i.e. thousands of images) these differences would decrease further, however in medical imaging this luxury of data is not usually available.

These experiments show that despite similar overall accuracies there are large differences between retrained CNNs. This difference is more marked at low N, where medical-based research often takes place, but even at large N concerning discrepancies exist. The stability of a model is especially important in the medical domain, as models need to give the same results

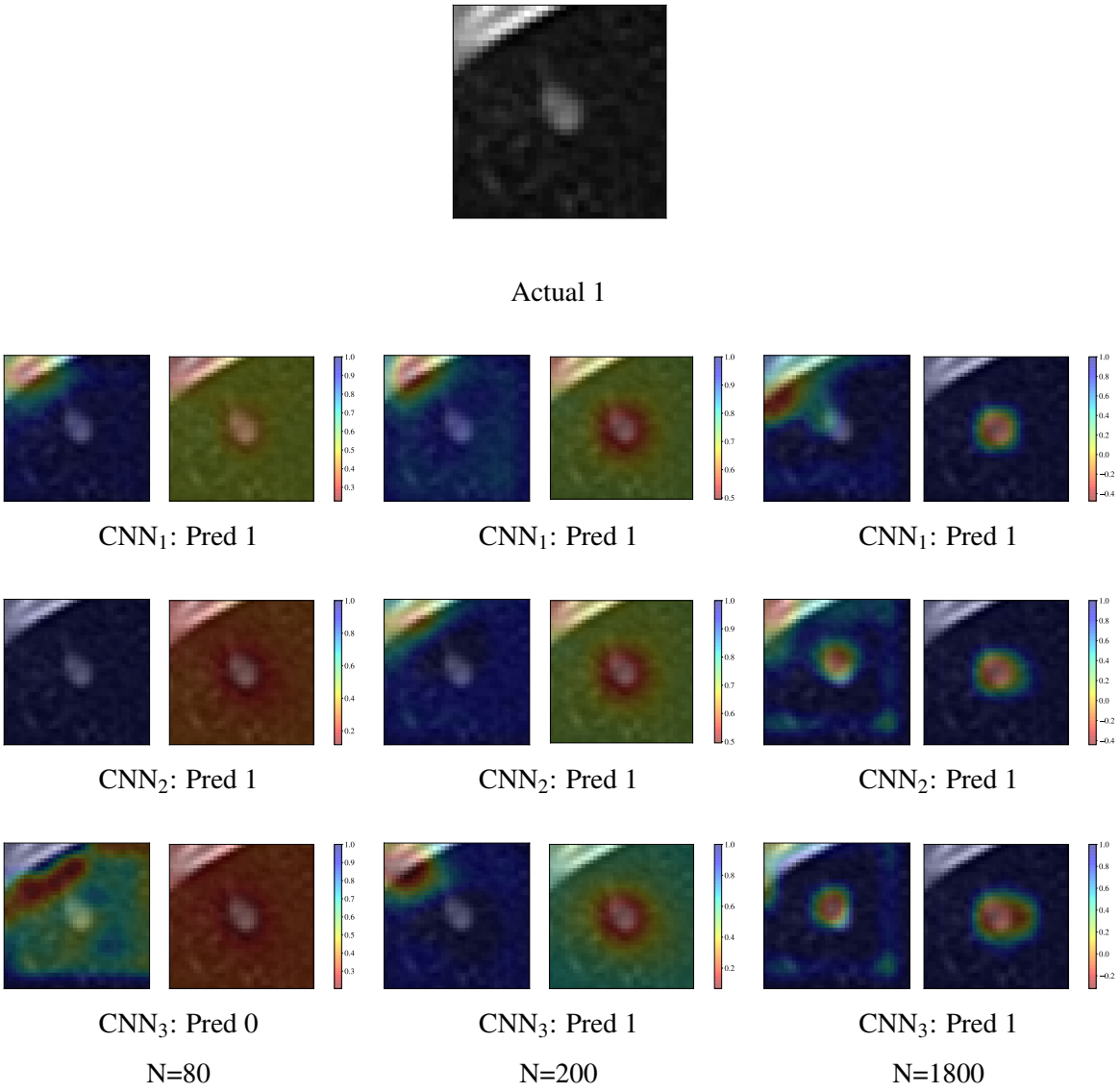


Fig. 7.29 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the LUNA dataset. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown

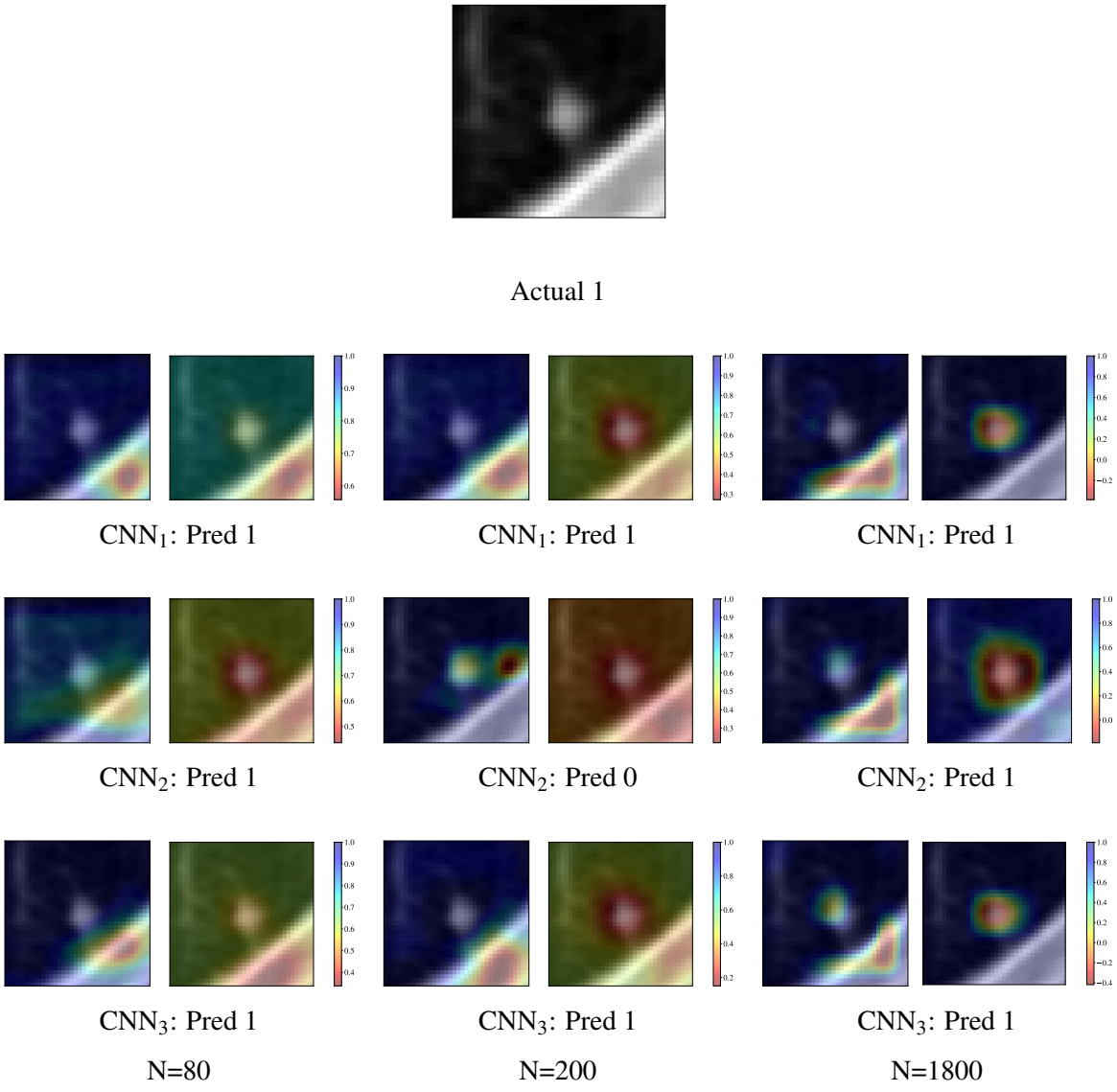


Fig. 7.30 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the LUNA dataset. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown

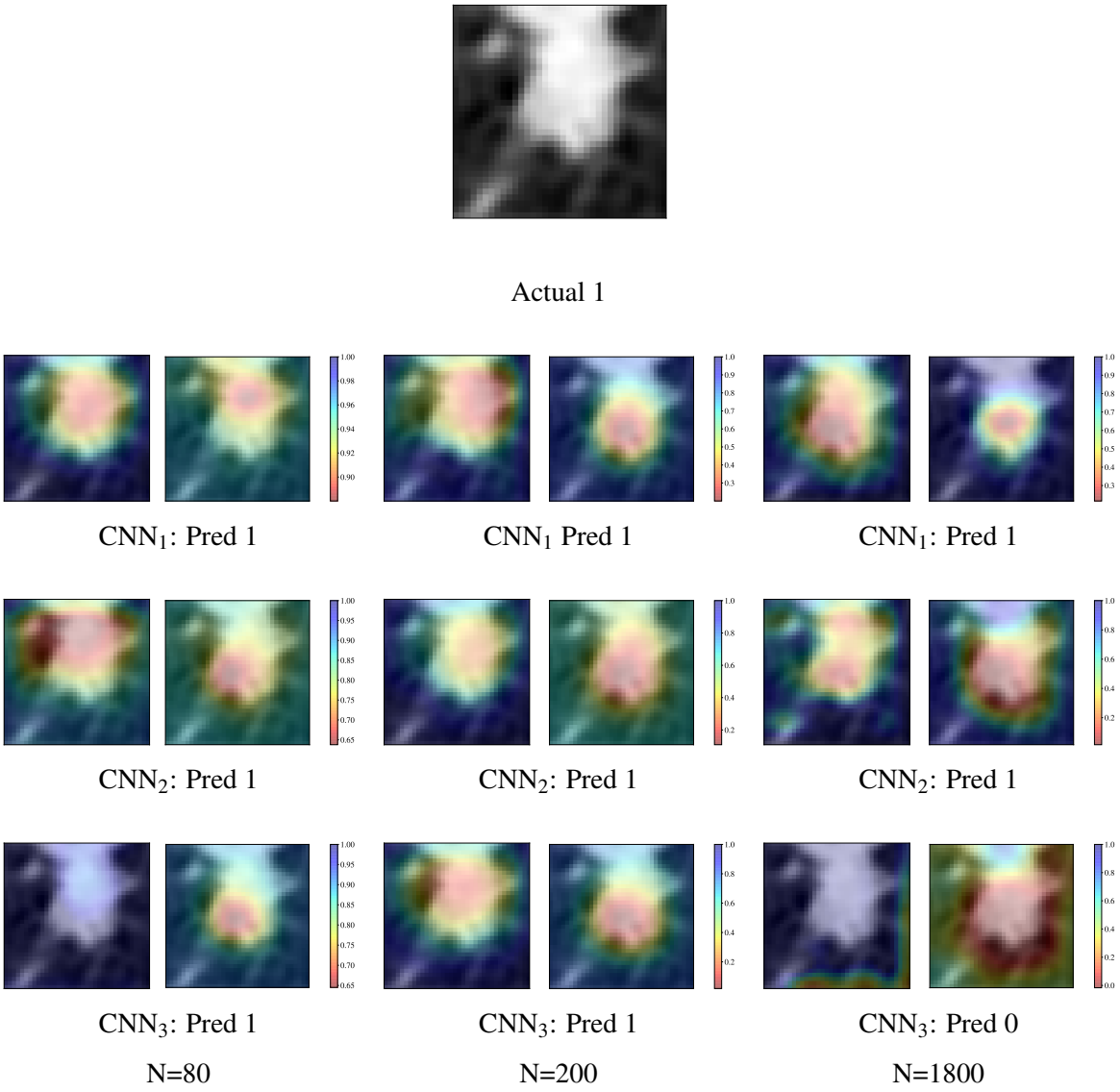
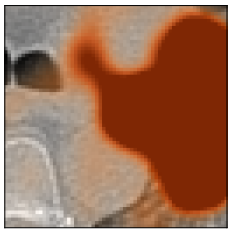
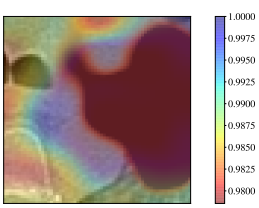


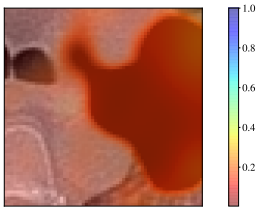
Fig. 7.31 Grad-CAM and occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the LUNA dataset. Grad-CAM maps are on the left and occlusion maps on the right. Red on the Grad-CAM maps represents high intensity (meaning the CNN was focusing on this area), whereas blue represents low intensity. For the occlusion maps red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown



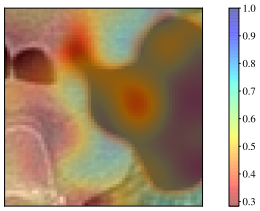
Actual 1



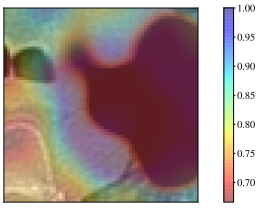
CNN_1: Pred 1



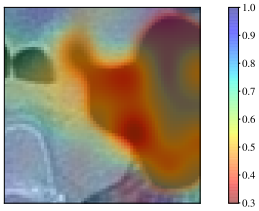
CNN_1: Pred 1



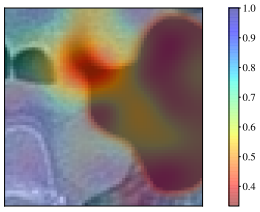
CNN_1: Pred 1



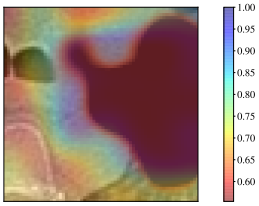
CNN_2: Pred 1



CNN_2: Pred 1

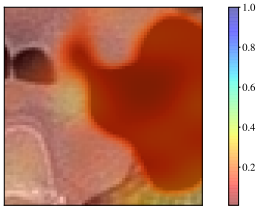


CNN_2: Pred 1



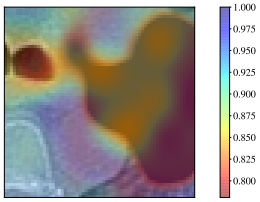
CNN_3: Pred 1

$N_{patient} = 10$



CNN_3: Pred 1

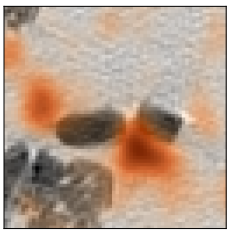
$N_{patient} = 40$



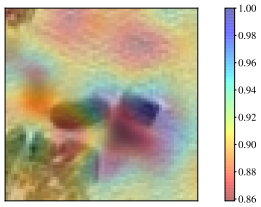
CNN_3: Pred 0

$N_{patient} = 80$

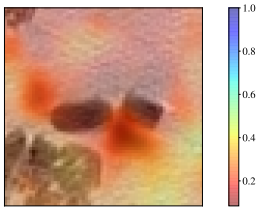
Fig. 7.32 Occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the mediastinal dataset. Red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown



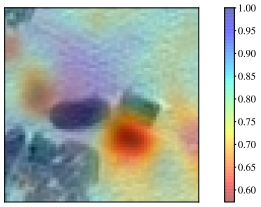
Actual 1



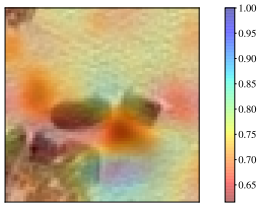
CNN_1: Pred 1



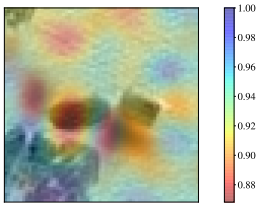
CNN_1: Pred 1



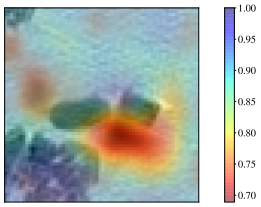
CNN_1: Pred 1



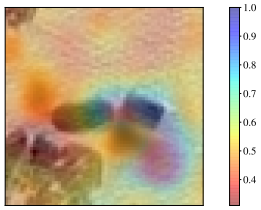
CNN_2: Pred 1



CNN_2: Pred 1

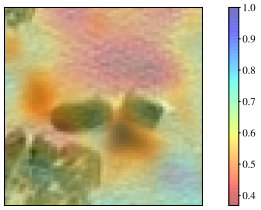


CNN_2: Pred 1



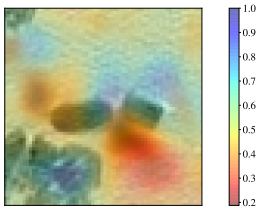
CNN_3: Pred 1

$N_{patient} = 10$



CNN_3: Pred 1

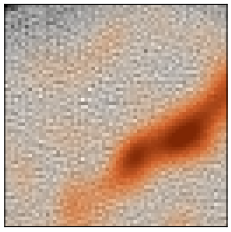
$N_{patient} = 40$



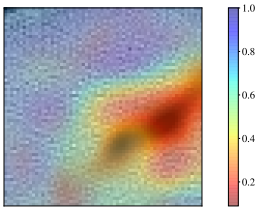
CNN_3: Pred 0

$N_{patient} = 80$

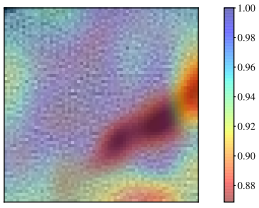
Fig. 7.33 Occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the mediastinal dataset. Red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown



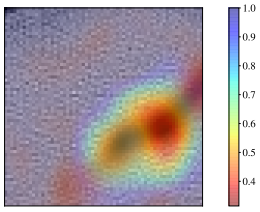
Actual 1



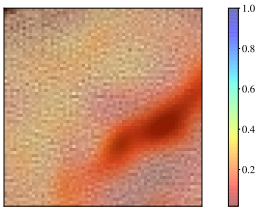
CNN_1: Pred 1



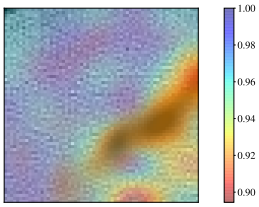
CNN_1: Pred 1



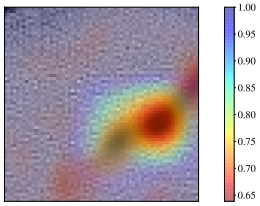
CNN_1: Pred 1



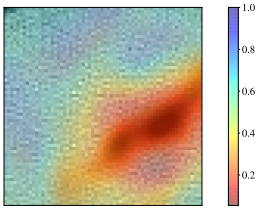
CNN_2: Pred 1



CNN_2: Pred 1

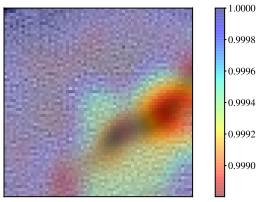


Act=1 Pred=1



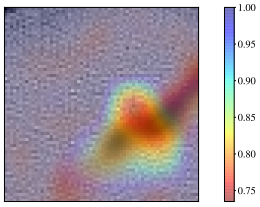
CNN_3: Pred 1

$N_{patient} = 10$



CNN_3: Pred 1

$N_{patient} = 40$



CNN_3: Pred 0

$N_{patient} = 80$

Fig. 7.34 Occlusion sensitivity maps for a single image (top) from three retrained CNNs at small, medium, and large N, for the mediastinal dataset. Red areas indicate a drop in the prediction when that area was occluded (i.e. that area is important for the classification) and blue indicates the inverse. The scale on each occlusion map is different, so scales are shown

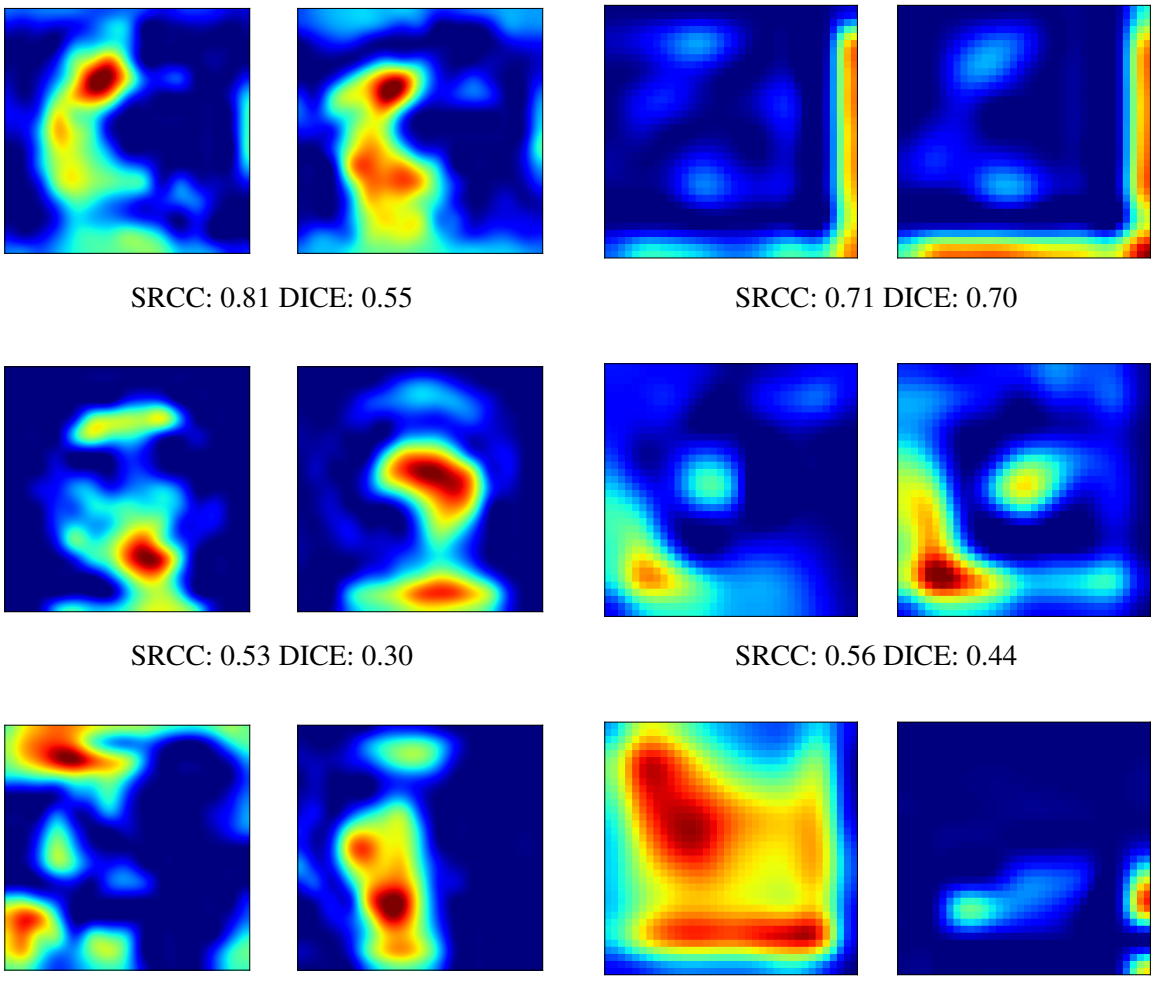


Fig. 7.35 Grad-CAM maps and their corresponding SRCC and DICE scores for three examples for the brain dataset (left) and the LUNA dataset (right). Note the LUNA dataset is 3D so these are single central slices

Table 7.10 Mean DICE and SRCC scores comparing CNNs retrained with small, medium, and large N for the brain and LUNA datasets

Dataset	***N***	**Mean DICE Score**			**Mean SRCC**		
		CNN_1 vs CNN_2	CNN_1 vs CNN_3	CNN_2 vs CNN_3	CNN_1 vs CNN_2	CNN_1 vs CNN_3	CNN_2 vs CNN_3
Brain	30	0.40	0.43	0.30	0.49	0.48	0.35
	90	0.44	0.48	0.47	0.50	0.52	0.48
	190	0.53	0.54	0.48	0.56	0.60	0.52
LUNA	80	0.46	0.23	0.18	0.52	0.20	0.05
	200	0.25	0.24	0.19	0.50	0.35	0.25
	1800	0.33	0.33	0.36	0.46	0.43	0.55

for the same images, and these problems will need to be addressed to allay clinicians' fears. One solution could be to use ensemble learning to get an aggregated prediction, which would be more stable [270] [234]. Another solution could be to measure the uncertainty in the CNN prediction directly. Dropout can be used during testing to give a Bayesian approximation that represents uncertainty [74] [271]. This would reduce the danger of differing CNN outputs by giving an uncertainty in the prediction. In addition to this, more work needs to be done to understand why retrained CNNs are producing such different results. This may involve a deeper investigation of the loss function topology and the differences between local minima in this space.

Although not discussed here there are many other secondary sources of instability in CNN predictions, such as image input noise [229] [230], scanner and protocol differences [206] [272], and even software differences [273]. Although radiomic features themselves are mathematically determined and thus stable, the array of parameters to set (such as normalisation, discretisation, feature specifications) mean results can often be hard to reproduce [27] [28]. Further work could look at quantifying the magnitudes of these sources of instability.

7.12 How Does Segmentation Affect the Performance?

The above models give good accuracies at high N, despite the tumours not being segmented. This is not so surprising for the CNN models, as with their adaptability one would expect them to be able to locate the relevant information within an image automatically. It is common for CNNs to be trained without segmentation. However, it is surprising for the radiomic models. Radiomic features are designed to work on tumour volumes, and it is

very unusual to train a radiomic-based classification model without tumour segmentation [26] [38]. In this section we investigate how the radiomic models are achieving such good performances without segmentation and how different segmentations affect the performance.

7.12.1 Method

Performance on Segmented vs Unsegmented Data Expert delineations were available for the brain dataset, so we first compared the performance of CNN and radiomic models with and without these segmentations. For the segmentation-based CNN models any pixels outside the tumour region were set to zero. The CNNs were then trained exactly as in previous experiments. For the radiomic models the segmentations were used as masks, resulting in ten more shape-based features compared to the non-segmented models. Experiments were repeated and SVMs trained exactly as above, and we again plotted curves showing $N_{patient}$ vs accuracy, using standard error for the error bars.

Is There a Hidden Bias in the Brain Images? We can perhaps understand the radiomic models' performances on the LUNA and mediastinal datasets, as although there are no exact segmentations, they are contained in bounding boxes. However, the performances of the radiomic models on the unsegmented brain dataset are harder to explain. It could be due to: (1) the radiomic features having the ability to extract information from the tumour, despite the surrounding tissue (this is unlikely as the tumour occupies a small region compared the image), (2) there also being clinically important information in the surrounding tissues that the radiomic features are using, or (3) a bias in the images which the radiomic features are using to cheat (some form of the *Clever Hans effect*). The dataset consists of 2D slices that must have been selected from original 3D MRI images. The three tumour types usually occur in different regions of the brain, so we suspected that the selection of the 2D slice could be biasing the models. Rather than categorising an image based on the tumour, the models could be using the position and orientation of the slice. To test this we built an SVM model using only three features: the number of zero-intensity pixels (i.e. the size of the background region in each image), the maximum value, and the orientation of the slice (sagittal, axial, or coronal). We rescaled the features then used them to train a linear SVM with C=1 and $N_{patient} = 190$, performing 30-fold cross-validation as above.

To further test for bias, we recalculated the radiomic features while (1) placing an 80×80 pixel square of intensity zero centred on the tumour mean and (2) thresholding the image by setting pixels less than 0.1 to zero and pixels greater than 0.1 to one. Examples of these images are shown in Figure 7.36. We used these features to train SVMs as previously, again with $N_{patient} = 190$ and 30-fold cross-validation.

(a) Full Image

(b) Image with an 80×80 pixel zero-intensity square covering the tumour

(c) Image with pixels < 0.1 set to zero and >0.1 set to one, meaning only the skull outline is shown

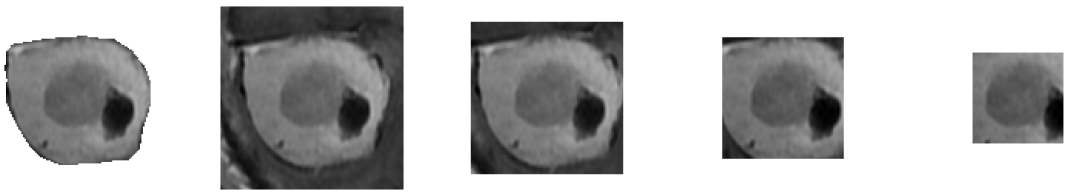
Fig. 7.36 Examples of images used to investigate potential bias in the brain dataset. The tumour outline is shown in red

Is a Precise Segmentation Important? There may be a bias affecting the brain dataset, but the LUNA and mediastinal radiomic models also produce good results without segmentation. This suggests that the features used are robust to changes in the bounding boxes containing the nodes (as these cubes can be thought of as a very rough delineations of the node, rather than precise segmentations). To test this, for both datasets we calculated the top ten radiomic features using cubes slightly smaller and larger than the original cubes (using cubes of side length 8 mm and 16 mm for the LUNA dataset and 32 mm and 64 mm for the mediastinal dataset). We then correlated the features extracted from the different cube sizes.

For the brain dataset we ran a similar experiment, recalculating radiomic features using square ROIs of side length 60 pixels, 80 pixels, 100 pixels, 120 pixels, and 140 pixels centred on the mean of each tumour. We compared models trained on these features to those trained using features from the precise tumour segmentations (see Figure 7.37 for some examples). SVMs were trained as above using $N_{patient} = 190$ and repeated 30 times.

7.12.2 Results

Performance on Segmented vs Unsegmented Data Figure 7.38 compares the learning curves for the brain dataset with and without segmentation. For the CNN models at low N the performances with and without segmentation were very similar. Above $N_{patient} = 90$ the CNNs trained without segmentation generally performed slightly better (at $N_{patient} = 190$ 0.917 ± 0.008 vs 0.894 ± 0.010 ($p = 0.06$)). For the radiomic models we see a similar trend (at $N_{patient} = 190$ 0.885 ± 0.007 vs 0.869 ± 0.007 ($p = 0.06$)). These results show that even



(a) Precise tumour mask (b) 120×120 pixel box (c) 100×100 pixel box (d) 80×80 pixel box (e) 60×60 pixel box

Fig. 7.37 Examples of images used to test the importance of precise segmentation using the brain dataset. (a) is an image using the exact segmentation and (b) to (e) are squares of different sizes centred on the tumour mean

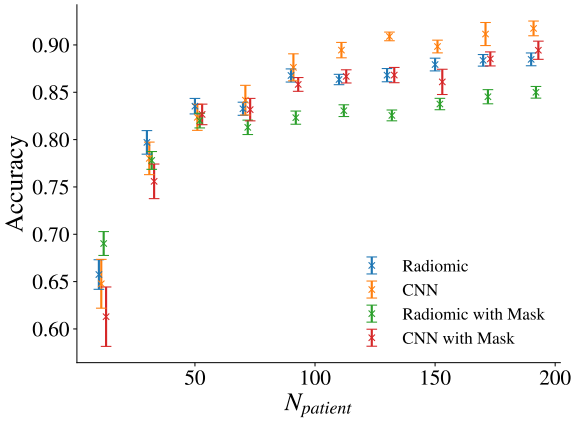


Fig. 7.38 Learning curves for radiomic and deep features with and without segmentation for the brain dataset (points staggered)

at low N the models trained without segmentation performed as well as those trained with segmentation. As mentioned this result is particularly surprising for the radiomic models.

Is There a Hidden Bias in the Brain Images? Using only three features; the number of zero-intensity pixels, the maximum value, and the orientation of the slice, we achieved an accuracy of 0.776 ± 0.006. Using features derived from images with the tumours masked by an 80×80 pixel square we achieved an accuracy of 0.866 ± 0.007. Deriving features from images with only the skull outline we achieved an accuracy of 0.743 ± 0.008. These can be compared to an accuracy of 0.885 ± 0.007 attained when using the full images in 7.5.

These results clearly shows that there is a bias in the images. It is possible to achieve a high accuracy without any information concerning the tumour itself. The three types of brain tumours classified here tend to grow in specific areas of the brain (meningiomas on membranes surrounding the brain and spinal chord, gliomas in the cerebrum or cerebellum,

and pituitary tumours on the pituitary gland [274] [275] [276]), so positional information and information outside the tumour are useful for classification. However, with 2D slices this information is implicitly contained within the slice, which has been selected because it contains the tumour. The position and orientation of the slice can therefore be used as a proxy for tumour type, which would not be possible if using full 3D images. This means that any results derived using the whole images are not valid measures of the performance on this task. Several publications have used this dataset with whole-image inputs into CNNs and made no reference to this bias [239] [240] [241] [121] [277]. This shows the care that needs to be taken when analysing images and using external datasets. Involving a radiologist would no doubt reduce the danger of this happening, demonstrating the importance of working in collaboration with medical experts.

This finding does not invalidate the results in the rest of this chapter. The task is still a classification problem, and the performance when using the segmentation shows that information contained within the tumours can also be used to give a high accuracy. However, we cannot say whether it is this information or the orientational information associated with the slices that the models are using to classify the images. This may to some degree explain the saliency maps in 7.8, which in some cases focused on the tumour and other cases on the area surrounding the skull. The fact that from the maps we could not identify this bias does not speak wonders for their utility.

Is a Precise Segmentation Important? The correlations between radiomic features derived from different bounding cube sizes for the LUNA and mediastinal datasets are shown in Tables 7.11 and 7.12 respectively. They show that most features are fairly robust (r>0.9) to changes in the cube size. Both problems involve classifying nodes (which are hyperdense) generally surrounded by less dense background tissue. We can reason that changing the bounding box in many cases only adds or removes this background tissue so does not affect the key features associated with the nodes.

Despite the evidence of bias associated with tumour location, we can still use different bounding boxes to test whether precise segmentation is important for the brain dataset. The bounding boxes are small so do not include the larger-scale orientational information that could confound the results. The results are shown in Table 7.13. We see that using bounding boxes rather than precise segmentations does not hinder the models' performances, and using larger bounding boxes actually improves the performance compared to using only the segmented region. The average size of the tumours is 74 pixels in diameter, so for the larger bounding boxes considerable surrounding tissue is being used. [235] showed that dilating the segmented regions could improve the performance when using radiomic features on this

Table 7.11 Correlations of the top LUNA radiomic features derived from cubes of side length 12 mm to those derived from cubes of different sizes

Feature Name	Cube Side Length/mm	
	8	16
Skewness	0.91	0.84
Median	0.94	0.91
GLCM Autocorrelation	0.93	0.75
LoG-2mm 10^{th} Percentile	0.81	0.74
LoG-2mm Interquartile Range	0.83	0.59
LoG-1mm Skewness	0.94	0.87
Wavelet-LLL Range	0.88	0.84
90^{th} Percentile	0.94	0.87
Exponential Variance	0.90	0.79
LoG-3mm Median	0.66	0.57

Table 7.12 Correlations of the top mediastinal radiomic features derived from cubes of side length 48 mm to those derived from cubes of different sizes

Feature Name	Cube Side Length/mm	
	32	64
CT LoG-3mm Uniformity	0.91	0.96
CT LoG-1mm Total Energy	0.91	0.94
CT GLCM Informations Measure of Correlation 1	0.90	0.95
CT Wavelet-HLH Entropy	0.99	0.99
CT LoG-3mm Robust Mean Absolute Deviation	0.89	0.94
CT Wavelet-LLL Variance	0.88	0.94
PET LoG-1mm Kurtosis	0.86	0.92
CT Wavelet-HLH Variance	0.98	0.99
PET LoG-1mm Skewness	0.89	0.94
PET LoG-3mm Skewness	0.92	0.95

Table 7.13 Performance using radiomic features derived using different bounding boxes compared to a precise segmentation for the brain dataset

Segmentation Type	**Accuracy**
Tumour Mask	0.869 ± 0.007
120×120 pixel box	0.895 ± 0.004
100×100 pixel box	0.886 ± 0.004
80×80 pixel box	0.896 ± 0.004
60×60 pixel box	0.875 ± 0.005

dataset, indicating that surrounding tissue is useful for classification. These results show that the surrounding tissue is useful even without using a precise segmentation.

No two image classification problems are the same, and therefore the usefulness and necessity of a segmentation will depend on the task at hand. However, given the time and ambiguity associated with segmentation, one should first consider whether it will bring actual significant benefits to clinical decisions using those images. Further work is needed to examine this effect in more detail on different datasets. We would also be interested to see how different segmentations (i.e. different experts or algorithms) affect results in a broader range of tasks.

7.13 Conclusions and Afterthoughts

In this chapter we investigated a wide range of properties of both radiomic and CNN-based models. Using three very different datasets we showed that the performances of both model types increased predictably as we increased the dataset size, following power-law curves. At larger N the radiomic features reached a performance limit, whereas because of their adaptability the performance of deep feature-based models continued to improve. As datasets get bigger than a few hundred patients we therefore expect deep features to outperform radiomic features. Future efforts in radiomics should be dedicated to identifying new features more suited to medical tasks. Combining radiomic and deep features did not improve the performance for any of our datasets, contrary to other studies.

We then attempted to interpret the CNN models using Grad-CAM and occlusion sensitivity maps. While we could gain some insight, the lack of consistency meant it was hard to know when we could trust the interpretations. It was also often unclear how these maps could be linked to concrete biological effects. More work needs to be done to systematically test the numerous interpretation methods to qualify what exactly each can give us and to quantify each method's performance and reliability. We found radiomic-based models to be more

interpretable, especially when built with simple first-order features. Higher-order features were harder to interpret but could partially by explained by correlation with other features. Further radiomics studies should look at better explaining these features or trying to create other features more directly linked to interpretable biological indicators. For both approaches attempts at interpretation are rare in published studies, but interpretation is essential for clinical use. We hope that future work will assign more importance to this critical factor, rather than just judging models by performance.

We also examined whether the more opaque deep features could be recreated using radiomic features. Using these replacements to build new models, we measured how faithfully these models could reproduce the original CNN models' results. We found that when replacing each of the top five deep features by three radiomics features we could reproduce the original models' results with accuracies over 0.75 for all three datasets and with an accuracy of 0.84 for the brain dataset. Although the problems associated with interpreting radiomic features remain, this method presents a new mechanism for understanding the CNN black box. Further work could look to improve these results by exploring different methods of feature reduction and examining why some deep features are more amenable to replacement than others.

Moving onto stability, we found that despite similar overall performances, retrained CNNs predict very different results image-by-image and often focus on entirely different regions of images when retrained. This problem is particularly acute with less data, suggesting that a rethinking of our methods of CNN evaluation is needed. In the future, Bayesian-based uncertainty approximations or ensemble models may be the best ways to quantify or reduce this instability. A comparison to other sources of instability, such as image reconstruction method or scanner differences, could help determine the importance of this effect.

Finally, we evaluated the value of prior segmentation. We showed that for the brain dataset models without prior segmentation performed better, but on further investigation found that there is a bias in these images. For all three datasets we showed that bounding boxes can give good classification performance even using radiomic models, calling into question the necessity of the time-consuming segmentation process. More work needs to be done to study this effect on different datasets.

Chapter 8

Adapting a Model to Data from a Different Scanner

8.1 Problem Background and Previous Work

Scientific publications show the great promise and potential of automated medical image analysis. However, very few published scientific results have been clinically realised. One barrier to wider adoption of automated models is *domain shift*. Academic models are often trained on small, single-scanner datasets, and these do not generalise well to data from different scanners. The quality of the machine, its parameters, and the acquisition protocol, as well as epidemiological variation between patient populations (e.g. in different countries), can all affect imaging data. This has been demonstrated in several MRI and CT studies [206] [278] [279] [280] [281].

Several solutions have been proposed to solve the problem of domain shift. [280] [281] and [279] create more heterogeneous training datasets by combining data from different scanners, resulting in models that generalise better. A disadvantage of this method is that data needs to be shared between centres, which is not always possible because of data privacy issues. Another solution is to use transfer learning, using data from one scanner to fine-tune a model initially trained on data from a different scanner. In this scenario the data does not have to be shared, only the model. This method has been used to adapt models in several studies [282] [283] [284] [285]. Other groups have attempted to harmonise the input data, either by harmonising the image acquisition procedure (e.g. EARL for PET acquisitions [286]) or by using GANs to transform images from one domain to another [287] [288] [131].

Another method, specifically for harmonising radiomic features, is *ComBat*. Originally developed for genomic data, ComBat harmonises features from different domains by shifting

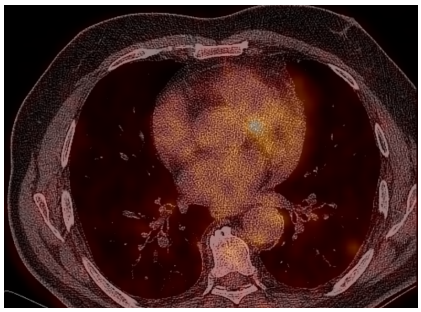
and rescaling their distributions. It has been shown to successfully remove imaging protocol effects while preserving relevant individual information in CT phantom and patient data [289], FDG-PET breast cancer scans [42], and MRI phantom and patient data [290]. ComBat is appealing because it is easy to understand and deploy and does not need a large dataset from the second domain. Images do not have to be shared between centres, although the radiomic features do.

In this chapter we propose a new solution to the problem of deep learning-based domain shift by adapting ComBat for deep features. We compare this method to using no domain adaptation, using a mixture of data from two centres, and using a transfer learning approach. We also investigate how the number of data available for domain adaptation affects the results.

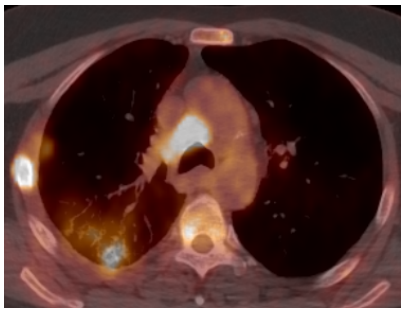
8.2 Dataset

To test these approaches we used the mediastinal dataset from Chapter 5 in addition to a second cohort of 75 patients acquired on a different scanner. These data were acquired on a digital GE Discovery MI PET/CT (DMI) system with a 25 cm FOV. PET scans were performed around 60 minutes after intravenous injection of 2.5 MBq/kg of FDG with 120 second acquisition per bed position. Data were reconstructed using BSREM (Q.Clear) with a penalization β-factor set at 800. FDG-PET voxels were of size 2.8 mm inter-plane and varied from 2.34 mm to 2.73 mm in-plane. CT in-plane resolution varied from 0.8 mm to 1.37 mm and inter-plane resolution was 1.25 mm. These scans, from a newer scanner, had improved contrast (see Figure 8.1 for a comparison). As with the first cohort, the scans were analysed by a physician with 15 years' experience in thoracic imaging and positions of pathological mediastinal nodes were marked. This gave a total of 65 positive nodes.

Our original mediastinal model consisted of two phases; the first phase used a U-Net to generate candidate cubes, and the second phase used a ResNet to classify these cubes. To test inter-scanner effects we only used the second phase of this process (we found that phase one was not strongly affected by inter-scanner effects). For both sets of patients we therefore preprocessed the images as in 5.4 then used the U-Net test 53 setup detailed in 5.10 to generate a set of candidate cubes marked as positive or negative. To measure the performances of the different domain adaptation methods we used this cube-level labelling, rather than worrying about converting to node-level labels. Full details of the dataset are shown in Table 8.1. We used cross-validation to evaluate the different approaches (as the dataset was not large enough to keep an independent test set), randomly splitting the data into training and validation cohorts for each test. For each of these splits we ensured that



(a) Scanner 1



(b) Scanner 2

Fig. 8.1 Comparison of images from (a) scanner 1, Gemini TF (Philips Medical Systems) and (b) scanner 2, Discovery MI (GE Medical Systems). Scanner 2 is newer and the images are much clearer and less blurred. Images captured using *LIFEx* [162]

Table 8.1 Summary of both mediastinal datasets, showing the numbers of patients and positive nodes in each cohort. Also shown are the number of positive and negative cubes in each cohort for one cross-validation (these numbers varied slightly across different cross-validations)

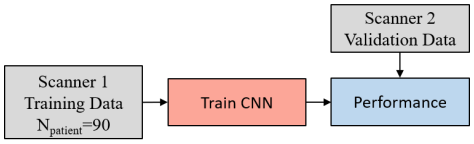
Scanner	**Cohort**	**Patients**	**Positive Nodes**	**Positive Cubes**	**Negative Cubes**
Scanner 1	Training Set 1	45	45	2012	3135
	Training Set 2	45	45	2205	3188
Scanner 2	Training Set	45	45	1927	3927
	Validation Set	30	20	683	2601

we had the same number of patients and positive nodes in each cohort. Even with the same number of patients and positive nodes, the number of cubes in the training and validation cohorts was slightly different for each cross-validation. However, the differences were small and random so averaged out over repeated tests.

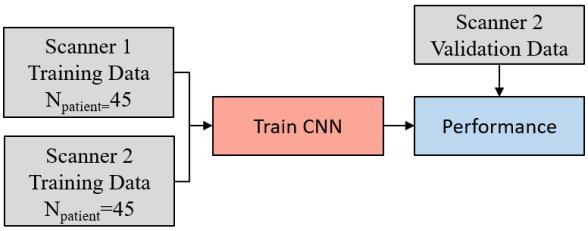
8.3 Method

8.3.1 Comparing Domain Adaptation Methods

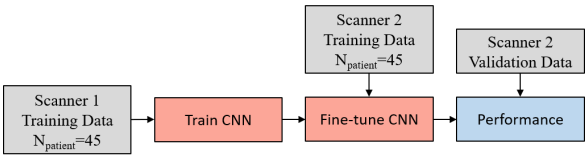
The training and validation cohorts from the two scanners were used to test different domain adaption methods, summarised in Figure 8.2. In each case a 3D ResNet-50 model was trained as in 5.12 and evaluated on the scanner 2 validation set. For all cases we calculated the sensitivity, specificity, and AUC, using five-fold cross-validation to give the mean result and associated standard error. The dataset was not hugely imbalanced, so AUC was a reasonable measure of performance independent of the classification threshold.



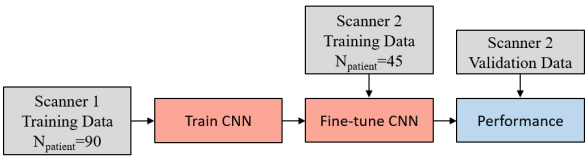
Train on scanner 1. Test on scanner 2



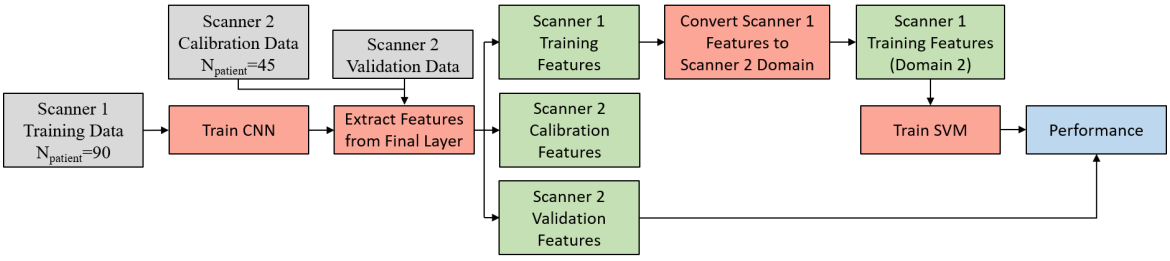
Train on mixture. Test on scanner 2



Train on scanner 1. Fine-tune on scanner 2. Test on scanner 2



Train on scanner 1. Fine-tune on scanner 2. Test on scanner 2 (more data)



Train on scanner 1. Apply ComBat. Test on scanner 2

Fig. 8.2 Different domain adaptation methods tested

Train on scanner 1. Test on scanner 2 The ResNet model was trained using the scanner 1 training set ($N_{patient} = 90$ and $N_{node} = 90$) for 20 epochs with a learning rate of 10^{-5} exactly as in 5.12. This model was then evaluated directly on the scanner 2 validation data.

Train on mixture. Test on scanner 2 The ResNet model was trained on half of the scanner 1 training data ($N_{patient} = 45$ and $N_{node} = 45$) and the scanner 2 training data ($N_{patient} = 45$ and $N_{node} = 45$), for 20 epochs with a learning rate of 10^{-5}. It was then evaluated on the scanner 2 validation data.

Train on scanner 1. Fine-tune on scanner 2. Test on scanner 2 The ResNet model was first trained using half of the scanner 1 training data ($N_{patient} = 45$ and $N_{node} = 45$) for ten epochs then fine-tuned using the scanner 2 training data ($N_{patient} = 45$ and $N_{node} = 45$) for a further ten epochs, with a learning rate of 10^{-5} in each case (other learning rates and numbers of epochs were tested). The model was then evaluated on the scanner 2 validation data.

Train on Cohort 1. Fine-tune on Cohort 2. Test on Cohort 2 (more data) We only used half of the scanner 1 training data in the previous experiment to ensure that the number of data used to train the CNN was consistent with the first two experiments. However, one of the motivations for using transfer learning is to allow more data (from different scanners) to be used during training. We therefore ran an experiment first training the model with all the scanner 1 training data ($N_{patient} = 90$ and $N_{node} = 90$) for 20 epochs then fine-tuning using the scanner 2 training data ($N_{patient} = 45$ and $N_{node} = 45$) for a further 20 epochs, with a learning rate of 10^{-5} in each case (other learning rates and numbers of epochs were tested). The model was evaluated on the scanner 2 validation data.

Train on scanner 1. Apply ComBat. Test on scanner 2 The ResNet model was first trained on the scanner 1 training data ($N_{patient} = 90$ and $N_{node} = 90$). We then extracted features from the final layer of this network, giving a set of 2048 features for the scanner 1 training data, the scanner 2 training data (which we have defined as *scanner 2 calibration data*, as it was not directly used to train the CNN), and the scanner 2 test data. The idea of ComBat is that the scanner 2 features will still contain the information necessary for classification, but their distributions will have been shifted. To align the distributions, the scanner 1 features were shifted to match the scanner 2 features using the transformations

$$\mathbf{x}_1^{(i)*} = (\mathbf{x}_1^{(i)} - \boldsymbol{\mu}_1^0)\frac{\boldsymbol{\sigma}_2^0}{\boldsymbol{\sigma}_1^0} + \boldsymbol{\mu}_2^0 \quad \text{if } y^{(i)} = 0 \tag{8.1}$$

$$\mathbf{x}_1^{(i)*} = (\mathbf{x}_1^{(i)} - \boldsymbol{\mu}_1^1)\frac{\boldsymbol{\sigma}_2^1}{\boldsymbol{\sigma}_1^1} + \boldsymbol{\mu}_2^1 \quad \text{if } y^{(i)} = 1 \tag{8.2}$$

where $\mathbf{x}_1^{(i)}$ are the original scanner 1 features and $\mathbf{x}_1^{(i)*}$ are the scanner 1 features shifted into the scanner 2 domain. $\boldsymbol{\mu}_1^j$ and $\boldsymbol{\sigma}_1^j$ represent the mean and standard deviation of the scanner 1 training features for $y = j$ and $\boldsymbol{\mu}_2^j$ and $\boldsymbol{\sigma}_2^j$ represent the mean and standard deviation of the scanner 2 calibration data.

The transformation was performed separately for the positive and negative cases because in general the number of data in each class are not the same. Performing a single transformation would therefore distort the separations of the classes. This also explains why we cannot transform the scanner 2 features into the scanner 1 domain (which seems the more intuitive solution). We do not know the class of the scanner 2 test data so would not know which transformation to use.

Once transformed, we reduced the number of features using a two-stage feature reduction process. Feature numbers were reduced as in Chapter 7, using a univariate ANOVA F-test to keep the best 140 features then removing any features above a correlation threshold of 0.95 (for two correlated features the mean absolute correlation of each feature was calculated and the feature with the largest mean absolute correlation was removed). Before analysis, features were scaled to have zero mean and unit variance (using the training set to fit the scaling).

To visualise the features before and after ComBat transformation, we plotted histograms of some of the most discriminative features.

8.3.2 Varying the Number of Data Available for Domain Adaptation

To test how the number of data available for domain adaptation affected the effectiveness of the different methods, we repeated the 'Train on scanner 1. Fine-tune on scanner 2. Test on scanner 2 (more data)' and 'Train on scanner 1. Apply ComBat. Test on scanner 2' methods above but with 10, 20, and 30 patients (with 10, 20, and 30 positive nodes) in the scanner 2 training set. As above, these experiments were repeated using five-fold cross-validation. For the transfer learning approach we fine-tuned the model with a learning rate of 10^{-5} and 10 epochs in each case (other learning rates and numbers of epochs were tested). Again, we calculated the sensitivity, specificity, and AUC to give the mean result and associated standard error.

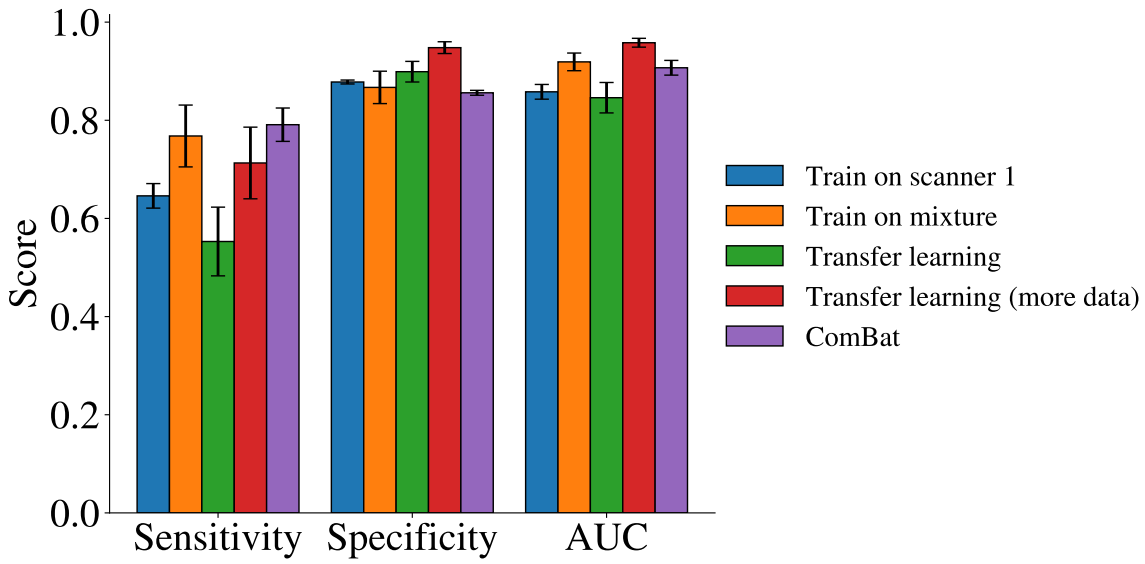


Fig. 8.3 Sensitivity, specificity, and AUC scores for the different domain adaptation methods

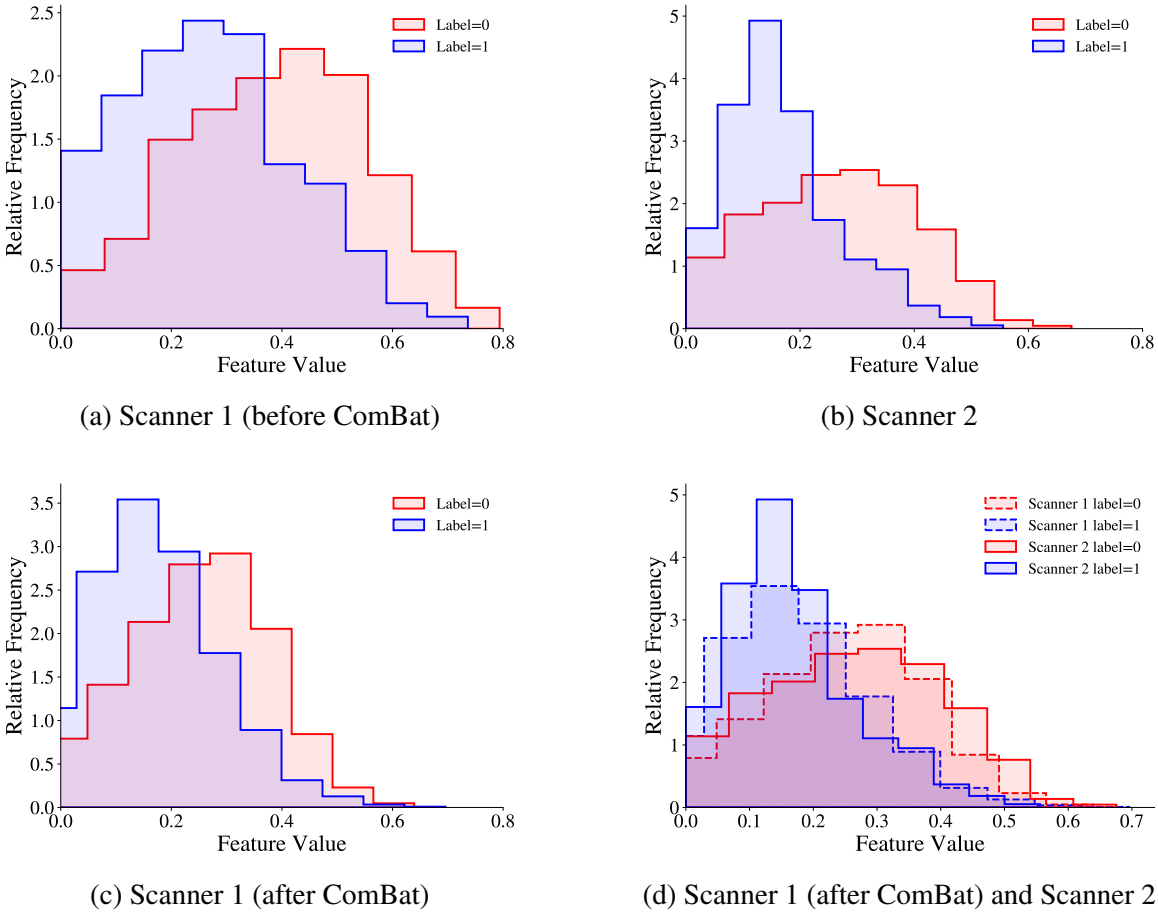
8.4 Results

8.4.1 Comparing Domain Adaptation Methods

The results comparing the different domain adaptation methods are shown in Figure 8.3. Applying the model directly to scanner 2 data resulted in a poor performance, with a sensitivity of 0.65 ± 0.03 and a specificity of 0.878 ± 0.004. Training on a mixed dataset improved the performance significantly, achieving a sensitivity of 0.77 ± 0.06 and a specificity of 0.87 ± 0.03. Using transfer learning with $N_{patient} = 45$ at each stage gave poor results (sensitivity 0.55 ± 0.07 and specificity 0.90 ± 0.02), but using more data in the initial training set improved the performance (sensitivity 0.71 ± 0.07 and specificity 0.95 ± 0.01). Finally, using ComBat to harmonise the features resulted in a sensitivity of 0.79 ± 0.03 and specificity of 0.856 ± 0.005.

Some of these differences in performance can be explained by a shift in the classification threshold, i.e. trading sensitivity for specificity. However, the AUC scores show that the overall performance also improves significantly when training on a mixture of data, transfer learning with more data, and using ComBat, compared to applying the model directly to scanner 2 data.

Figures 8.4 to 8.6 show distributions of individual features before and after ComBat harmonisation. These allow us to see how effective the features are at separating the positive and negative classes. Figure 8.4 shows the effect of domain shift; the feature distributions from scanner 2 are shifted compared to those from scanner 1 (Figure 8.4a vs Figure 8.4b).



(a) Scanner 1 (before ComBat)

(b) Scanner 2

(c) Scanner 1 (after ComBat)

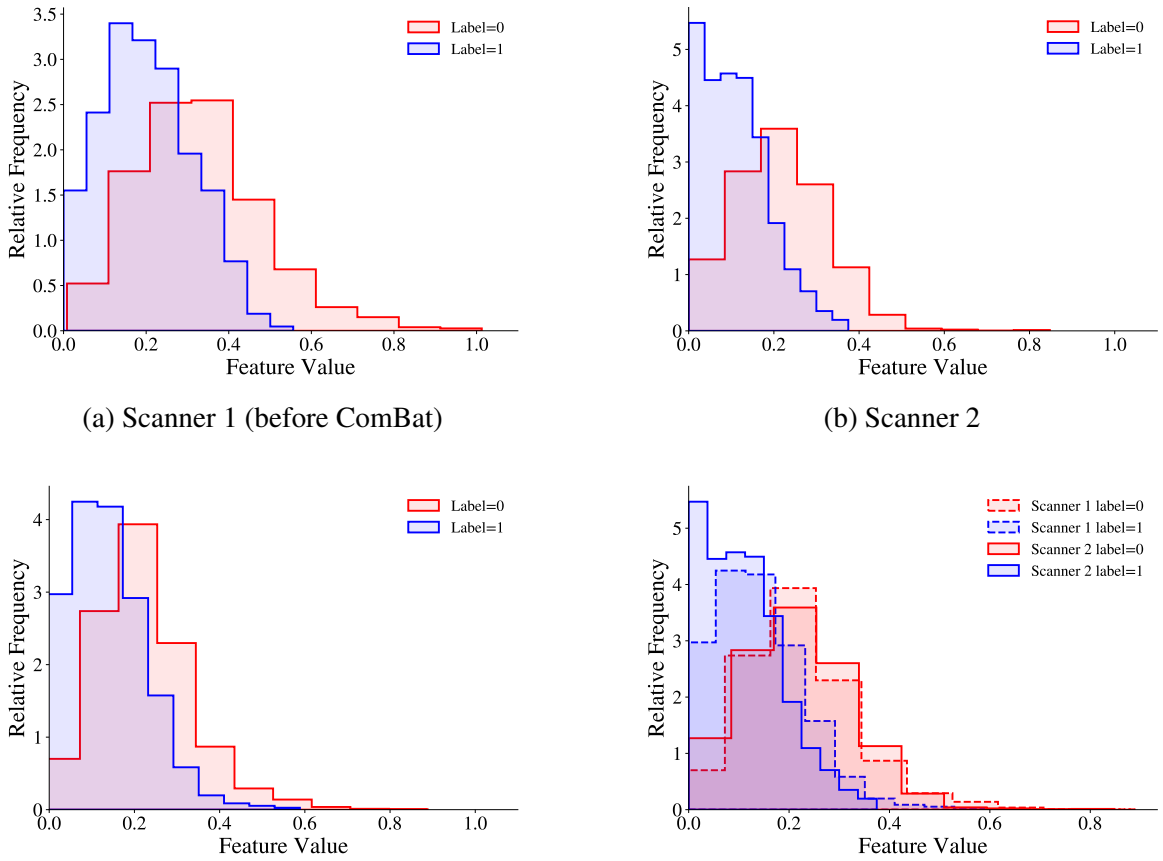
(d) Scanner 1 (after ComBat) and Scanner 2

Fig. 8.4 Distributions showing how one feature separates the positive and negative classes for scanner 1 and scanner 2 data, before and after harmonisation using ComBat

We also see how this could reduce the performance; a feature value of 0.3 on scanner 1 would indicate a positive classification, whereas on scanner 2 it would indicate a negative classification. After harmonisation the feature distributions align (Figure 8.4d overlays the distributions from scanner 2 and scanner 1 after ComBat). Figure 8.5 shows a similar case. Figure 8.6 shows a potential shortcoming of the method. The feature distributions from the first scanner (Figure 8.6a) are more separable than those from the second scanner (Figure 8.6b). This means that when we apply ComBat we lose part of the discriminative power of this feature (Figure 8.6c).

8.4.2 Varying the Number of Data Available for Domain Adaptation

Figure 8.7 illustrates results comparing transfer learning and ComBat methods when using different amounts of scanner 2 training data. We see that when using transfer learning



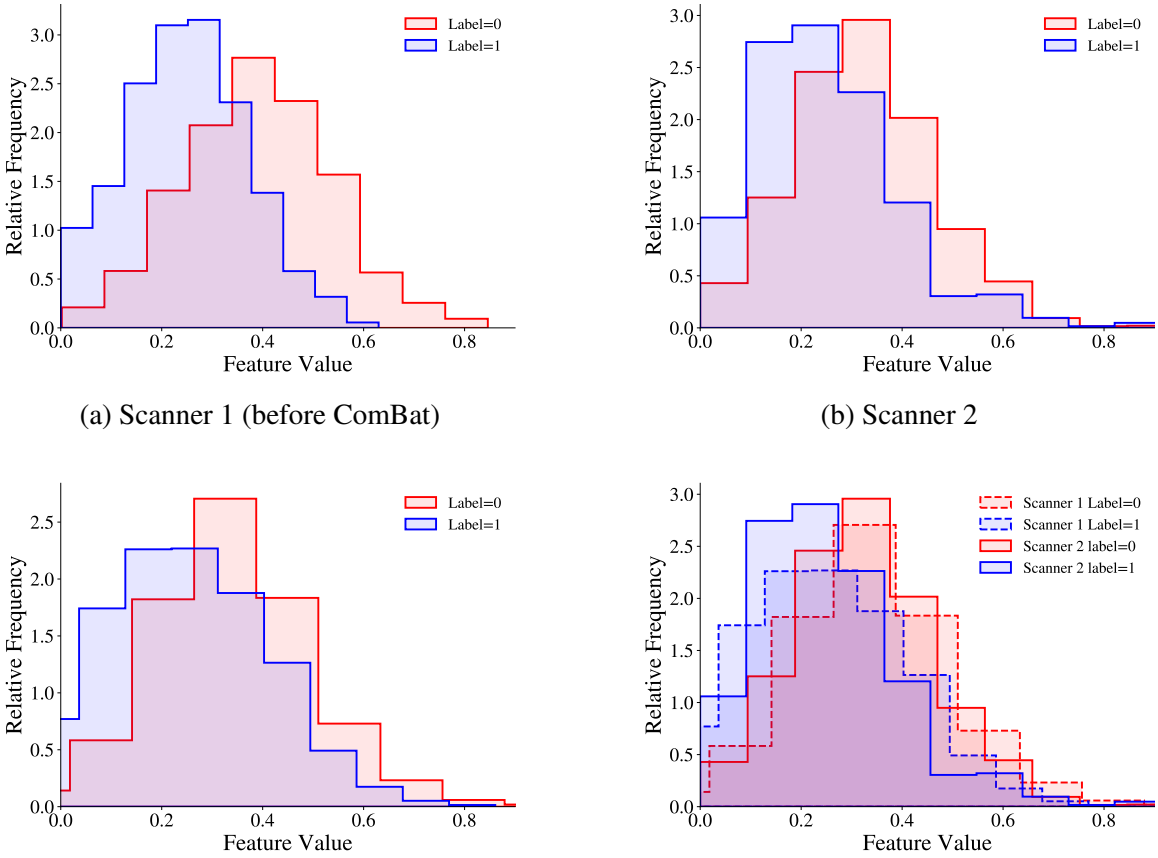
(a) Scanner 1 (before ComBat)

(b) Scanner 2

(c) Scanner 1 (after ComBat)

(d) Scanner 1 (after ComBat) and Scanner 2

Fig. 8.5 Distributions showing how one feature separates the positive and negative classes for scanner 1 and scanner 2 data, before and after harmonisation using ComBat



(a) Scanner 1 (before ComBat)

(b) Scanner 2

(c) Scanner 1 (after ComBat)

(d) Scanner 1 (after ComBat) and Scanner 2

Fig. 8.6 Distributions showing how one feature separates the positive and negative classes for scanner 1 and scanner 2 data, before and after harmonisation using ComBat

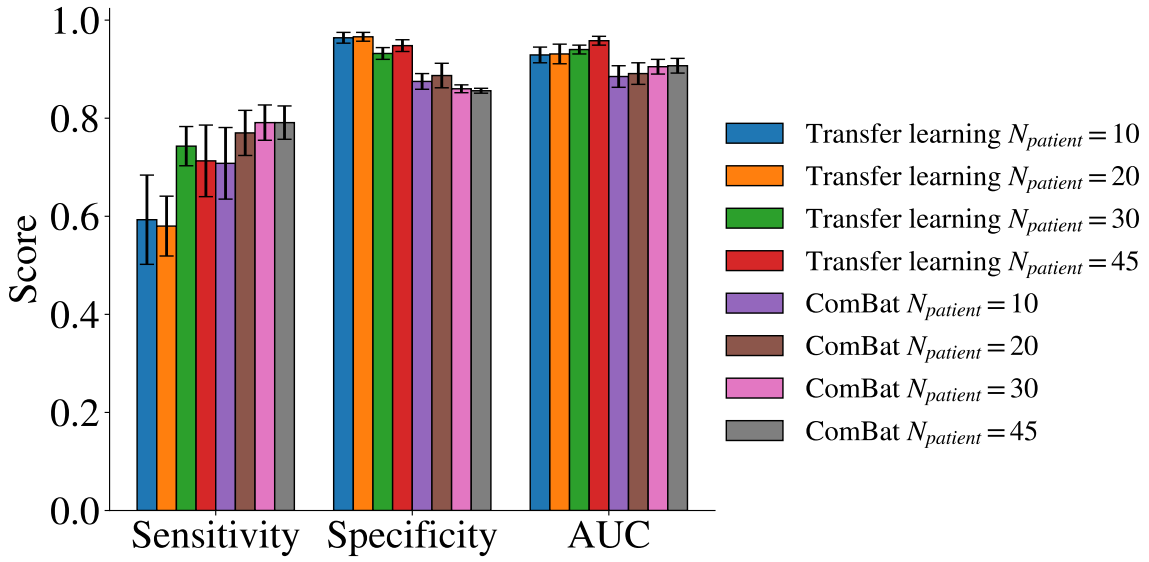


Fig. 8.7 Comparison of the performances of transfer learning and ComBat-based domain adaptation methods with different amounts of training data from scanner 2

we need 30 patients or more in the scanner 2 training cohort to significantly improve the sensitivity. The specificity scores do not change significantly, but looking at the AUC scores we see a gradual improvement as the training cohort increases in size. Using ComBat, even with only 10 patients in the scanner 2 training cohort we significantly improve the sensitivity. Again the specificity does not significantly change, but the AUC shows small improvements as the number of data is increased. Overall, the AUC scores are higher for the transfer learning method compared to the ComBat method. However, the differences are very small compared to the errors and consequently it is hard to draw any strong conclusions. More data is needed to further this study.

8.5 Discussion

Applying our model directly to data from a different scanner gave poor results. Adapting the model by training on a mixture of data, using transfer learning, and using ComBat all significantly improved the performance. Some of this performance difference was due to a shift in the classification cutoff (the tradeoff between sensitivity and specificity). This is still relevant, as we cannot a priori know this shift. The AUC scores also demonstrated significant overall improvement when using these methods, showing that the methods were not simply readjusting the cutoff.

Transfer learning with the full scanner 1 training cohort resulted in the highest AUC, though the differences between this and training on a mixture of data and using ComBat were small. We could not find many other studies quantitatively comparing different domain adaptation methods. [282] compared applying a model directly to data from a different scanner, fine-tuning the model using transfer learning, and training the model on a mixture of data. They found that using transfer learning was most effective, though this was for the task of ultra-low-count amyloid PET/MRI enhancement, quite different to the task investigated here. This is the first study we know of to apply ComBat to deep learning features. The overall AUC was slightly lower than the transfer learning-based approach, but it may still be a viable method. ComBat is easy to implement compared to the laborious task of optimising a transfer learning model, and is conceptually much easier to understand.

We tried to investigate how the number of data available affected the performance of the different methods, but our dataset was too small to draw any firm conclusions. From other studies we know that calibration of the ComBat transformation does not need much data [289] [290]. A comparison of ComBat and transfer learning methods at small dataset sizes would therefore be of interest. We expect that a large amount of data from scanner 2 would favour the transfer learning approach. This is because when using ComBat we only train the CNN on scanner 1 data, whereas when using transfer learning the CNN is trained scanner 1 data then fine-tuned on scanner 2 data. With more data to train on, we expect the CNN to be more discriminative.

We only had data from two scanners, but it would be instructive to investigate how these methods perform on data from more scanners. [280] showed that the performance of a CNN tested on data from a different domain depended on the similarity between different images and imaging protocols. Greater differences between the domains would likely increase the amount of data required when using transfer learning. A large domain difference may also reduce the effectiveness of the ComBat method, as features from the second scanner may not be well separated (as seen in Figure 8.6).

Finally, we did not test all domain adaptation methods. As mentioned in 8.1, some studies have used GANs to transform images from one domain to another. A comparison of this method and the methods discussed here would be of use.

8.6 Conclusions and Afterthoughts

We demonstrated that our model from Chapter 5 could be adapted to data from a second scanner. This is an important proof of the robustness and generalisability of the model. We tested several domain adaptation methods and proposed a new method, ComBat. The

Table 8.2 Advantages and disadvantages of different domain adaptation methods

Method	**Advantages**	**Disadvantages**
Mixed Dataset	• Good Performance • Model robust to data from both scanners and maybe to data from a third scanner	• Need to share data • May need sufficient data from both scanners?
Transfer Learning	• Good Performance • Do not need to share data only the model • Should be more adaptable to data from very different sources (with enough training data)	• Adds more complication to the model • Hard to optimise/fine-tune • Need sufficient data to be effective?
ComBat	• Good Performance • Easy to implement and understand • Does not need many data from the second scanner • Do not need to share data only the features	• Unsure how the performance would suffer with very different images • CNN is trained on fewer data (only from one scanner)

advantages and disadvantages of the methods tested are summarised in Table 8.2. While these results are a promising start, it is clear that much more work needs to be done to explain how domain shift affects different images and how we can mitigate it.

Chapter 9

Conclusions and Perspectives

The work in this manuscript (and the time spent during the thesis itself) can be split into two halves. In the first half, the aim was to use deep learning methods to automate medical imaging problems. In chapter 5 we successfully created a model to automatically detect pathological tumours at an accuracy comparable to that of a physician. With more data and testing, we hope that in the not-too-distant future a model similar to this will be helping detect nodes in real clinical situations. In chapter 6 we built a deep learning model to differentiate PVE from NSI in cardiac PET/CT scans. The method showed potential, and we hope that with more data this can be explored further. These results, and any cursory glance in recent medical journals, make clear the trend towards AI-based solutions. It is clear that automated analysis represents the future of medical imaging.

After working on the problems in chapters 5 and 6, and after reading more and more publications, our interest turned to the deeper methodological aspects of deep learning. Despite the incredible progress and results, very few models have been approved for clinical use (with a few exceptions e.g. [291]). Here we briefly summarise our findings in these areas, and give our views on the future challenges facing the field.

Theoretical Issues with Neural Networks We spent a lot of time in chapters 5 and 6 optimising the hyperparameters in our models with our approach amounting to not much more than educated guesswork. Others have described this optimisation process as alchemy [80]. As someone with a theoretical background this is very frustrating, but this is a problem the whole AI community is wrestling with. As our results show, hyperparameter choices interact in unpredictable ways to strongly impact performance. In addition, many techniques routinely used in deep learning applications, e.g. batch normalisation, dropout, and transfer learning, are poorly understood from a theoretical standpoint [105] [106] [124] [125] [292] [293]. These unknowns lead to the deeper question: How do neural networks actually work?

Despite their enormous capacity to overfit, neural networks tend to generalise well to unseen data. It has been suggested that this is because the loss functions for these models, which are highly non-convex, have many good local minima that give similar performances on the test set [294]. A more complete understanding of the learning process would help to answer some of these theoretical questions.

Interpretability Linking to this lack of theoretical understanding is the lack of interpretability of results. In chapter 7 we showed that common interpretation methods fall far short of giving satisfactory explanations for models' decisions, for both CNN and radiomic models. Not enough effort is made in the literature to interpret or explain (often very complicated) models, and this leads to a lack of trust in the models. Perhaps we need to take a step back from producing ever-more-complicated models, solely focusing on accuracy or AUC, and try to build simpler models that combine good performance with interpretability. More close collaboration and consultation with medical experts could help align the clinical goals with those of the AI engineers. More domain-specific knowledge would also prevent biases in studies, such as found in 7.12.

Lack of Data The most important factor determining the success or failure of a machine learning model is probably the amount of data available to train with. While in other fields networks are trained on millions of images, medical imaging datasets are typically of the order of hundreds of images. This is not because large datasets are not available - hospital computer systems are filled with millions of images - but rather because of the lack of annotations for these images. Any findings recorded are typically stored in a free-text format, from which it is hard to extract structured labels. Converting these records is a field of research in itself (which also often uses deep learning), and we expect more research into leveraging free-text records in the future. Effort has also been made to create standard report templates for future records, which would greatly improve access to useable data [295]. Another solution is to design models that can use the unlabelled data directly (e.g. *self-supervised* or *semi-supervised* models), or even synthesise data using GANs [296]. We would like to see future work rigorously testing the effectiveness of these methods.

As demonstrated in chapter 8, CNN models are sensitive to scanner effects, meaning models trained on single-centre data fail to generalise to data from other centres or scanners. Regulatory and institutional barriers need to be overcome to enable the creation of large multi-centre databases. Some solutions have been proposed, for example *DataSHIELD* and *euroCAT*, and we hope that in the future more data sharing will allow for larger, more heterogeneous datasets [297] [298].

Quality of Scientific Publications One problem endemic to scientific publications in our field is the lack of reproducibility. We wasted many hours trying and failing to reproduce other groups' results, even with access to the code and data. Colleagues have reported the same problem, as have publications [40]. This lack of reproducibility means that it is impossible to compare different methods, and this hinders overall scientific progress. We found this to be the case in chapters 5 and 6, where inconsistent metrics of comparison prevented fair analysis between previous publications. [299] found that many publications in information retrieval research were using weak baselines and that actual performance has not improved over the last decade. [300] reported similar results in neural network pruning, stating that 'this deficiency is substantial enough that it is hard to compare pruning techniques to one another or determine how much progress the field has made over the past three decades'. A personal gripe of mine, papers also too often quote results without error estimates (without which the results are meaningless).

More comprehensive reporting of results and experimental setup would help improve the quality of scientific publications and benefit other researchers. In radiomics the IBSI have proposed common parameter definitions and the RQS has been proposed to assess the quality of radiomic studies [31] [41]. Publication of negative results and more information on different hyperparameters tested during optimsation would give other researchers a broader view of which factors are most important to the problems.

In spite of these problems, we are hugely optimistic about the field of automated image analysis. New research will no doubt continue to push the boundaries of what is possible, adapt and evolve, surprise and amaze. We believe that our three years of hard work and toil have been worthwhile and hope that our contributions will be of use to future researchers. So to the future reader of this work we say: Good luck!

References

[1] C. Liao, J. Chen, J. Liang, J. Yeh, and C. Kao. Meta-analysis study of lymph node staging by 18F-FDG PET/CT scan in non-small cell lung cancer: Comparison of TB and non-TB endemic regions. *Eur J Radiol*, 81:3518–3523, 2012.

[2] A. Wang, E. Athan, P. A. Pappas, et al. Contemporary clinical profile and outcome of prosthetic valve endocarditis. *JAMA*, 297:1354–1361, 2007.

[3] W. S. Halsted. The results of operations for the cure of cancer of the breast performed at the Johns Hopkins hospital from June, 1889 to January, 1894. *Annals of Surgery*, 20:497–555, 1894.

[4] L. Chen, H. Linden, B. Anderson, et al. Trends in 5-year survival rates among breast cancer patients by hormone receptor status and stage. *Breast Cancer Res Treat*, 147:609–616, 2014.

[5] C. Roobottom, G. Mitchell, and G. Morgan-Hughes. Radiation-reduction strategies in cardiac computed tomographic angiography. *Clin Radiol*, 65:859 – 867, 2010.

[6] W. De Vos, J. Casselman, and G. Swennen. Cone-beam computerized tomography (CBCT) imaging of the oral and maxillofacial region: A systematic review of the literature. *Int J Oral Maxillofac Surg*, 38:609 – 625, 2009.

[7] W. R. Webb, W. E. Brant, and N. M. Major. *Fundamentals of Body CT*. Elsevier Health Sciences, 2014.

[8] TEXAS Geosciences. Resolution and size limitations. URL: **https://www.ctlab.geo.utexas.edu/about-ct/resolution-and-size-limitations/**. [Accessed 2020-05-03].

[9] M. Häggström via Wikimedia Commons. CT of a normal brain, axial plane. URL: **https://commons.wikimedia.org/wiki/File:CT_of_a_normal_brain,_axial_1.png**. [Accessed 2021-01-28].

[10] M. Häggström via Wikimedia Commons. CT of a normal abdomen and pelvis, axial plane. URL: **https://commons.wikimedia.org/wiki/File:CT_of_a_normal_abdomen_and_pelvis,_axial_plane_1.png**. [Accessed 2021-01-28].

[11] M. Häggström via Wikimedia Commons. Computed Tomographs of Normal Cervical Vertebrae Axial Plane. URL: **https://commons.wikimedia.org/wiki/File:Computed_tomographs_of_normal_cervical_vertebrae,_axial_plane_1.jpg**. [Accessed 2021-01-28].

[12] F. Bloch. Nuclear induction. *Phys Rev*, 70:460–474, 1946.

[13] P. A. Rinck. *Magnetic Resonance in Medicine, Chapter 13: Contrast Agents*. BoD, 2014.

[14] V. P. Grover, J. M. Tognarelli, M. M. E. Crossey, et al. Magnetic Resonance Imaging: Principles and Techniques: Lessons for Clinicians. *J Clin Exp Hepatol*, 5:246–255, 2015.

[15] E. Van Reeth, I. W. K. Tham, C. H. Tan, and C. L. Poh. Super-resolution in magnetic resonance imaging: A review. *Concepts Magn Reson Part A*, 40A:306–325, 2012.

[16] D. F. van Wijk, A. C. Strang, R. Duivenvoorden, et al. Increasing spatial resolution of 3T MRI scanning improves reproducibility of carotid arterial wall dimension measurements. *MAGMA*, 27:219–226, 2014.

[17] L. G. Nyúl and J. K. Udupa. On standardizing the MR image intensity scale. *Magn Reson Med*, 42:1072–1081, 1999.

[18] A. Carré, G. Klausner, M. Edjlali, et al. Standardization of brain MR images across machines and protocols: bridging the gap for MRI-based radiomics. *Sci Rep*, 10:12340, 2020.

[19] G. D. Nunzio, R. Cataldo, and A. Carlà. Robust Intensity Standardization in Brain Magnetic Resonance Images. *J Digit Imaging*, 28:727–737, 2015.

[20] K. Maher at English Wikibooks via Wikimedia Commons. T1 T2 PD-weighted MRI Images. URL: **https://commons.wikimedia.org/wiki/File:T1t2PD.jpg**. [Accessed 2021-01-28].

[21] J. J. Vaquero and P. Kinahan. Positron Emission Tomography: Current Challenges and Opportunities for Technological Advances in Clinical and Preclinical Imaging Systems. *Annu Rev Biomed Eng*, 17:385–414, 2015.

[22] M. Unterrainer, C. Eze, H. Ilhan, et al. Recent advances of PET imaging in clinical radiation oncology. *Radiat Oncol*, 15:88, 2020.

[23] A. Alavi, S. Hess, T. J. Werner, and P. F. Høilund-Carlsen. An update on the unparalleled impact of FDG-PET imaging on the day-to-day practice of medicine with emphasis on management of infectious/inflammatory disorders. *Eur J Nucl Med Mol Imaging*, 47:18–27, 2019.

[24] Mco44 via Wikimedia Commons. Brain PET-MRI. URL: **https://commons.wikimedia.org/wiki/File:PET-IRM-cabeza-Keosys.JPG**. [Accessed 2021-01-01].

[25] J. E. van Timmeren, D. Cester, S. Tanadini-Lang, H. Alkadhi, and B. Baessler. Radiomics in medical imaging—"how-to" guide and critical reflection. *Insights Imaging*, 11:91, 2020.

[26] G. Langs, S. Röhrich, J. Hofmanninger, F. Prayer, C. H. J. Pan, and H. Prosch. Machine learning: from radiomics to discovery and routine. *Der Radiologe*, 58, 2018.

[27] B. Wichtmann, U. Attenberger, F. M. Harder, et al. Influence of image processing on the robustness of radiomic features derived from magnetic resonance imaging—a phantom study. *ISMRM*, 2018.

[28] B. A. Altazi, G. G. Zhang, D. C. Fernandez, et al. Reproducibility of F18-FDG PET radiomic features for different cervical tumor segmentation methods, gray-level discretization, and reconstruction algorithms. *J Appl Clin Med Phys*, 18:32–48, 2017.

[29] S. S. F. Yip and H. J. W. L. Aerts. Applications and limitations of radiomics. *Phys Med Biol*, 61:R150–R166, 2016.

[30] R. T. Leijenaar, G. Nalbantov, S. Carvalho, et al. The effect of SUV discretization in quantitative FDG-PET Radiomics: the need for standardized methodology in tumor texture analysis. *Sci Rep*, 5:11075, 2015.

[31] A. Zwanenburg, S. Leger, M. Vallières, and S. Löck. Image biomarker standardisation initiative - feature definitions. *arXiv*, 2016.

[32] A. Traverso, M. Kazmierski, M. L. Welch, et al. Sensitivity of radiomic features to inter-observer variability and image pre-processing in Apparent Diffusion Coefficient (ADC) maps of cervix cancer patients. *Radiother Oncol*, 143:88 – 94, 2020.

[33] M. Schwier, J. van Griethuysen, M. G. Vangel, et al. Repeatability of Multiparametric Prostate MRI Radiomics Features. *Sci Rep*, 9:9441, 2019.

[34] Pyradiomics. Pyradiomics: Radiomic Features. **https://pyradiomics.readthedocs.io/en/2.0.1/features.html**, 2020. [Accessed 2020-02-19].

[35] C.-H. Chen, C.-K. Chang, C.-Y. Tu, et al. Radiomic features analysis in computed tomography images of lung nodule classification. *PLOS ONE*, 13:e0192002, 2018.

[36] G. Latif, M. M. Butt, A. H. Khan, O. Butt, and D. N. F. A. Iskandar. Multiclass brain Glioma tumor classification using block-based 3D Wavelet features of MR images. In *2017 4th International Conference on Electrical and Electronic Engineering (ICEEE)*, pages 333–337, 2017.

[37] T. M. Devi, G. Ramani, and S. X. Arockiaraj. MR Brain Tumor Classification and Segmentation Via Wavelets. In *2018 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pages 1–4, 2018.

[38] A. Vial, D. A. Stirling, M. Field, et al. The role of deep learning and radiomic feature extraction in cancer-specific predictive modelling: a review. *Transl Cancer Res*, 7, 2018.

[39] H. J. W. L. Aerts. The Potential of Radiomic-Based Phenotyping in Precision Medicine: A Review. *JAMA Oncol*, 2:1636–1642, 2016.

[40] G. S. Collins, J. B. Reitsma, D. G. Altman, and K. G. M. Moons. Transparent reporting of a multivariable prediction model for individual prognosis or diagnosis (TRIPOD): the TRIPOD statement. *BMJ*, 350:g7594, 2015.

[41] P. Lambin, R. T. Leijenaar, T. M. Deist, et al. Radiomics: the bridge between medical imaging and personalized medicine. *Nat Rev Clin Oncol*, 14:749–762, 2017.

[42] F. Orlhac, S. Boughdad, C. Philippe, et al. A Postreconstruction Harmonization Method for Multicenter Radiomic Studies in PET. *J Nucl Med*, 59:1321–1328, 2018.

[43] A. Hosny, H. J. Aerts, and R. H. Mak. Handcrafted versus deep learning radiomics for prediction of cancer therapy response. *Lancet Digit Health*, 1:e106–e107, 2019.

[44] M. R. Tomaszewski and R. J. Gillies. The Biological Meaning of Radiomic Features. *Radiology*, 298:505–516, 2021.

[45] B. Skourt, A. E. Hassani, and A. Majda. Lung CT Image Segmentation Using Deep Neural Networks. *Procedia Comput Sci*, 127:109–113, 2018.

[46] M. Giger. Machine Learning in Medical Imaging. *J Am Coll Radiol*, 15:512–520, 2018.

[47] A. Esteva, B. Kuprel, R. A. Novoa, et al. Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542:115–118, 2017.

[48] A. Hosny, C. Parmar, J. Quackenbush, L. H. Schwartz, and H. J. W. L. Aerts. Artificial intelligence in radiology. *Nat Rev Cancer*, 18:500–510, 2018.

[49] G. Litjens, T. Kooi, B. E. Bejnordi, et al. A survey on deep learning in medical image analysis. *Med Image Anal*, 42:60–88, 2017.

[50] M. Ghafoorian, N. Karssemeijer, T. Heskes, et al. Location Sensitive Deep Convolutional Neural Networks for Segmentation of White Matter Hyperintensities. *Sci Rep*, 7:5110, 2017.

[51] National Library of Medicine. PubMed Article Search "Machine Learning". URL: **https://pubmed.ncbi.nlm.nih.gov/?term=Machine+Learning**. [Accessed 2020-11-29].

[52] D. Silver, A. Huang, C. Maddison, et al. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[53] J. Konar, P. Khandelwal, and R. Tripathi. Comparison of Various Learning Rate Scheduling Techniques on Convolutional Neural Network. In *2020 IEEE International Students' Conference on Electrical,Electronics and Computer Science (SCEECS)*, pages 1–5, 2020.

[54] R. Rao and G. Fung. On the Dangers of Cross-Validation. An Experimental Evaluation. In *Proceedings of the SIAM International Conference on Data Mining*, pages 588–596, 2008.

[55] D. Kingma and J. Ba. Adam: A Method for Stochastic Optimization. *arXiv*, 2017.

[56] F.-F. Li, R. Krishna, and D. Xu. Cs231n: Convolutional neural networks for visual recognition. *Stanford University*, 2020.

[57] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht. The Marginal Value of Adaptive Gradient Methods in Machine Learning. *arXiv*, 2018.

[58] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. SMOTE: Synthetic Minority Over-sampling Technique. *J Artif Intell Res*, 16, 2002.

[59] M. Fatima and M. Pasha. Survey of Machine Learning Algorithms for Disease Diagnostic. *J Intell Syst Appl*, 9:1–16, 2017.

[60] M. I. Jordan and T. M. Mitchell. Machine learning: Trends, perspectives, and prospects. *Science*, 349:255–260, 2015.

[61] J. Stallkamp, M. Schlipsing, J. Salmen, and C. Igel. Man vs. computer: Benchmarking machine learning algorithms for traffic sign recognition. *Neural Networks*, 32:323 – 332, 2012.

[62] L. Breiman. Random Forests. *Machine Learning*, 45:5–32, 2001.

[63] M. Vallières, E. Kay-Rivest, L. Perrin, et al. Radiomics strategies for risk assessment of tumour failure in head-and-neck cancer. *Sci Rep*, 7:10117, 2017.

[64] J. Ma, Q. Wang, Y. Ren, H. Hu, and J. Zhao. Automatic lung nodule classification with radiomics approach. In *Medical Imaging 2016: PACS and Imaging Informatics: Next Generation and Innovations*, volume 9789, pages 26 – 31, 2016.

[65] Y. Zhang, Y. Zhu, X. Shi, et al. Soft Tissue Sarcomas: Preoperative Predictive Histopathological Grading Based on Radiomics of MRI. *Acad Radiol*, 26:1262 – 1268, 2019.

[66] P. Protopapas, K. Rader, R. Dave, and M. Levine. Lecture 15: Regression Trees and Random Forests. *Harvard CS109A: Introduction to Data Science*.

[67] V. Vapnik and R. Lerner. Pattern recognition using generalized portrait method. *Autom*, 24:774–780, 1963.

[68] H. Shakir, H. Rasheed, and T. Khan. Radiomic feature selection for lung cancer classifiers. *J Intell Fuzzy Syst*, 38:1–9, 2020.

[69] Y. Wu, J.-H. Jiang, L. Chen, et al. Use of radiomic features and support vector machine to distinguish parkinson's disease cases from normal controls. *Ann Transl Med*, 7:773, 2019.

[70] L. Xu, P. Yang, W. Lang, et al. A radiomics approach based on support vector machine using MR images for preoperative lymph node status evaluation in intrahepatic cholangiocarcinoma. *Theranostics*, 9:5374–5385, 2019.

[71] Data Flair. Kernel Functions-Introduction to SVM Kernel and Examples. URL: **https://data-flair.training/blogs/svm-kernel-functions/**, 2020. [Accessed 2020-08-01].

[72] W. S. McCulloch and W. Pitts. A logical calculus of the ideas immanent in nervous activity. *Bull Math Biophys*, 5:115–133, 1943.

[73] I. Goodfellow. *Deep Learning (Adaptive Computation and Machine Learning series)*. The MIT Press; Illustrated edition, 2016.

[74] Y. Gal and Z. Ghahramani. Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning. *arXiv*, 2016.

[75] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger. On Calibration of Modern Neural Networks. *arXiv*, 2017.

[76] S. Maitra, R. K. Ojha, and K. Ghosh. Impact of Convolutional Neural Network Input Parameters on Classification Performance. In *2018 4th International Conference for Convergence in Technology (I2CT)*, pages 1–5, 2018.

[77] I. Kandel and M. Castelli. The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset. *ICT Express*, 6:312 – 315, 2020.

[78] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On Large-Batch Training for Deep Learning: Generalization Gap and Sharp Minima. *arXiv*, 2016.

[79] H. Bertrand. Hyper-parameter optimization in deep learning and transfer learning: applications to medical imaging. *Université Paris-Saclay*, 2019.

[80] T. Yu and H. Zhu. Hyper-Parameter Optimization: A Review of Algorithms and Applications. *arXiv*, 2020.

[81] C. Parmar, P. Grossmann, J. Bussink, P. Lambin, and H. J. W. L. Aerts. Machine Learning methods for Quantitative Radiomic Biomarkers. *Sci Rep*, 5:13087, 2015.

[82] A.-C. Haury, P. Gestraud, and J.-P. Vert. The Influence of Feature Selection Methods on Accuracy, Stability and Interpretability of Molecular Signatures. *PLOS ONE*, 6:1–12, 2011.

[83] J. Cai, J. Luo, S. Wang, and S. Yang. Feature selection in machine learning: A new perspective. *Neurocomputing*, 300:70 – 79, 2018.

[84] L. Toloşi and T. Lengauer. Classification with correlated features: unreliability of feature ranking and solutions. *Bioinformatics*, 27:1986–1994, 2011.

[85] D. P. Chakraborty. A Brief History of Free-Response Receiver Operating Characteristic Paradigm Data Analysis. *Acad Radiol*, 20:915–919, 2013.

[86] A. Oliver. Automatic Mass Segmentation in Mammographic Images. Evaluation Methodology: ROC and FROC Curves. *Universitat de Girona Department of Electronics, Computer Science and Automatic Control*, 2008.

[87] J. Cohen. A Coefficient of Agreement for Nominal Scales. *Educ Psychol Meas*, 20:1, 1960.

[88] J. R. Landis and G. G. Koch. The measurement of observer agreement for categorical data. *Biometrics*, 33:159–174, 1977.

[89] S. Gallagher and A. Russell. Similarity Metric and Non Metric Scores: A Comparison.

[90] J. Bertels, T. Eelbode, M. Berman, et al. Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory and Practice. In *Medical Image Computing and Computer Assisted Intervention: MICCAI 2019*, pages 92–100, 2019.

[91] C. Clopper and E. Pearson. The use of confidence or fiducial limits illustrated in the case of the binomial. *Biometrica*, 26:404–413, 1934.

[92] Kent State University. SPSS Tutorials:Independent Samples t-Test. URL: **https://libguides.library.kent.edu/spss/independentttest**, 2021. [Accessed 2021-02-01].

[93] C. Aschwanden. Not Even Scientists Can Easily Explain P-Values. URL: **https://fivethirtyeight.com/features/not-even-scientists-can-easily-explain-p-values/**, 2015. [Accessed 2021-02-01].

[94] W. W. Piegorsch. Tables of p-values for t- and chi-square reference distributions. *University of South Carolina Statistics Technical Report*, 194, 2002.

[95] B. L. Welch. The generalization of 'Student S' problem when several different population variances are involved. *Biometrika*, 34:28–35, 1947.

[96] L. Sullivan. Hypothesis Testing - Analysis of Variance (ANOVA). URL: **https://sphweb.bumc.bu.edu/otlt/mph-modules/bs/bs704_hypothesistesting-anova/bs704_hypothesistesting-anova_print.html**. [Accessed 2021-02-01].

[97] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. *Proc IEEE*, 86:2278–2324, 1998.

[98] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet Classification with Deep Convolutional Neural Networks. In *Advances in Neural Information Processing Systems*, volume 25, pages 1097–1105, 2012.

[99] S. Raschka. About Feature Scaling and Normalization. URL: **https://sebastianraschka.com/Articles/2014_about_feature_scaling.html**, 2014. [Accessed 2021-01-28].

[100] M. D. Zeiler and R. Fergus. Visualizing and Understanding Convolutional Networks. In *Computer Vision – ECCV 2014*, pages 818–833, 2014.

[101] A. Ruderman, N. C. Rabinowitz, A. S. Morcos, and D. Zoran. Learned Deformation Stability in Convolutional Neural Networks. *arXiv*, 2018.

[102] B. Graham. Fractional Max-Pooling. *arXiv*, 2014.

[103] J. Springenberg, A. Dosovitskiy, T. Brox, and M. Riedmiller. Striving for Simplicity: The All Convolutional Net. In *ICLR (workshop track)*, 2015.

[104] S. Ioffe and C. Szegedy. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *arXiv*, 2015.

[105] S. Santurkar, D. Tsipras, A. Ilyas, and A. Mądry. How Does Batch Normalization Help Optimization? In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 2488–2498, 2018.

[106] G. Yang, J. Pennington, V. Rao, J. Sohl-Dickstein, and S. S. Schoenholz. A Mean Field Theory of Batch Normalization. In *International Conference on Learning Representations*, 2019.

[107] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv*, 2012.

[108] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv*, 2014.

[109] Neurohive. VGG16 – Convolutional Network for Classification and Detection. URL: **https://neurohive.io/en/popular-networks/vgg16/**, 2018. [Accessed 2020-09-21].

[110] A. S. Dar and D. Padha. Medical Image Segmentation A Review of Recent Techniques, Advancements and a Comprehensive Comparison. *J Comput Sci Eng*, 7:114–124, 2019.

[111] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv*, 2015.

[112] L. Liu, J. Cheng, Q. Quan, F.-X. Wu, Y.-P. Wang, and J. Wang. A survey on U-shaped networks in medical image segmentations. *Neurocomputing*, 409:244 – 258, 2020.

[113] C. Szegedy. Going deeper with convolutions. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1–9, 2015.

[114] R. K. Srivastava, K. Greff, and J. Schmidhuber. Training Very Deep Networks. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, page 2377–2385, 2015.

[115] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. *arXiv*, 2015.

[116] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the Inception Architecture for Computer Vision. *arXiv*, 2015.

[117] Google Cloud. Advanced Guide to Inception v3 on Cloud TPU. URL: **https://cloud.google.com/tpu/docs/inception-v3-advanced**, 2019. [Accessed 2019-08-21].

[118] Y. Landau and N. Kiryati. Dataset Growth in Medical Image Analysis Research. *arXiv*, 2019.

[119] A. Hosny, C. Parmar, T. P. Coroller, et al. Deep learning for lung cancer prognostication: A retrospective multi-cohort radiomics study. *PLOS Med*, 15:e1002711, 2018.

[120] C. Lei, B. Hu, D. Wang, S. Zhang, and Z. Chen. A Preliminary Study on Data Augmentation of Deep Learning for Image Classification. In *Internetware 2019: Proceedings of the 11th Asia-Pacific Symposium on Internetware*, pages 1–6, 2019.

[121] S. A. Abdelaziz Ismael, A. Mohammed, and H. Hefny. An enhanced deep learning approach for brain cancer MRI images classification using residual networks. *Artif Intell Med*, 102:101779, 2020.

[122] V. Sandfort, K. Yan, P. J. Pickhardt, and R. M. Summers. Data augmentation using generative adversarial networks (CycleGAN) to improve generalizability in CT segmentation tasks. *Sci Rep*, 9:16844, 2020.

[123] C. Han, K. Murao, S. Satoh, and H. Nakayama. Learning More with Less: GAN-based Medical Image Augmentation. *Med Image Technol*, 37:137–142, 2019.

[124] S. Kornblith, J. Shlens, and Q. V. Le. Do Better ImageNet Models Transfer Better? *arXiv*, 2018.

[125] K. He, R. Girshick, and P. Dollar. Rethinking ImageNet Pre-Training. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4917–4926, 2019.

[126] M. Raghu, C. Zhang, J. M. Kleinberg, and S. Bengio. Transfusion: Understanding Transfer Learning with Applications to Medical Imaging. *arXiv*, 2019.

[127] N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, et al. Convolutional Neural Networks for Medical Image Analysis: Full Training or Fine Tuning? *IEEE Trans Med Imaging*, 35:1299–1312, 2016.

[128] A. Opbroek, M. Ikram, M. Vernooij, and M. de Bruijne. Transfer Learning Improves Supervised Image Segmentation Across Imaging Protocols. *IEEE Trans Med Imaging*, 34:1018–1030, 2015.

[129] A. Menegola, M. Fornaciali, R. Pires, F. V. Bittencourt, S. E. F. de Avila, and E. Valle. Knowledge Transfer for Melanoma Screening with Deep Learning. *arXiv*, 2017.

[130] V. Cheplygina, M. de Bruijne, and J. P. Pluim. Not-so-supervised: A survey of semi-supervised, multi-instance, and transfer learning in medical image analysis. *Med Image Anal*, 54:280 – 296, 2019.

[131] A. Choudhary, L. Tong, Y. Zhu, and M. D. Wang. Advancing Medical Imaging Informatics by Deep Learning-Based Domain Adaptation. *Yearb Med Inform*, 29:129–138, 2020.

[132] A. van Engelen, A. C. van Dijk, M. T. B. Truijman, et al. Multi-Center MRI Carotid Plaque Component Segmentation Using Feature Normalization and Transfer Learning. *IEEE Trans Med Imaging*, 34:1294–1305, 2015.

[133] X. Zhai, A. Oliver, A. Kolesnikov, and L. Beyer. S^4L: Self-Supervised Semi-Supervised Learning. *arXiv*, 2019.

[134] R. Zhang, P. Isola, and A. A. Efros. Colorful Image Colorization. In *Computer Vision – ECCV 2016*, pages 649–666, 2016.

[135] G. Larsson, M. Maire, and G. Shakhnarovich. Learning Representations for Automatic Colorization. *arXiv*, 2016.

[136] D. Pathak, P. Krähenbühl, J. Donahue, T. Darrell, and A. A. Efros. Context Encoders: Feature Learning by Inpainting. *arXiv*, 2016.

[137] C. Doersch, A. Gupta, and A. A. Efros. Unsupervised Visual Representation Learning by Context Prediction. In *2015 IEEE International Conference on Computer Vision (ICCV)*, pages 1422–1430, 2015.

[138] G. Montavon, W. Samek, and K.-R. Müller. Methods for interpreting and understanding deep neural networks. *Digit Signal Process*, 73:1 – 15, 2018.

[139] A. Singh, S. Sengupta, and V. Lakshminarayanan. Explainable Deep Learning Models in Medical Image Analysis. *J Imaging*, 6:52, 2020.

[140] N. Xie, G. Ras, M. van Gerven, and D. Doran. Explainable Deep Learning: A Field Guide for the Uninitiated. *arXiv*, 2020.

[141] K. Simonyan, A. Vedaldi, and A. Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv*, 2014.

[142] D. Smilkov, N. Thorat, B. Kim, F. B. Viégas, and M. Wattenberg. SmoothGrad: removing noise by adding noise. *arXiv*, 2017.

[143] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra. Grad-CAM: Visual Explanations from Deep Networks via Gradient-Based Localization. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 618–626, 2017.

[144] A. Shrikumar, P. Greenside, and A. Kundaje. Learning Important Features Through Propagating Activation Differences. *arXiv*, 2017.

[145] R. Fong and A. Vedaldi. Interpretable Explanations of Black Boxes by Meaningful Perturbation. *arXiv*, 2017.

[146] Q. Zhang, Y. N. Wu, and S.-C. Zhu. Interpretable Convolutional Neural Networks. *arXiv*, 2018.

[147] B. Kim, M. Wattenberg, J. Gilmer, et al. Interpretability Beyond Feature Attribution: Quantitative Testing with Concept Activation Vectors (TCAV). *arXiv*, 2018.

[148] R. Chen, H. Chen, G. Huang, J. Ren, and Q. Zhang. Explaining Neural Networks Semantically and Quantitatively. *arXiv*, 2018.

[149] R. Caruana, Y. Lou, J. Gehrke, P. Koch, M. Sturm, and N. Elhadad. Intelligible Models for HealthCare: Predicting Pneumonia Risk and Hospital 30-Day Readmission. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, page 1721–1730, 2015.

[150] N. Arun, N. Gaw, P. Singh, et al. Assessing the (Un)Trustworthiness of Saliency Maps for Localizing Abnormalities in Medical Imaging. *medRxiv*, 2020.

[151] J. Crosby, S. Chen, F. Li, H. MacMahon, and M. Giger. Network output visualization to uncover limitations of deep learning detection of pneumothorax. In *Medical Imaging 2020: Image Perception, Observer Performance, and Technology Assessment*, volume 11316, pages 125 – 128, 2020.

[152] C. Rudin. Stop explaining black box machine learning models for high stakes decisions and use interpretable models instead. *Nat Mach Intell*, 1:206–215, 2019.

[153] T. Fel and D. Vigouroux. Representativity and Consistency Measures for Deep Neural Network Explanations. *arXiv*, 2020.

[154] J. Adebayo, J. Gilmer, M. Muelly, I. J. Goodfellow, M. Hardt, and B. Kim. Sanity Checks for Saliency Maps. *arXiv*, 2018.

[155] N. Stoddard, J. Heil, and D. Lowery. Anatomy, Thorax, Mediastinum. *StatPearls*, 2020.

[156] M. Häggström via Wikimedia Commons. Mediastinum image. URL: **https://commons.wikimedia.org/wiki/File:Mediastinum.jpg**, 2017. [Accessed 2021-01-03].

[157] V. W. Rusch, H. Asamura, H. Watanabe, et al. The IASLC Lung Cancer Staging Project. A proposal for a new international lymph node map in the forthcoming seventh edition of the TNM classification for lung cancer. *J Thorac Oncol*, 4:568–577, 2009.

[158] F. Bray, J. Ferlay, I. Soerjomataram, R. L. Siegel, L. A. Torre, and A. Jemal. Global cancer statistics 2018: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries. *CA Cancer J Clin*, 68:394–424, 2018.

[159] D. S. Ettinger, G. Bepler, R. Bueno, et al. Non-Small Cell Lung Cancer: Clinical Practice Guidelines in Oncology. *J Natl Compr Canc Netw*, 8:548–582, 2010.

[160] K. Pak, S. Park, G. J. Cheon, et al. Update on nodal staging in non-small cell lung cancer with integrated positron emission tomography/computed tomography: a meta-analysis. *Ann Nucl Med*, 29:409–419, 2015.

[161] M. Hofman, N. C. Smeeton, S. C. Rankin, T. Nuna, and M. J. O'Doherty. Observer variation in FDG PET-CT for staging of non-small-cell lung carcinoma. *Eur J Nucl Med Mol Imaging*, 36:194–199, 2009.

[162] C. Nioche, F. Orlhac, S. Boughdad, et al. LIFEx: a freeware for radiomic feature calculation in multimodality imaging to accelerate advances in the characterization of tumor heterogeneity. *Cancer Res*, 76:4786–4879, 2018.

[163] A. Setio, F. Ciompi, G. Litjens, et al. Pulmonary Nodule Detection in CT Images: False Positive Reduction Using Multi-View Convolutional Networks. *IEEE Trans Med Imaging*, 35:1160–1159, 2016.

[164] Z. Zhang, X. Li, Q. You, and X. Luo. Multicontext 3D residual CNN for false positive reduction of pulmonary nodule detection. *Int J Imaging Syst Technol*, 29:42–49, 2018.

[165] D. Ardila, A. Kiraly, S. Bharadwaj, et al. End-to-end lung cancer screening with three-dimensional deep learning on low-dose chest computed tomography. *Nat Med*, 25:954–961, 2019.

[166] A. J. Weisman, M. W. Kieler, S. B. Perlman, et al. Convolutional Neural Networks for Automated PET/CT Detection of Diseased Lymph Node Burden in Patients with Lymphoma. *Radiology Artif Intell*, 2:e200016, 2020.

[167] M. Feuerstein, D. Deguchi, T. Kitasaki, et al. Automatic mediastinal lymph node detection in chest CT. In *Medical Imaging 2009: Computer-Aided Diagnosis*, volume 7260, pages 266 – 276, 2009.

[168] H. Oda, K. K. Bhatia, M. Oda, et al. Automated mediastinal lymph node detection from CT volumes based on intensity targeted radial structure tensor analysis. *J Med Imaging (Bellingham)*, 4:044502, 2017.

[169] H. R. Roth, L. Lu, A. Seff, et al. A new 2.5D representation for lymph node detection using random sets of deep convolutional neural network observations. *Med Image Comput Comput Assist Interv*, 17:520—527, 2014.

[170] M. Schmidt-Hansen, D. R. Baldwin, E. Hasler, et al. PET-CT for assessing mediastinal lymph node involvement in patients with suspected resectable non-small cell lung cancer. *Chochrane Database Syst Rev*, 2014:CD009519, 2014.

[171] S. A. Smulders, C. M. Gundy, A. van Lingen, E. F. Comans, F. W. J. M. Smeenk, and O. S. Hoekstra. Observer variation of 2-deoxy-2-[F-18]fluoro-D-glucose-positron emission tomography in mediastinal staging of non-small cell lung cancer as a function of experience, and its potential clinical impact. *Mol Imaging Biol*, 9:318–22, 2007.

[172] Y. Wu, P. Li, H. Zhang, et al. Diagnostic value of fluorine 18 fluorodeoxyglucose positron emission tomography/computed tomography for the detection of metastases in non–small-cell lung cancer patients. *Int J Cancer*, 132:E37–E47, 2013.

[173] L. K. Toney and H. J. Vesselle. Neural Networks for Nodal Staging of Non–Small Cell Lung Cancer with FDG PET and CT: Importance of Combining Uptake Values and Sizes of Nodes and Primary Tumor. *Radiology*, 270:91–98, 2014.

[174] P. Goldstraw, J. Crowley, K. Chansky, et al. The IASLC Lung Cancer Staging Project: Proposals for the Revision of the TNM Stage Groupings in the Forthcoming (Seventh) Edition of the TNM Classification of Malignant Tumours. *J Thorac Oncol*, 2:706–14, 2007.

[175] H. Wang, Z. Zhou, Y. Li, et al. Comparison of machine learning methods for classifying mediastinal lymph node metastasis of non-small cell lung cancer from 18F-FDG PET/CT images. *EJNMMI Res*, 7:11, 2017.

[176] E. H. W. Meijering. Spline interpolation in medical imaging: Comparison with other convolution-based approaches. In *2000 10th European Signal Processing Conference*, pages 1–8, 2000.

[177] X. Zhang, Y. Feng, W. Chen, et al. Linear Registration of Brain MRI Using Knowledge-Based Multiple Intermediator Libraries. *Front Neurosci*, 13:909, 2019.

[178] X. Cao, J. Yang, L. Wang, Q. Wang, and D. Shen. Non-rigid Brain MRI Registration Using Two-stage Deep Perceptive Networks. *Proc Int Soc Magn Reson Med Sci Meet Exhib*, 2018:1176, 2018.

[179] Y. Fu, Y. Lei, T. Wang, W. J. Curran, T. Liu, and X. Yang. Deep Learning in Medical Image Registration: A Review. *Phys Med Biol*, 65:20TR01, 2020.

[180] Y. Fu, Y. Lei, T. Wang, et al. LungRegNet: An unsupervised deformable image registration method for 4D-CT lung. *Med Phys*, 47:1763–1774, 2020.

[181] T. Fechter and D. Baltas. One Shot Learning for Deformable Medical Image Registration and Periodic Motion Tracking. *arXiv*, 2020.

[182] T. Lustberg, J. van Soest, M. Gooding, et al. Clinical evaluation of atlas and deep learning based automatic contouring for lung cancer. *Radiother Oncol*, 126:312–317, 2018.

[183] Kaggle Data Science Bowl 2017. Lung Segmentation. URL: **https://www.kaggle.com/zstarosolski/lung-segmentation**. [Accessed 2020-07-03].

[184] N. O. Antropova, H. Abe, and M. L. Giger. Use of clinical MRI maximum intensity projections for improved breast lesion classification with deep convolutional neural networks. *J Med Imaging (Bellingham)*, 5:014503, 2018.

[185] F. Kanavati, S. Islam, E. O. Aboagye, and A. Rockall. Automatic L3 slice detection in 3D CT images using fully-convolutional networks. *arXiv*, 2018.

[186] L. Sibille, R. Seifert, N. Avramovic, et al. 18F-FDG PET/CT Uptake Classification in Lymphoma and Lung Cancer by Using Deep Convolutional Neural Networks. *Radiology*, 294:445–452, 2020.

[187] Northern California PET Imaging Center. Maximum Intensity Projection. URL: **https://norcalscans.org/what-is-pet/our-images/mip**. [Accessed 2021-01-27].

[188] D. Bychkov, N. Linder, R. Turkki, et al. Deep learning based tissue analysis predicts outcome in colorectal cancer. *Sci Rep*, 8:3395, 2018.

[189] O. Iizuka, F. Kanavati, K. Kato, M. Rambeau, K. Arihiro, and M. Tsuneki. Deep Learning Models for Histopathological Classification of Gastric and Colonic Epithelial Tumours. *Sci Rep*, 10:1504, 2020.

[190] D. Nie, J. Lu, H. Zhang, et al. Multi-Channel 3D Deep Feature Learning for Survival Time Prediction of Brain Tumor Patients Using Multi-Modal Neuroimages. *Sci Rep*, 9:1103, 2019.

[191] A. Bizzego, N. Bussola, D. Salvalai, et al. Integrating deep and radiomics features in cancer bioimaging. In *2019 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, pages 1–8, 2019.

[192] H. I. Libshitz and R. J. McKenna Jr. Mediastinal lymph node size in lung cancer. *AJR Am J Roentgenol*, 143:715–8, 1984.

[193] W. Huang and L. Hu. Using a Noisy U-Net for Detecting Lung Nodule Candidates. *IEEE Access*, 7:67905–67915, 2019.

[194] W. Alakwaa, M. Nassef, and A. Badr. Lung Cancer Detection and Classification with 3D Convolutional Neural Network (3D-CNN). *Int J Adv Comput Sci Appl*, 8:8, 2017.

[195] D. Hellwig, T. P. Graeter, D. Ukena, et al. 18F-FDG PET for Mediastinal Staging of Lung Cancer: Which SUV Threshold Makes Sense? *J Nucl Med*, 48:1761–1766, 2007.

[196] S. Kannojia and G. Jaiswal. Effects of Varying Resolution on Performance of CNN based Image Classification An Experimental Study. *J Comput Sci Eng*, 6:451–456, 2018.

[197] Ping An Technology (Shenzhen) Co.,Ltd, China. 3D CNN for False Positive Reduction in Lung Nodule Detection. URL: **https://grand-challenge-public.s3.amazonaws.com/f/challenge/71/f1dddec3-4421-4154-a203-23eb3e894a1c/20171220_083208_PAtech_FPRED.pdf**. [Accessed 2020-07-21].

[198] C. M. Bishop. *Neural Networks for Pattern Recognition*. Oxford University Press, Inc., 1995.

[199] The Data Detective. Towards Data Science: Preprocessing: Differences in Standardization Methods. URL: **https://towardsdatascience.com/preprocessing-differences-in-standardization-methods-de53d2525a87**, 2019. [Accessed 2020-12-19].

[200] Z. Alom, C. Yakopcic, M. Hasan, T. M. Taha, and V. K. Asari. Recurrent residual U-Net for medical image segmentation. *J Medi Imaging (Bellingham)*, 6:1 – 16, 2019.

[201] P. Blanc-Durand, A. Van Der Gucht, N. Schaefer, E. Itti, and J. O. Prior. Automatic lesion detection and segmentation of 18F-FET PET in gliomas: A full 3D U-Net convolutional neural network study. *PLOS ONE*, 13:1–11, 2018.

[202] F. Isensee, P. Kickingereder, W. Wick, et al. Brain Tumor Segmentation and Radiomics Survival Prediction: Contribution to the BRATS 2017 Challenge. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pages 287–297, 2018.

[203] N. Abraham and N. M. Khan. A Novel Focal Tversky Loss Function With Improved Attention U-Net for Lesion Segmentation. In *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, pages 683–687, 2019.

[204] L. Fidon, W. Li, L. C. Garcia-Peraza-Herrera, et al. Generalised Wasserstein Dice Score for Imbalanced Multi-class Segmentation Using Holistic Convolutional Networks. In *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pages 64–76, 2018.

[205] Z. Zhou, L. Chen, D. Sher, et al. Predicting Lymph Node Metastasis in Head and Neck Cancer by Combining Many-objective Radiomics and 3-dimensioal Convolutional Neural Network through Evidential Reasoning. In *2018 40th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, volume 2018, pages 1–4, 2018.

[206] B. Glocker, R. Robinson, D. C. Castro, Q. Dou, and E. Konukoglu. Machine Learning with Multi-Site Imaging Data: An Empirical Study on the Impact of Scanner Effects. *arXiv*, 2019.

[207] G. Habib, P. Lancellotti, M. J. Antunes, et al. 2015 ESC Guidelines for the management of infective endocarditis: The Task Force for the Management of Infective Endocarditis of the European Society of Cardiology (ESC). *Eur Heart J*, 36:3075–3128, 2015.

[208] G. Habib. Management of infective endocarditis. *Heart*, 92:124–130, 2006.

[209] W. G. Daniel, A. Mugge, J. Grote, et al. Comparison of transthoracic and transesophageal echocardiography for detection of abnormalities of prosthetic and bioprosthetic valves in the mitral and aortic positions. *Am J Cardiol*, 71:210 – 215, 1993.

[210] S. Nuvoli, V. Fiore, S. Babudieri, et al. The additional role of 18F-FDG PET/CT in prosthetic valve endocarditis. *Eur Rev Med Pharmacol Sci*, 22:1744–1751, 2018.

[211] A. M. Scholtens, R. P. J. Budde, M. G. E. H. Lam, and H. J. Verberne. FDG PET/CT in prosthetic heart valve endocarditis: There is no need to wait. *J Nucl Cardiol*, 24:1540–1541, 2017.

[212] A. Jiménez-Ballvé, M. J. Perez-Castejon, R. C. Delgado-Bolton, et al. Assessment of the diagnostic accuracy of 18 F-FDG PET/CT in prosthetic infective endocarditis and cardiac implantable electronic device infection: comparison of different interpretation criteria. *Eur J Nucl Med Mol Imaging*, 43:2401–2412, 2016.

[213] M. N. Pizzi, A. Roque, N. Fernandez-Hidalgo, et al. Improving the Diagnosis of Infective Endocarditis in Prosthetic Valves and Intracardiac Devices With 18F-Fluordeoxyglucose Positron Emission Tomography/Computed Tomography Angiography. *Circulation*, 132:1113–1126, 2015.

[214] R. San, P. Lim, and E. Itti. How to differentiate infective from physiologic 18F-Fluorodeoxyglucose positron emission tomography uptake pattern in left prosthetic heart valve? *Arch Cardiovasc Dis*, 12:53, 2020.

[215] D. Shanmugam, D. Blalock, G. Balakrishnan, and J. Guttag. When and Why Test-Time Augmentation Works. *arXiv*, 2020.

[216] G. Wang, W. Li, M. Aertsen, J. Deprest, S. Ourselin, and T. Vercauteren. Test-time augmentation with uncertainty estimation for deep learning-based medical image segmentation. In *MIDL 2018 Conference Submission*, 2018.

[217] N. Moshkov, B. Mathe, A. Kertesz-Farkas, R. Hollandi, and P. Horvath. Test-time augmentation for deep learning-based cell segmentation on microscopy images. *Sci Rep*, 10:5068, 2020.

[218] A.-S. Dirand. Développements méthodologiques pour l'utilisation de caractéristiques radiomiques. *Université Paris-Saclay*, 2020.

[219] O. Russakovsky, J. Deng, H. Su, et al. ImageNet Large Scale Visual Recognition Challenge. *arXiv*, 2015.

[220] Q. Sun, X. Lin, Y. Zhao, et al. Deep Learning vs. Radiomics for Predicting Axillary Lymph Node Metastasis of Breast Cancer Using Ultrasound Images: Don't Forget the Peritumoral Region. *Front Oncol*, 10:53, 2020.

[221] M. Dinesh Kumar, M. Babaie, S. Zhu, S. Kalra, and H. R. Tizhoosh. A comparative study of CNN, BoVW and LBP for classification of histopathological images. In *2017 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1–7, 2017.

[222] T. Xiao, W. Hua, C. Li, and S. Wang. Glioma Grading Prediction by Exploring Radiomics and Deep Learning Features. In *Proceedings of the Third International Symposium on Image Computing and Digital Medicine*, page 208–213, 2019.

[223] J. Hestness, S. Narang, N. Ardalani, et al. Deep Learning Scaling is Predictable, Empirically. *arXiv*, 2017.

[224] D. Amodei, R. Anubhai, E. Battenberg, et al. Deep Speech 2: End-to-End Speech Recognition in English and Mandarin. *arXiv*, 2015.

[225] J. Cho, K. Lee, E. Shin, G. Choy, and S. Do. How much data is needed to train a medical image deep learning system to achieve necessary high accuracy? *arXiv*, 2015.

[226] A.-S. Dirand, F. Frouin, and I. Buvat. A downsampling strategy to assess the predictive value of radiomic features. *Sci Rep*, 9:17869, 2019.

[227] B. Goodman and S. Flaxman. European Union Regulations on Algorithmic Decision-Making and a "Right to Explanation". *AI Mag*, 38:50–57, 2017.

[228] C. Szegedy, W. Zaremba, I. Sutskever, et al. Intriguing properties of neural networks. In *International Conference on Learning Representations*, 2014.

[229] P. Roy, S. Ghosh, S. Bhattacharya, and U. Pal. Effects of Degradations on Deep Neural Network Architectures. *ArXiv*, 2018.

[230] H. H. Aghdam., E. J. Heravi., and D. Puig. Analyzing the Stability of Convolutional Neural Networks against Image Degradation. In *Proceedings of the 11th Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications - Volume 4: VISAPP, (VISIGRAPP 2016)*, pages 370–382, 2016.

[231] J. Z. Forde and M. Paganini. The Scientific Method in the Science of Machine Learning. *arXiv*, 2019.

[232] M. Lucic, K. Kurach, M. Michalski, O. Bousquet, and S. Gelly. Are GANs Created Equal? A Large-Scale Study. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 698–707, 2018.

[233] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger. Deep Reinforcement Learning that Matters. *arXiv*, 2017.

[234] J. R. Zech, J. Z. Forde, and M. L. Littman. Individual predictions matter: Assessing the effect of data ordering in training fine-tuned CNNs for medical imaging. *arXiv*, 2019.

[235] J. Cheng, W. Huang, S. Cao, et al. Enhanced Performance of Brain Tumor Classification via Tumor Region Augmentation and Partition. *PLOS ONE*, 10:e0140381, 2015.

[236] J. Cheng. Brain Tumour Dataset. URL: **https://figshare.com/articles/dataset/brain_tumor_dataset/1512427**, 2017. [Accessed 2021-01-17].

[237] A. Islam, M. F. Hossain, and C. Saha. A new hybrid approach for brain tumor classification using BWT-KSVM. In *2017 4th International Conference on Advances in Electrical Engineering (ICAEE)*, pages 241–246, 2017.

[238] F. P. Polly, S. K. Shil, M. A. Hossain, A. Ayman, and Y. M. Jang. Detection and classification of HGG and LGG brain tumor using machine learning. In *2018 International Conference on Information Networking (ICOIN)*, pages 813–817, 2018.

[239] A. Pashaei, H. Sajedi, and N. Jazayeri. Brain Tumor Classification via Convolutional Neural Network and Extreme Learning Machines. In *2018 8th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 314–319, 2018.

[240] P. Afshar, K. N. Plataniotis, and A. Mohammadi. Capsule Networks for Brain Tumor Classification Based on MRI Images and Coarse Tumor Boundaries. In *ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 1368–1372, 2019.

[241] J. S. Paul, A. J. Plassard, B. A. Landman, and D. Fabbri. Deep learning for brain tumor classification. In *Medical Imaging 2017: Biomedical Applications in Molecular, Structural, and Functional Imaging*, volume 10137, pages 253 – 268, 2017.

[242] C. Jacobs and B. van Ginneken. LUng Nodule Analysis 2016. URL: **https://luna16.grand-challenge.org/Home/**. [Accessed 2020-03-21].

[243] S. G. Armato 3rd, G. McLennan, L. Bidaut, et al. The Lung Image Database Consortium (LIDC) and Image Database Resource Initiative (IDRI): a completed reference database of lung nodules on CT scans. *Med Phys*, 38:915–931, 2011.

[244] Fonova. 3D Deep Convolution Neural Network Application in Lung Nodule Detection on CT Images. URL: **https://grand-challenge-public.s3.amazonaws.com/f/challenge/71/fda79ef2-f2a9-4a04-b90e-efebf928edb6/20170915_095225_LUNA16FONOVACAD_FPRED.pdf**. [Accessed 2020-07-21].

[245] R. N. D'souza, P.-Y. Huang, and F.-C. Yeh. Structural Analysis and Optimization of Convolutional Neural Networks with a Small Sample Size. *Sci Rep*, 10:834, 2020.

[246] J. J. M. van Griethuysen, A. Fedorov, C. Parmar, et al. Computational Radiomics System to Decode the Radiographic Phenotype. *Cancer Res*, 77:e104–e107, 2017.

[247] Keras. Keras Applications. URL: **https://keras.io/api/applications/**. [Accessed 2019-12-03].

[248] J. Cervantes, F. Garcia-Lamont, L. Rodríguez-Mazahua, and A. Lopez. A comprehensive survey on support vector machine classification: Applications, challenges and trends. *Neurocomputing*, 408:189 – 215, 2020.

[249] R. A. Collazo, L. A. M. Pessôa, L. Bahiense, et al. A comparative study between artificial neural network and support vector machine for acute coronary syndrome prognosis. *Pesquisa Operacional*, 36:321 – 343, 2016.

[250] Y. Chang, K. Lafata, W. Sun, et al. An investigation of machine learning methods in delta-radiomics feature analysis. *PLOS ONE*, 14:1–14, 2019.

[251] H. Lin, C. Huang, W. Wang, J. Luo, X. Yang, and Y. Liu. Measuring Interobserver Disagreement in Rating Diagnostic Characteristics of Pulmonary Nodule Using the Lung Imaging Database Consortium and Image Database Resource Initiative. *Acad Radiol*, 24:401 – 410, 2017.

[252] T. K. Ho, M. Basu, and M. H. C. Law. *Data Complexity in Pattern Recognition*. Springer Science and Business Media, 2006.

[253] A. C. Lorena, L. P. F. Garcia, J. Lehmann, M. C. P. Souto, and T. K. Ho. How Complex Is Your Classification Problem? A Survey on Measuring Classification Complexity. *ACM Comput Surv*, 52:5, 2019.

[254] E. Ataer-Cansizoglu, V. Bolon-Canedo, J. P. Cambell, et al. Computer-Based Image Analysis for Plus Disease Diagnosis in Retinopathy of Prematurity: Performance of the "i-ROP" System and Image Features Associated With Expert Diagnosis. *Tansl Vis Sci Technol*, 30:5, 2015.

[255] W. Christens-Barry and A. Partin. Quantitative grading of tissue and nuclei in prostate cancer for prognosis prediction. *Johns Hopkins APL Tech Dig*, 18:226–232, 1997.

[256] Kotikalapudi, Raghavendra and contributors. keras-vis. URL: **https://github.com/raghakot/keras-vis**, 2017. [Accessed 2021-02-01].

[257] S. Lundberg and S. Lee. A unified approach to interpreting model predictions. *arXiv*, 2017.

[258] Z. Bylinskii, T. Judd, A. Oliva, A. Torralba, and F. Durand. What Do Different Evaluation Metrics Tell Us About Saliency Models? *IEEE Trans Pattern Anal Mach Intell*, 41:740–757, 2016.

[259] M. Kümmerer, T. S. A. Wallis, and M. Bethge. Saliency Benchmarking Made Easy: Separating Models, Maps and Metrics. *arXiv*, 2018.

[260] R. M. Haralick, K. Shanmugam, and I. Dinstein. Textural features for image classification. *IEEE Trans Syst Man Cybern B Cybern*, SMC-3:610–621, 1973.

[261] I. Tunali, O. Stringfield, A. Guvenis, et al. Radial gradient and radial deviation radiomic features from pre-surgical CT scans are associated with survival among lung adenocarcinoma patients. *Oncotarget*, 8:96013–96026, 2017.

[262] G. Bhatnagar, J. Makanyanga, B. Ganeshan, et al. MRI texture analysis parameters of contrast-enhanced T1-weighted images of Crohn's disease differ according to the presence or absence of histological markers of hypoxia and angiogenesis. *Abdom Radiol (NY)*, 41:1261–1269, 2016.

[263] G. Penzias, A. Singanamalli, R. Elliott, et al. Identifying the morphologic basis for radiomic features in distinguishing different Gleason grades of prostate cancer on MRI: Preliminary findings. *PLoS One*, 13:e0200730, 2018.

[264] I. S. Klyuzhin, Y. Xu, A. Ortiz, J. M. Lavista Ferres, G. Hamarneh, and A. Rahmim. Testing the Ability of Convolutional Neural Networks to Learn Radiomic Features. *medRxiv*, 2020.

[265] M. W. Dusenberry, D. Tran, E. Choi, et al. Analyzing the Role of Model Uncertainty for Electronic Health Records. In *Proceedings of the ACM Conference on Health, Inference, and Learning*, page 204–213, 2020.

[266] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics*, volume 9 of *Proceedings of Machine Learning Research*, pages 249–256, 2010.

[267] C. Spearman. The proof and measurement of association between two things. *Am J Psychol*, 100:441—471, 1987.

[268] A. Das, H. Agrawal, L. Zitnick, D. Parikh, and D. Batra. Human Attention in Visual Question Answering: Do Humans and Deep Networks Look at the Same Regions? *Comput Vis Image Underst*, 163:90 – 100, 2017.

[269] N. Singh, K. Lee, D. Coz, et al. Agreement Between Saliency Maps and Human-Labeled Regions of Interest: Applications to Skin Disease Classification. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3172–3181, 2020.

[270] L. Chen. Learning Ensembles of Convolutional Neural Networks. *The University of Chicago: Toyota Technological Institute of Chicago*. [Accessed 2021-01-28].

[271] C. Leibig, V. Allken, M. S. Ayhan, P. Berens, and S. Wahl. Leveraging uncertainty information from deep neural networks for disease detection. *Sci Rep*, 7:17816, 2017.

[272] Q. Liu, Q. Dou, L. Yu, and P. A. Heng. MS-Net: Multi-Site Network for Improving Prostate Segmentation With Heterogeneous MRI Data. *IEEE Trans Med Imaging*, 39:2713–2724, 2020.

[273] M. Crane. Questionable Answers in Question Answering Research: Reproducibility and Variability of Published Results. *Trans Assoc Comput Linguist*, 6:241–252, 2018.

[274] Johns Hopkins Medicine. Conditions and diseases: Meningioma. URL: **https://www.hopkinsmedicine.org/health/conditions-and-diseases/meningioma**. [Accessed 2021-01-28].

[275] Johns Hopkins Medicine. Conditions and diseases: Gliomas. URL: **https://www.hopkinsmedicine.org/health/conditions-and-diseases/gliomas**. [Accessed 2021-01-28].

[276] Johns Hopkins Medicine. Conditions and Diseases: Pituitary Tumours. URL: **https://www.hopkinsmedicine.org/health/conditions-and-diseases/pituitary-tumors**. [Accessed 2021-01-28].

[277] Z. N. K. Swati, Q. Zhao, M. Kabir, et al. Brain tumor classification for MR images using transfer learning and fine-tuning. *Comput Med Imaging Graph*, 75:34–46, 2019.

[278] E. H. P. Pooch, P. L. Ballester, and R. C. Barros. Can we trust deep learning models diagnosis? The impact of domain shift in chest radiograph classification. *arXiv*, 2020.

[279] L. Yao, J. Prosky, B. Covington, and K. Lyman. A Strong Baseline for Domain Adaptation and Generalization in Medical Imaging. *arXiv*, 2019.

[280] G. Mårtensson, D. Ferreira, T. Granberg, et al. The reliability of a deep learning model in clinical out-of-distribution MRI data: A multicohort study. *Med Image Anal*, 66:101714, 2020.

[281] E. A. AlBadawy, A. Saha, and M. A. Mazurowski. Deep learning for segmentation of brain tumors: Impact of cross-institutional training and testing. *Med Phys*, 45:1150–1158, 2018.

[282] K. T. Chen, M. Schurer, J. Ouyang, et al. Generalization of deep learning models for ultra-low-count amyloid PET/MRI using transfer learning. *Eur J Nucl Med Mol Imaging*, 47:2998–3007, 2020.

[283] M. Ghafoorian, A. Mehrtash, T. Kapur, et al. Transfer Learning for Domain Adaptation in MRI: Application in Brain Lesion Segmentation. In *Medical Image Computing and Computer Assisted Intervention - MICCAI 2017*, volume 10435, pages 516–524, 2017.

[284] K. Chang, N. Balachandar, C. Lam, et al. Distributed deep learning networks among institutions for medical imaging. *J Am Med Inform Assoc*, 25:945–954, 2018.

[285] M. J. Sheller, G. A. Reina, B. Edwards, J. Martin, and S. Bakas. Multi-Institutional Deep Learning Modeling Without Sharing Patient Data: A Feasibility Study on Brain Tumor Segmentation. *arXiv*, 2018.

[286] N. Aide, C. Lasnon, P. Veit-Haibach, et al. Eanm/earl harmonization strategies in pet quantification: from daily practice to multicentre oncological studies. *Eur J Nucl Med Mol Imaging*, 44:17–31, 2017.

[287] C. Chen, Q. Dou, H. Chen, and P. Heng. Semantic-Aware Generative Adversarial Nets for Unsupervised Domain Adaptation in Chest X-ray Segmentation. *arXiv*, 2018.

[288] F. Mahmood, R. J. Chen, and N. J. Durr. Unsupervised Reverse Domain Adaptation for Synthetic Medical Images via Adversarial Training. *arXiv*, 2017.

[289] F. Orlhac, F. Frouin, C. Nioche, N. Ayache, and I. Buvat. Validation of A Method to Compensate Multicenter Effects Affecting CT Radiomics. *Radiology*, 291:53–59, 2019.

[290] F. Orlhac, A. Lecler, J. Savatovski, et al. How can we combat multicenter variability in MR radiomics? Validation of a correction procedure. *Eur Radiol*, 31:2272–2280, 2020.

[291] M. D. Abràmoff, P. T. Lavin, M. Birch, N. Shah, and J. C. Folk. Pivotal trial of an autonomous AI-based diagnostic system for detection of diabetic retinopathy in primary care offices. *NPJ Digit Med*, 1:39, 2018.

[292] P. Mianjy, R. Arora, and R. Vidal. On the Implicit Bias of Dropout. *arXiv*, 2018.

[293] H. Wu and X. Gu. Towards dropout training for convolutional neural networks. *Neural Networks*, 71:1–10, 2015.

[294] A. Choromanska, M. Henaff, M. Mathieu, G. B. Arous, and Y. LeCun. The Loss Surfaces of Multilayer Networks. *arXiv*, 2015.

[295] D. P. dos Santos and B. Baeßler. Big data, artificial intelligence, and structured reporting. *Eur Radiol Exp*, 2:42, 2018.

[296] X. Yi, E. Walia, and P. Babyn. Generative adversarial network in medical imaging: A review. *Med Image Anal*, 58:101552, 2019.

[297] I. Budin-Ljøsne, P. Burton, J. Isaeva, et al. DataSHIELD: An Ethically Robust Solution to Multiple-Site Individual-Level Data Analysis. *Public Health Genomics*, 18:87–96, 2015.

[298] T. Deist, A. Jochems, J. van Soest, et al. Infrastructure and distributed learning methodology for privacy-preserving multi-centric rapid learning health care: euroCAT. *Clin Transl Radiat Oncol*, 4:24–31, 2017.

[299] W. Yang, K. Lu, P. Yang, and J. Lin. Critically Examining the "Neural Hype". In *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2019.

[300] D. Blalock, J. J. G. Ortiz, J. Frankle, and J. Guttag. What is the State of Neural Network Pruning? *arXiv*, 2020.

[301] NVIDIA. Cuda, release: 10.1.243. URL: **https://developer.nvidia.com/cuda-toolkit**, 2019. [Accessed 2021-01-14].

[302] Python Software Foundation. Python language reference, version 3.8.3. URL: **http://www.python.org**. [Accessed 2021-01-14].

[303] C. R. Harris, K. J. Millman, S. J. van der Walt, et al. Array programming with NumPy. *Nature*, 585:357–362, 2020.

[304] F. Pérez and B. E. Granger. IPython: a system for interactive scientific computing. *Comput Sci Eng*, 9:21–29, 2007.

[305] M. Abadi. TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems. URL: **https://www.tensorflow.org/**, 2015. [Accessed 2021-01-14].

[306] F. Chollet et al. Keras. URL: **https://keras.io**, 2015. [Accessed 2021-01-14].

[307] F. Pedregosa, G. Varoquaux, A. Gramfort, et al. Scikit-learn: Machine Learning in Python. *J Mach Learn Res*, 12:2825–2830, 2011.

[308] J. D. Hunter. Matplotlib: A 2D Graphics Environment. *Comput Sci Eng*, 9:90–95, 2007.

[309] M. Amadasun and R. King. Textural features corresponding to textural properties. *IEEE Trans Syst Man Cybern Syst*, 19:1264–1274, 1989.

[310] M. M. Galloway. Texture analysis using gray level run lengths. *Comput Gr Image Process*, 4:172–179, 1975.

[311] G. Thibault, B. Fertil, C. Navarro, et al. Texture indexes and gray level size zone matrix. Application to cell nuclei classification. In *10th International Conference on Pattern Recognition and Information Processing, PRIP 2009*, pages 140–145, 2009.

[312] W. Shen, M. Zhou, F. Yang, C. Yang, and J. Tian. Multi-scale convolutional neural networks for lung nodule classification. *Inf Process Med Imaging*, 24:588–599, 2015.

Appendix A

Software and Hardware Used

Here we list the most relevant software and hardware used during this thesis. Different software versions may have been used at some points during the thesis (we did not note if or when software was updated), but these are the most relevant versions for most of the work undertaken:

A.1 Software

Nvidia CUDA Toolkit 10.1.243 Enables GPU-based computation [301]

python 3.8.3 Programming language used throughout [302]

NumPy 1.18.5 Python library adding support for large, multi-dimensional arrays and matrices [303]

IPython 7.15.0 Interactive command shell for python [304]

TensorFlow 2.2.2 Interface for executing machine learning algorithms, used as backend to Keras [305]

Keras 2.4.3 All neural networks were built and run using the Keras interface [306]

scikit-learn 0.17.2 Software library for various machine learning algorithms [307]

Matplotlib 3.2.2 All graphs were created using Matplotlib [308]

LIFEx 6.30 Used to visualise medical images [162]

PyRadiomics 2.2.0 Tool used to extract radiomic features from images [246]

keras-vis 0.5.0 Python toolkit used to create Grad-CAM maps from CNNs [256]

A.2 Hardware

Neural networks were trained using a Nvidia GeForce RTX 2080Ti graphics card.

Appendix B

Radiomic Features

Here we give an overview of the radiomic features used in the thesis. Table B.1 gives a full list of the features used, and full mathematical formulations can be found in the PyRadiomics documentation [34]. These are mostly based on IBSI feature definitions [31]. Unless otherwise stated we used the default PyRadiomic settings for feature calculation.

B.1 Radiomic Features

2D Shape Features These features describe the shape and size of a 2D region of interest (ROI). They are thus independent of the intensity levels inside the ROI.

3D Shape Features These features describe the shape and size of a 3D ROI. They are thus independent of the intensity levels inside the ROI.

First Order Features First-order (also called histogram-based) features describe the distribution of voxel intensities within the ROI. They are global features and independent of voxel locations.

Grey Level Co-occurence Matrix Features Originally proposed in 1973 [260], grey level co-occurence matrix (GLCM) features aim to capture spatial relationships between voxels. Let P_{ij} be the GLCM of a quantised ROI. The $(i,j)^{\text{th}}$ element of this matrix represents the number of times that voxels of intensity i are neighbours with voxels of intensity j. The GLCM is thus a symmetric matrix of size $N_I \times N_I$ where N_I is the number of quantised intensity levels. We define any adjacent voxel as neighbouring (making eight neighbours in 2D and 26 neighbours in 3D) and do not weight the neighbours by distance from the central voxel.

GLCM features can be calculated from this matrix. A completely uniform image with no intensity variation would give a diagonal GLCM. As image texture increases (i.e. as local variations between voxels increase) the off-diagonal elements of the GLCM become larger. GLCM features therefore capture the heterogeneity of an ROI. The scale of the texture features is local (because the GLCM only captures relationships between neighbouring voxels).

Neighbouring Grey Tone Difference Matrix Features The neighbouring grey tone difference matrix (NGTDM) was first proposed in 1989 [309]. It quantifies the difference between a grey value and the average grey value of its neighbours. For a voxel v with N_n neighbours v_n, the average grey level difference (GLD) is defined as

$$GLD = I(v) - \frac{1}{N_n}\sum_{v_n} I(v_n) \tag{B.1}$$

where $I(v)$ is the intenity of voxel v. Each entry P_i in the NGTDM is then the sum of all GLDs for voxels of intensity i in the ROI. The NGTDM is therefore a 1D matrix of size N_I, the number of quantised intensity levels.

Grey Level Dependence Matrix Features The grey level dependence matrix (GLDM) quantifies grey level dependencies in an ROI. A grey level dependency is defined as the number of neighbouring voxels that are dependent on the centre voxel. In our calculations we consider voxels dependent if they have the same intensity values. The $(i,j)^{\text{th}}$ element of a GLDM P_{ij} describes the number of times a voxel with grey level i and j dependent voxels appears in the ROI.

NGTDM and GLDM features measure the magnitude of changes in intensity between neighbouring voxels and can therefore be considered local.

Grey Level Run-Length Matrix Features Grey level run-length matrix (GLRLM) features were originally proposed in 1975 [310] to quantify grey level runs, defined as the number of consecutive voxels that have the same grey level value (i.e. intensity). In a GLRLM matrix $P_{ij}(\theta)$ the $(i,j)^{\text{th}}$ element represents the number of runs with grey level i and length j in direction θ. The GLRLM is of size $N_I \times N_L$ where N_I is the number of quantised intensity levels and N_L is the length of the longest grey level run. In this thesis we calculate the value for each direction separately then use the mean, without weighting by distance.

Grey Level Size Zone Matrix Features Grey level size zone matrix (GLSZM) features were first used to classify cell nuclei in 2009 [311]. Similar to the GLRLM, the GLSZM counts the number of zones of voxels with the same grey level. For a GLZSM matrix P_{ij} the $(i,j)^{\text{th}}$ element gives the number of zones of grey level i with size j from the quantised ROI. A grey level zone is defined as the number of connected voxels that share the same grey level intensity. The GLSZM is therefore of size $N_I \times N_Z$ where N_I is the number of quantised intensity levels and N_Z is the size of the largest grey level zone.

Features derived from the GLRLM and GLSZM are influenced by the size and shape of homogeneous regions within an ROI and can therefore be considered regional.

B.2 Filter-Based Features

Filters can be applied before feature extraction to extract different information. We use the following filters:

Exponential Filter The exponential filter applies the transformation:

$$f(x) = e^{cx} \tag{B.2}$$

where:

$$c = \frac{\log(\max(|x|))}{\max(|x|)} \tag{B.3}$$

and x is the original voxel intensity.

Gradient Filter Applying the gradient filter gives the gradient of the image.

Laplacian of Gaussian Filter The Laplacian of Gaussian (LoG) filter convolves the image with the second derivative of the Gaussian kernel. The Gaussian kernel is defined as

$$G(x,y,z,\sigma) = \frac{1}{(\sigma\sqrt{2\pi})^3} e^{-\frac{x^2+y^2+z^2}{2\sigma^2}} \tag{B.4}$$

The LoG kernel $\nabla^2 G(x,y,z,\sigma)$ enhances the edges of an image. Different values of σ emphasise finer or coarser textures.

Wavelet Filter Wavelet filters decompose an image using a discrete wavelet transform. In this thesis we use level one *coif* wavelet decompositions. Low and high-pass filters are denoted L and H respectively.

Table B.1 Texture features by feature type. Full mathematical details of the features can be found at [34]

Feature Type	Features
Shape (2D)	Mesh Surface, Pixel Surface, Perimeter, Perimeter to Surface Ratio Sphericity, Spherical Disproportion, Maximum 2D Diameter, Major Axis Length, Minor Axis Length, Elongation
Shape (3D)	Mesh Volume, Voxel Volume, Surface Area, Surface Area to Volume Ratio, Sphericity, Compactness_1, Compactness_2, Spherical Disproportion, Maximum 3D Diameter, Maximum 2D Diameter (by slice, column, row), Major Axis Length, Minor Axis Length, Least Axis Length, Elongation, Flatness
First-Order	Energy, Total Energy, Entropy, Minimum, 10^{th} Percentile, 90^{th} Percentile, Maximum, Mean, Median, Interquartile Range, Range, Mean Absolute Deviation, Robust Mean Absolute Deviation, Root Mean Squared, Standard Deviation, Skewness, Kurtosis, Variance, Uniformity
GLCM	Autocorrelation, Joint Average, Cluster Prominence, Cluster Shade Cluster Tendency, Contrast, Correlation, Difference Average, Difference Entropy, Difference Variance, Joint ENergy, Joint Entropy, Informational Measure of Correlation 1, Information Measure of Correlation 2, Inverse Difference Moment, Maximal Correlation Coefficient, Inverse Difference Moment Normalised, Inverse Difference, Inverse Difference Normalised, Inverse Variance, Maximum Probability, Sum Average, Sum Entropy, Sum of Squares
NGTDM	Coarseness, Contrast, Busyness, Complexity, Strength
GLRLM	Short Run Emphasis, Long Run Emphasis, Grey Level Non-Uniformity, Grey Level Non-Uniformity Normalised, Run Length Non-Uniformity, Run Length Non-Uniformity Normalised, Run Percentage, Grey Level Variance, Run Variance, Run Entropy, Low Grey Level Run Emphasis, High Grey Level Run Emphasis, Short Run Low Grey Level Run Emphasis, Short Run High Grey Level Emphasis, Long Run Low Grey Level Emphasis, Long Run High Grey Level Emphasis
GLSZM	Small Area Emphasis, Large Area Emphasis, Grey Level Non-Uniformity, Grey Level Non-Uniformity Normalised, Size-Zone Non-Uniformity, Size-Zone Non-Uniformity Normalised, Zone Percentage, Grey Level Variance, Zone Variance, Zone Entropy, Low Grey Level Zone Emphasis, High Grey Level Zone Emphasis, Small Area Low Grey Level Emphasis, Small Area High Grey Level Emphasis, Large Area Low Grey Level Emphasis, Large Area High Grey Level Emphasis
GLDM	Small Dependence Emphasis, Large Dependence Emphasis, Grey Level Non-Uniformity, Dependence Non-Uniformity, Dependence Non-Uniformity Normalised, Grey Level Variance, Dependence Variance, Dependence Entropy, Low Grey Level Emphasis, High Grey Level Emphasis, Small Dependence Low Grey Level Emphasis, Small Dependence High Grey Level Emphasis, Large Dependence Low Grey Level Emphasis, Large Dependence High Grey Level Emphasis

Appendix C

CNN Architecture Details

Full details for all models used in the thesis are below. In all cases the convolutional kernels were of dimension 3 (i.e. 3x3 in 2D and 3x3x3 in 3D) with a stride of 1. The pooling kernels were of dimension 2 with a stride of 2.

C.1 Mediastinal - MIP Approach Models

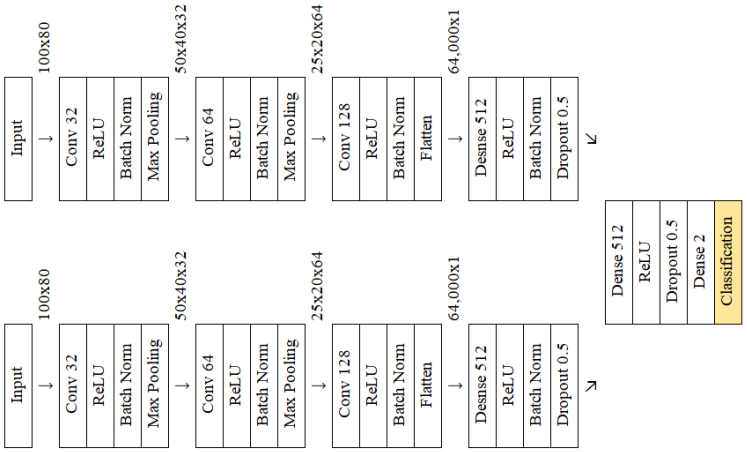


Fig. C.1 MIP_1

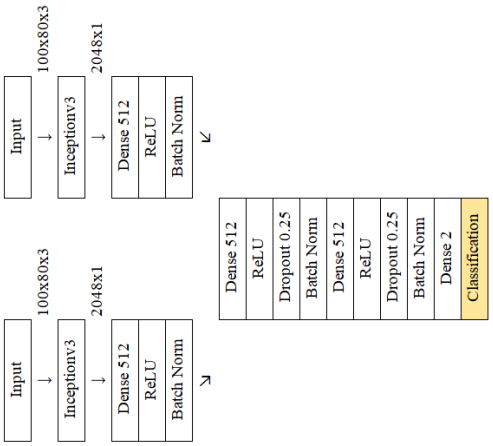


Fig. C.2 MIP_2

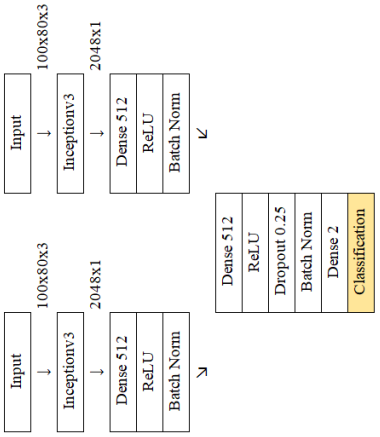


Fig. C.3 MIP_3

C.2 Mediastinal - Self-Supervised Jigsaw Task Models

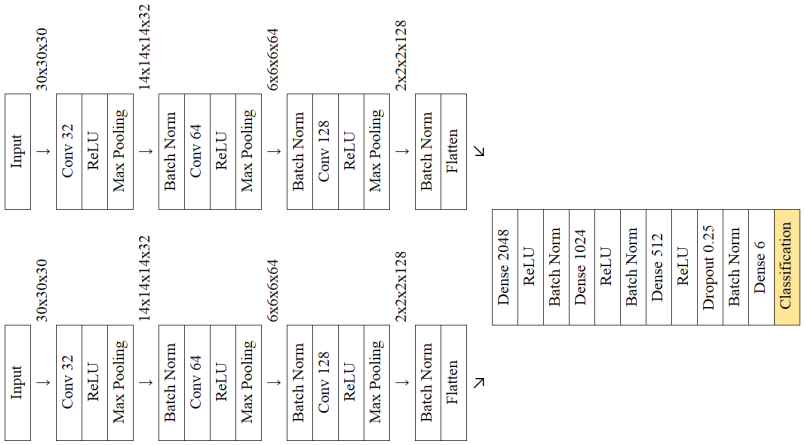


Fig. C.4 Jigsaw$_1$

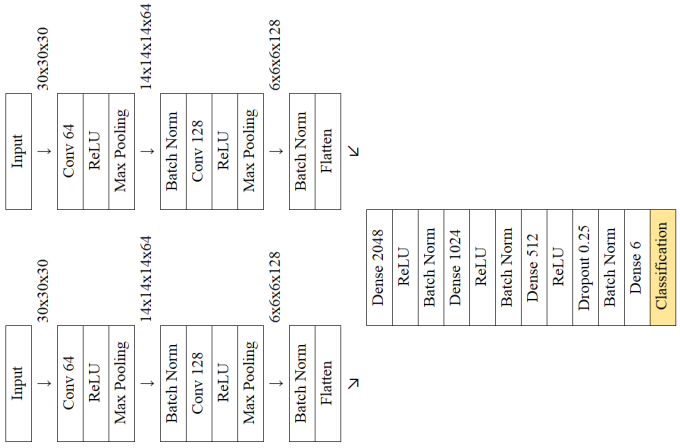


Fig. C.5 Jigsaw$_2$

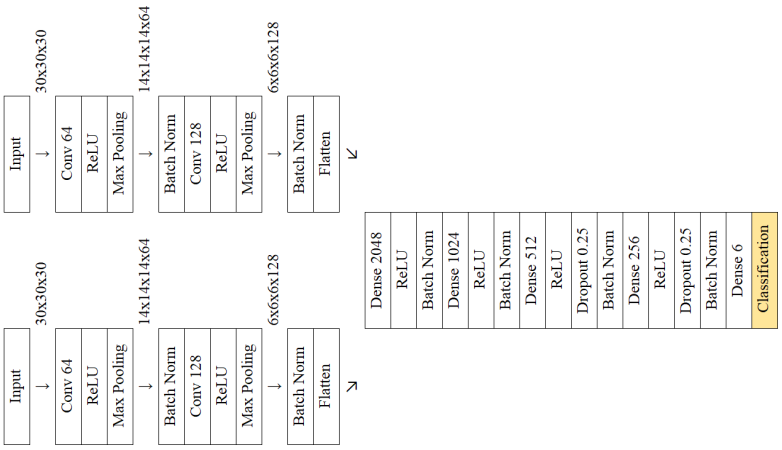


Fig. C.6 Jigsaw$_3$

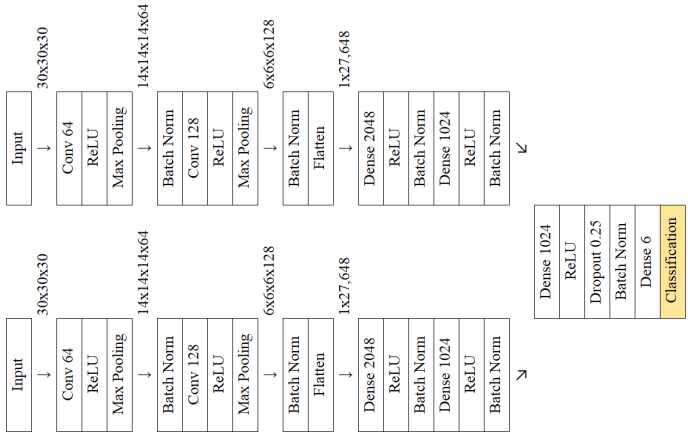


Fig. C.7 Jigsaw$_4$

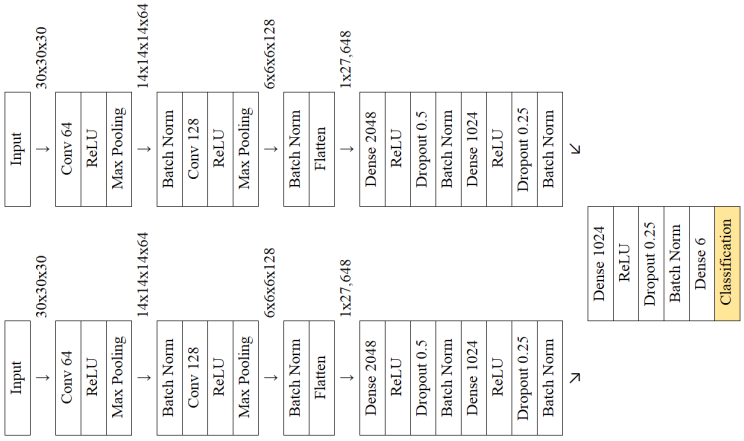


Fig. C.8 Jigsaw$_5$

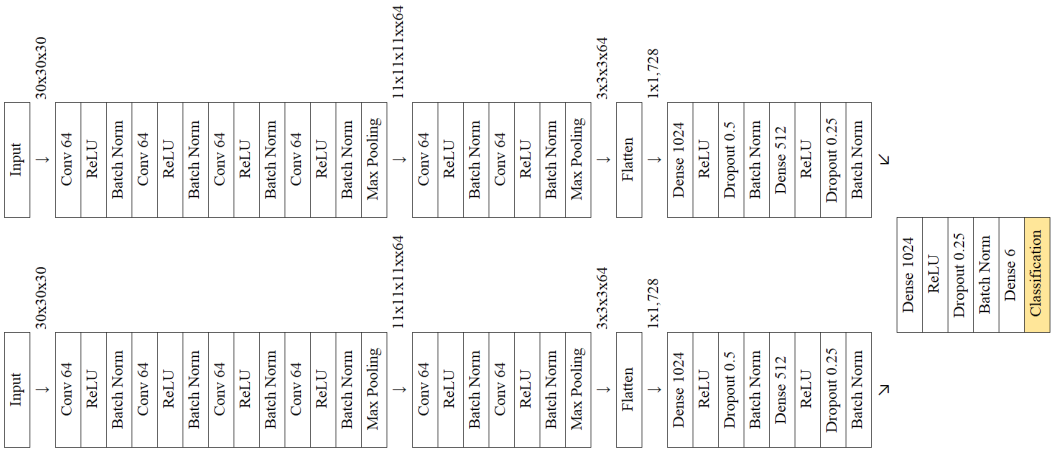


Fig. C.9 Jigsaw$_6$

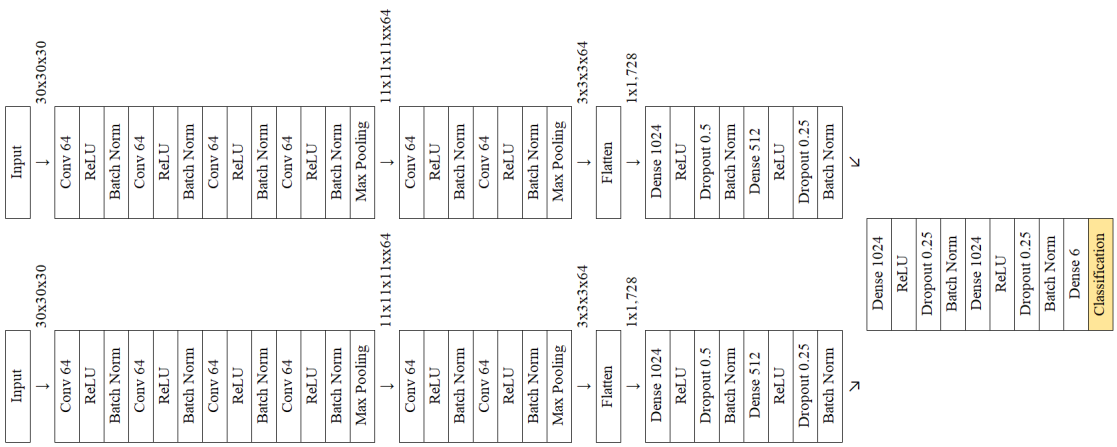


Fig. C.10 Jigsaw$_7$

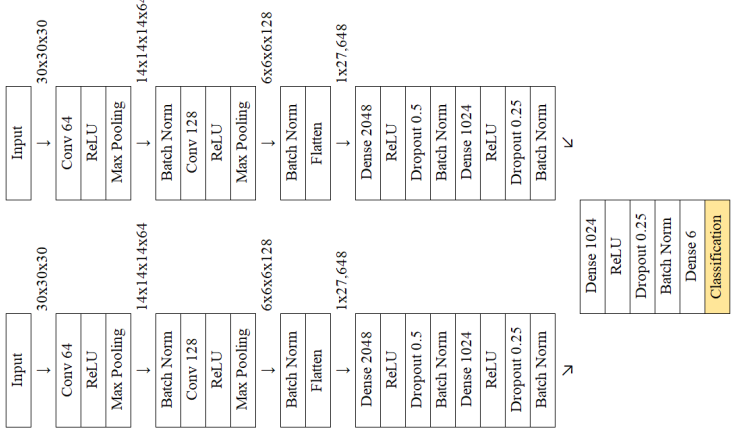


Fig. C.11 Jigsaw$_8$

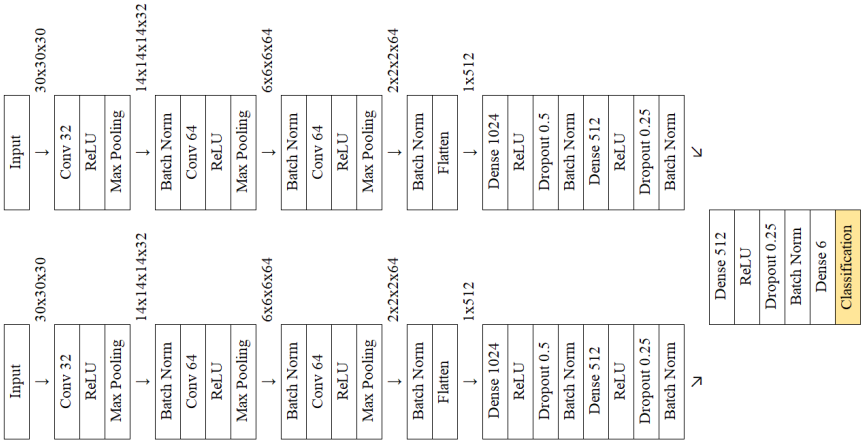


Fig. C.12 Jigsaw$_9$

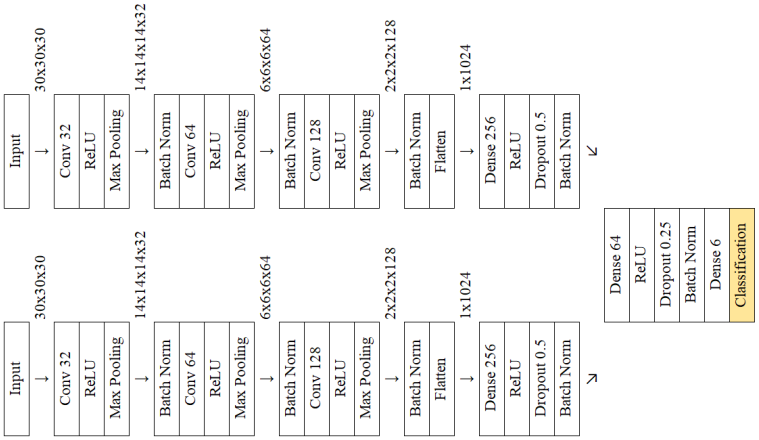


Fig. C.13 Jigsaw$_{10}$

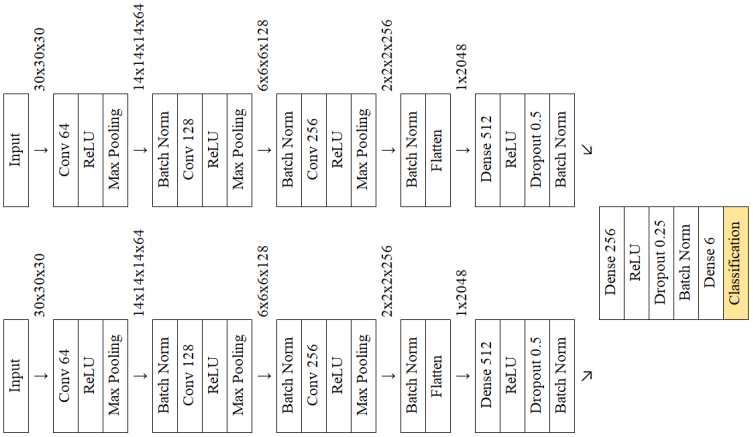


Fig. C.14 Jigsaw$_{11}$

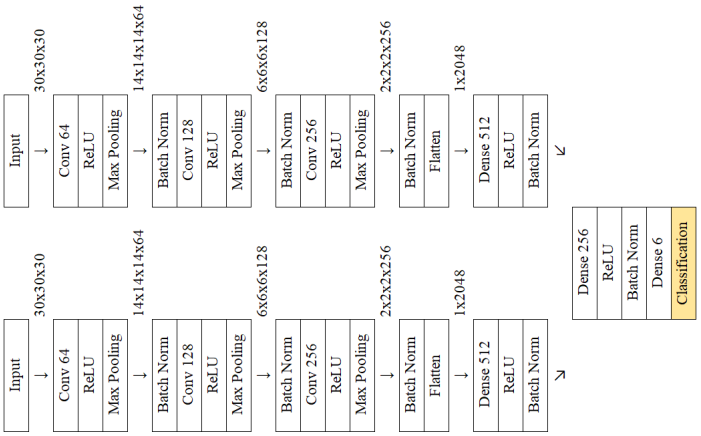


Fig. C.15 Jigsaw$_{12}$

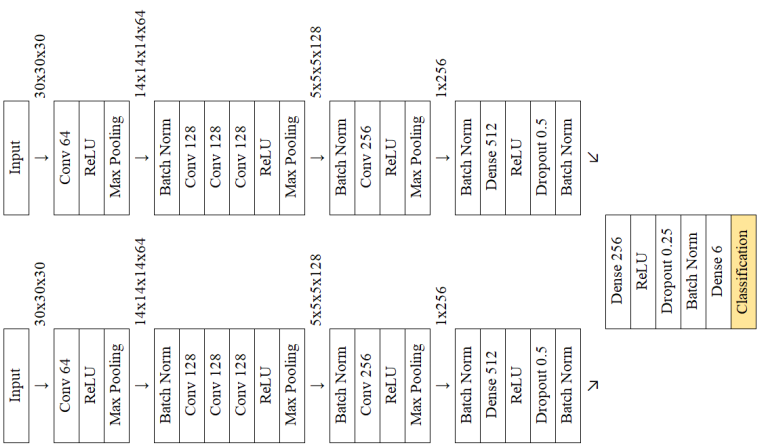


Fig. C.16 Jigsaw$_{13}$

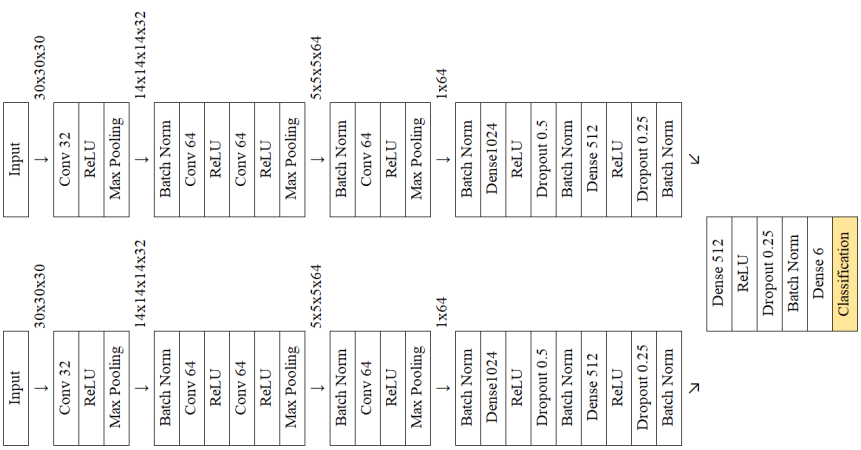


Fig. C.17 Jigsaw$_{14}$

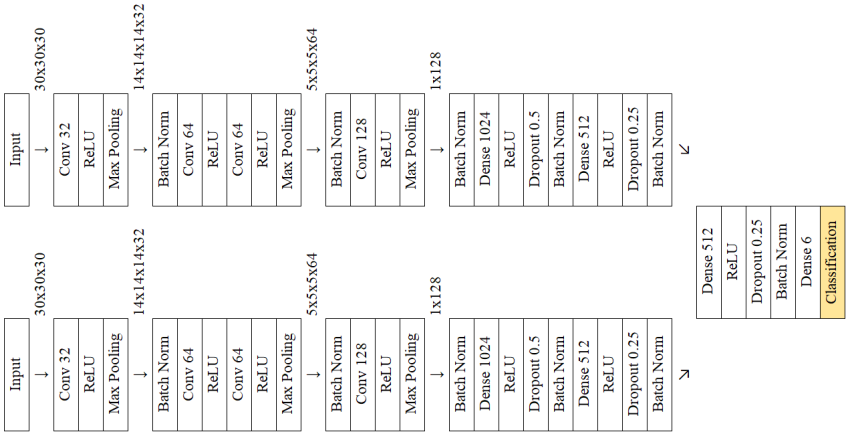


Fig. C.18 Jigsaw$_{15}$

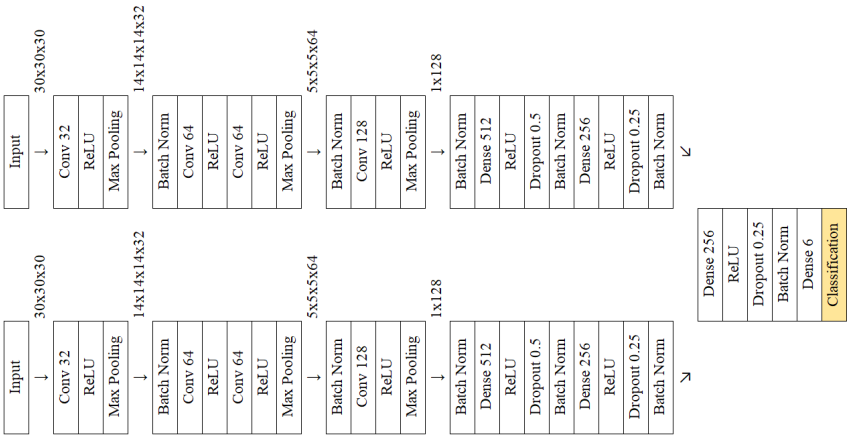


Fig. C.19 Jigsaw$_{16}$

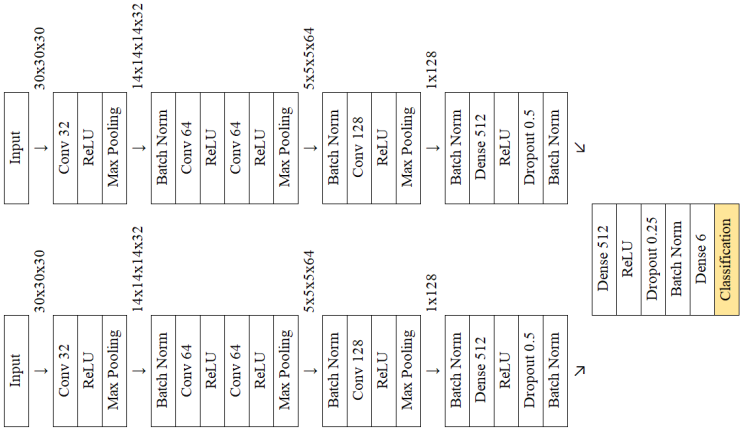


Fig. C.20 Jigsaw$_{17}$

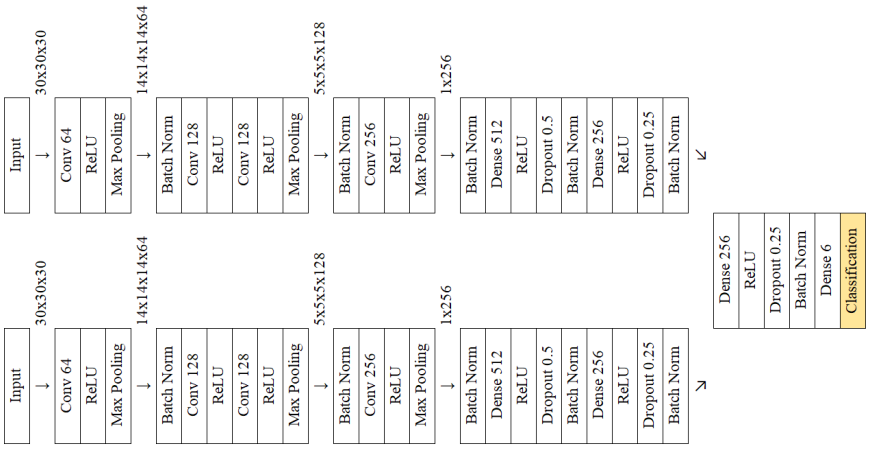


Fig. C.21 Jigsaw$_{18}$

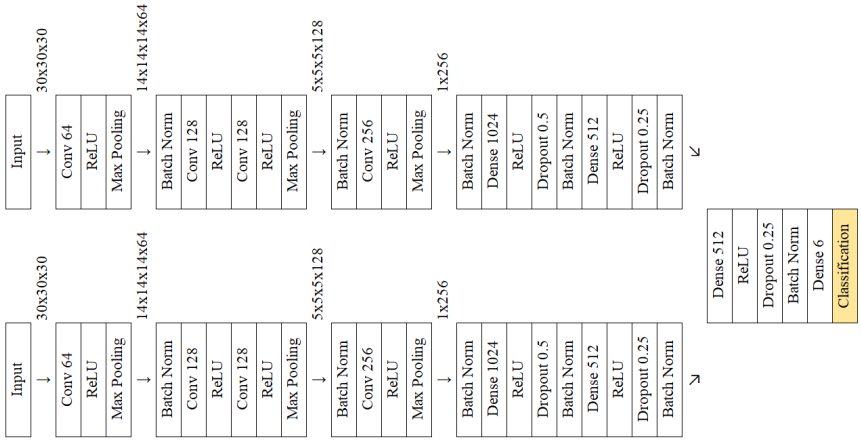


Fig. C.22 Jigsaw$_{19}$

C.3 Mediastinal - One Phase 3D Cube Approach Models

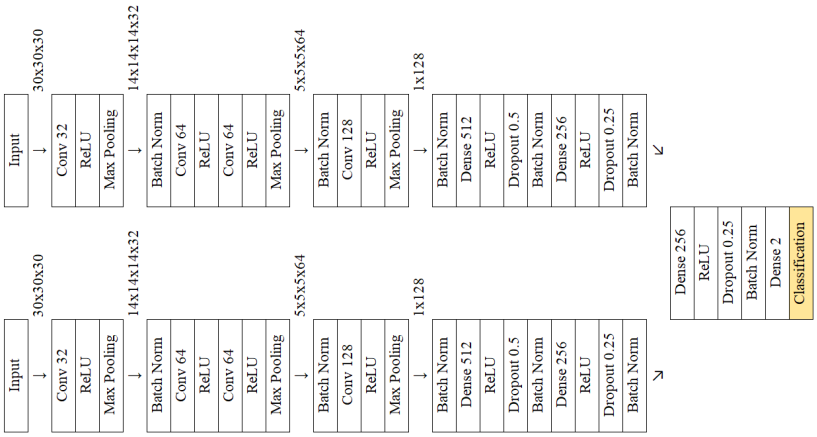


Fig. C.23 Cube_1

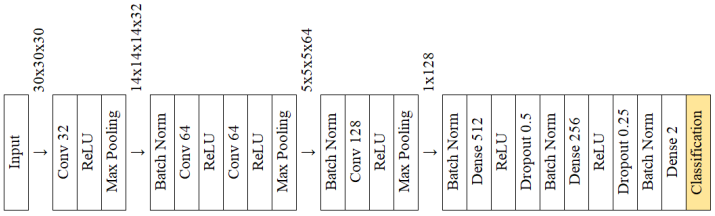


Fig. C.24 Cube_2

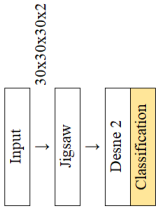


Fig. C.25 Cube_3

C.4 Mediastinal - Phase Two Models

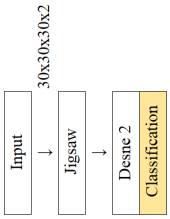


Fig. C.26 Phase2$_1$

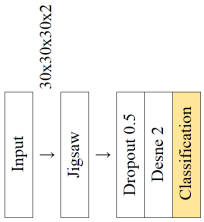


Fig. C.27 Phase2$_2$

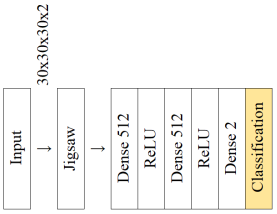


Fig. C.28 Phase2$_3$

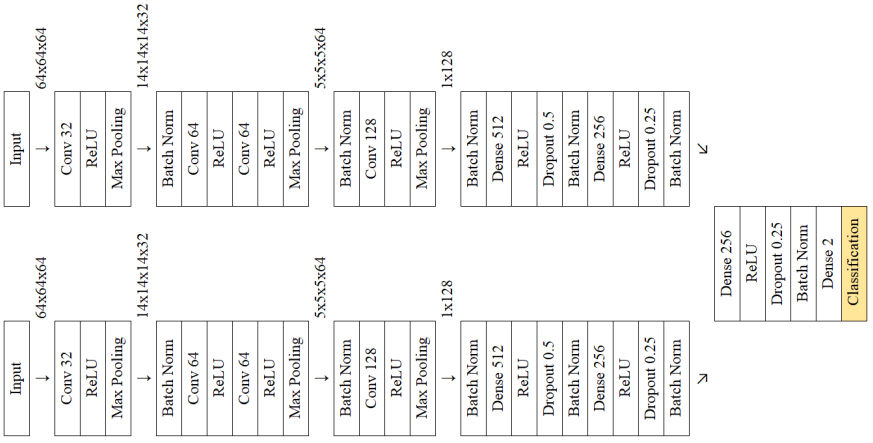


Fig. C.29 Phase2$_4$

C.5 Cardiac Models

Fig. C.30 Cardiac$_1$

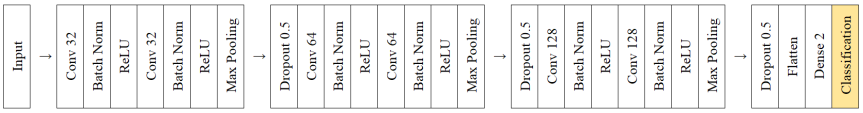


Fig. C.31 Cardiac$_2$

Appendix D

Multi-Scale Approach - Mediastinal Lymph Node Detection

D.1 Introduction

In 5.12 we ran several experiments building CNNs to classify 3D cubes as positive or negative, indicating the presence of a pathological mediastinal lymph node within each cube. We tested CNNs trained on different cube sizes. Other studies have proposed models incorporating different length scales, allowing small-scale and large-scale features to be captured [312]. To test if this could improve the performance of our model, we combined models trained on different cube sizes.

D.2 Method

Models trained on cubes of side length 20 mm, 30 mm, and 50 mm were selected (corresponding to tests 23, 26, and 27 from 5.12). Features were extracted from the final layer of the CNN in each case, giving a total of 256 features for each cube.

In the individually trained models, training cubes were marked as positive if there was a node within the cube. This meant that the labels for the different cube sizes were not equivalent (a node could be within the boundaries of a larger cube but not those of a smaller cube). To train the multi-scale model we used the labels for the 20 mm cubes.

These features were used to train SVM models. To ensure a fair comparison we also trained SVMs using features from single cube sizes. These results could then be compared to assess the benefit of the multi-scale approach. The number of features was first reduced to 140 using an ANOVA F-test, then any features with correlations greater than 0.95 were

Table D.1 Results - multi-scale approach. AUC values calculated on the test set are shown for models trained on features from the three individual cube sizes and on the combined feature set

Cube side length/mm	AUC
20	0.75
30	0.81
50	0.84
Combined	0.85

removed (as in 5.8.2). We used linear kernels with a C value of 1. Finally, the AUC was calculated on the test set.

D.3 Results

Results are shown in Table D.1. We did not explicitly calculate confidence intervals, but from other tests differences of a few hundredths are not significant. We therefore conclude that combining cubes from different length scales did not significantly improve the performance.

université PARIS-SACLAY
ÉCOLE DOCTORALE
Physique et ingénierie : Electrons, Photons, Sciences du vivant (EOBE)

Titre : Une étude des méthodes d'apprentissage automatique et d'apprentissage profond et de leur application à l'imagerie médicale

Mots clés : Imagerie Médicale, Intelligence Artificielle, Apprentissage Profond, Radiologie, Oncologie

Résumé : Nous utilisons d'abord des réseaux neuronaux convolutifs (CNNs) pour automatiser la détection des ganglions lymphatiques médiastinaux dans les images TEP/TDM. Nous construisons un modèle entièrement automatisé pour passer directement des images TEP/TDM à la localisation des ganglions. Les résultats montrent une performance comparable à celle d'un médecin. Dans la seconde partie de la thèse, nous testons la performance, l'interprétabilité et la stabilité des modèles radiomiques et CNN sur trois ensembles de données (IRM cérébrale 2D, TDM pulmonaire 3D, TEP/TDM médiastinale 3D). Nous comparons la façon dont les modèles s'améliorent lorsque davantage de données sont disponibles et nous examinons s'il existe des tendances communes aux différents problèmes. Nous nous demandons si les méthodes actuelles d'interprétation des modèles sont satisfaisantes. Nous étudions également comment une segmentation précise affecte les performances des modèles.

Nous utilisons d'abord des réseaux neuronaux convolutifs (CNNs) pour automatiser la détection des ganglions lymphatiques médiastinaux dans les images TEP/TDM. Nous construisons un modèle entièrement automatisé pour passer directement des images TEP/TDM à la localisation des ganglions. Les résultats montrent une performance comparable à celle d'un médecin. Dans la seconde partie de la thèse, nous testons la performance, l'interprétabilité et la stabilité des modèles radiomiques et CNN sur trois ensembles de données (IRM cérébrale 2D, TDM pulmonaire 3D, TEP/TDM médiastinale 3D). Nous comparons la façon dont les modèles s'améliorent lorsque davantage de données sont disponibles et nous examinons s'il existe des tendances communes aux différents problèmes. Nous nous demandons si les méthodes actuelles d'interprétation des modèles sont satisfaisantes. Nous étudions également comment une segmentation précise affecte les performances des modèles.

Title: A study of machine learning and deep learning methods and their application to medical imaging

Keywords: Medical Imaging, Artificial Intelligence, Deep Learning, Radiology, Oncology

Abstract: We first use Convolutional Neural Networks (CNNs) to automate mediastinal lymph node detection using FDG-PET/CT scans. We build a fully automated model to go directly from whole-body FDG-PET/CT scans to node localisation. The results show a comparable performance to an experienced physician. In the second half of the thesis we experimentally test the performance, interpretability, and stability of radiomic and CNN models on three datasets (2D brain MRI scans, 3D CT lung scans, 3D FDG-PET/CT mediastinal scans). We compare how the models improve as more data is available and examine whether there are patterns common to the different problems. We question whether current methods for model interpretation are satisfactory. We also investigate how precise segmentation affects the performance of the models.

We first use Convolutional Neural Networks (CNNs) to automate mediastinal lymph node detection using FDG-PET/CT scans. We build a fully automated model to go directly from whole-body FDG-PET/CT scans to node localisation. The results show a comparable performance to an experienced physician. In the second half of the thesis we experimentally test the performance, interpretability, and stability of radiomic and CNN models on three datasets (2D brain MRI scans, 3D CT lung scans, 3D FDG-PET/CT mediastinal scans). We compare how the models improve as more data is available and examine whether there are patterns common to the different problems. We question whether current methods for model interpretation are satisfactory. We also investigate how precise segmentation affects the performance of the models.