Faculty of Electrical Engineering, Computer Science
and Information Technology Osijek,
J. J. Strossmayer University of Osijek, Croatia

Faculty of Engineering and Architecture,
Ghent University, Belgium

MARIN BENČEVIĆ

Data Efficient Deep Learning Models for Biomedical Image Segmentation

Doctoral dissertation submitted to obtain the academic degrees of
Doctor of Science, scientific area of Technical Sciences,
scientific field of Computer Science at J. J. Strossmayer University of Osijek,
and Doctor of Computer Science Engineering at Ghent University

Osijek, Croatia, July 5, 2024

Data Efficient Deep Learning Models for Biomedical Image Segmentation

Podatkovno učinkoviti modeli dubokog učenja za segmentaciju biomedicinskih slika

Data-efficiënte deeplearning-modellen voor biomedische beeldsegmentatie

Marin Benčević, © February, 2023.

This doctoral dissertation was made at the Department of Software Engineering of the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, J. J. Strossmayer University of Osijek, Croatia, and the Department of Telecommunications and Information Processing of the Faculty of Engineering and Architecture, Ghent University, Belgium.

SUPERVISORS

prof. Irena Galić, Ph.D.[1]
prof. Aleksandra Pižurica, Ph.D.[2]
Danilo Babin, Ph.D.[2]

[1] Faculty of Electrical Engineering, Computer Science and Information Technology Osijek, J. J. Strossmayer University of Osijek, Croatia

[2] Department of Telecommunications and Information Processing, Ghent University, Belgium

Members of the examination board

Chairs

prof. Krešimir Nenadić, Ph.D., J. J. Strossmayer University of Osijek
prof. Patrick De Baets, Ph.D., Ghent University

Other members entitled to vote

prof. Hrvoje Leventić, Ph.D., J. J. Strossmayer University of Osijek
prof. Jan Aelterman, Ph.D., Ghent University
Srđan Lazendić, Ph.D., Ghent University
prof. Ivica Botički, Ph.D., University of Zagreb
prof. Peter Peer, Ph.D., University of Ljubljana

This thesis was defended in Osijek, Croatia, on 2 July 2024.

Acknowledgements

Ne boj se, nisi sam! Ima i drugih nego ti
koji nepoznati od tebe žive tvojim životom.
I ono sve što ti bje, ču i što sni
gori u njima istim žarom, ljepotom i čistotom.

Ne gordi se! Tvoje misli nisu samo tvoje! One u drugima žive.
Mi smo svi prešli iste putove u mraku,
mi smo svi jednako lutali u znaku
traženja, i svima jednako se dive.

—*Tin Ujević*

My view of the scientific process is that of stochastic gradient descent — we are all clumsily stumbling down the gradient of ignorance. I hope that this thesis presents a small incremental step in the right direction. If not, then at least I hope you find the local minima discovered here useful in your own stumbling.

Of course, like all things, this thesis only arose as a result of sufficient conditions, including people. I can only take partial credit, so here I would like to acknowledge a small subset of the people who made this thesis possible.

First, I would like to thank you supervisor in Osijek prof. Irena Galić for always providing support, teaching me how to navigate the academic environment, and often saving me from my own bad decisions. I would also like to thank my supervisor in Ghent prof. Aleksandra Pižurica for lending her great expertise. I thank my other supervisor in Ghent, dr. Danilo Babin, for always putting my work in the context of medical practice and expanding my professional network.

Thank you to all of the members of the examination board for taking the time out of your busy schedules to provide valuable feedback and to participate in the evaluation process.

Alongside my official mentors, over the years I have picked up a few unofficial ones as well. My office mate and colleague Marija Habijan was always there to patiently answer my endless questions and provide support. My dear friend Matej Džijan, whether he liked it or not, was on the receiving end of many half-thought-out bad ideas that he promptly challenged and improved. Thank you both!

I would also like to thank the friends, coauthors, and colleagues who joined me on my Ph.D. journey, including H. Leventić, K. Romić and R. Šojo from the Research Group for Computer Science and Human-Computer Interaction in Osijek, as well as S. Lazendić, N. Vercheval, N. Žižakić, N. Nadisic, Y. Arhant, Y. Qiu, and X. Li from the Group for Artificial Intelligence and Sparse Modelling in Ghent.

I would like to acknowledge the support of FERIT for employing me during my Ph.D. and all of its staff for their support. This work has also been supported in part by the Croatian Science Foundation under the project UIP-2017-05-4968.

Thank you to my parents and family for always being there for me. Finally, I would like to thank my wife Dora for making my life infinitely more joyous.

This thesis is dedicated to my favorite person in the world, my son Javor.

Marin Benčević
January, 2024.

Abstract

Medical image segmentation plays a pivotal role in various medical applications from surgical planning to diagnosis and research. Typically, medical image segmentation heavily relies on neural networks to achieve accurate segmentation across different imaging modalities such as CT, MRI, microscopy, dermatoscopy, and others. However, the effectiveness of neural networks is significantly influenced by the availability and quality of training data, which is often challenging to acquire due to the high time and financial cost of image acquisition, invasiveness of some imaging procedures, large file sizes, and regulatory challenges. The labor-intensive and expert-driven nature of annotating medical images for segmentation further compounds these challenges, making the assembly of large, high-quality datasets for training medical image segmentation models particularly difficult.

Statistical learning theory principles indicate that more complex medical image segmentation tasks require neural networks with a greater number of parameters for effective segmentation. This requirement for more parameters, in turn, demands larger sample sizes to avoid overfitting. In medical imaging, however, there is a lack of datasets with large sample sizes. Therefore, there is a large need for data-efficient segmentation methods that can deliver reliable results with limited training samples.

In this thesis, we present an overview of medical image segmentation methods and existing strategies for improving their data efficiency. In addition, we propose various new methods that simplify the segmentation task, allowing convolutional neural networks to perform accurate segmentation with fewer parameters and, by extension, smaller sample sizes. Our methods center on transforming the segmentation boundary into a representation that can modeled with fewer parameters. We do so by leveraging domain knowledge and traditional image-processing techniques to identify beneficial image transformations. The parameters of the image transformations are dynamically selected for each image using neural networks, breaking down the segmentation task into two more manageable stages: an initial rough localization of the target object followed by the segmentation of a simplified representation of the image.

More specifically, we use two neural networks to tackle segmentation tasks effectively. The first network focuses on roughly locating the target object, using either Gaussian distributions centered on the object or simple segmentation maps as initial guides. We construct a function that, given this rough localization, produces parameters for the image transformation aimed at simplifying the segmentation task. The decision-making process for the function and the image transformations relies on both domain knowledge and empirical results. For example, to segment elliptical objects more efficiently, we apply the polar transform with the polar origin positioned at the object's center. The image undergoes the image transformation according to the resulting parameters, preparing it for the second neural network, which performs the detailed segmentation. This second network is specifically trained on transformed images using parameters obtained from ground truth segmentation masks.

By training the second network on transformed images, we simplify the segmentation boundary and thus can use networks with fewer parameters for the task. Networks with fewer parameters need less data to train, and so our approach naturally leads to

increases in data efficiency. Our results demonstrate comparable or improved accuracy using fewer data samples or labels across various medical imaging applications, such as segmentation of liver and epicardial adipose tissue from CT scans, skin lesions, polyps from colonoscopy images, and cells from microscopic images.

The two networks can be separate neural networks or they can be connected and trained end-to-end. A notable benefit of this approach is that the rough localization and fine segmentation networks can share the same commonly used medical image segmentation architecture. This allows for easy transfer learning between the two networks, speeding up training times and improving performance. In addition, the networks can be pretrained using existing available medical image segmentation datasets or networks.

In addressing the challenge of data efficiency in medical image segmentation, this thesis introduces the following original scientific contributions:

1. **A new biomedical image segmentation method based on polar transform preprocessing with a learned center point.** We introduce a novel preprocessing technique for medical images, particularly those with elliptically shaped objects. By applying a polar transform to the image, we simplify circular decision boundaries into linear ones, making segmentation more straightforward. A key contribution is developing a neural network to identify the optimal origin for the polar transformation, enhancing segmentation performance and allowing the use of less complex networks.

2. **An improved method of reducing the input image size in neural networks using salient image crops.** Building on the insights of model-driven polar transformations, we propose a model-driven cropping technique to minimize neural network input sizes without sacrificing fine details. By localizing the target object in a downsampled image and extracting identified regions of interest from a high-resolution image, we maintain precise segmentation with smaller network input sizes. Since model complexity increases with image size, this reduction in input sizes improves data efficiency.

3. **A new neural network architecture for high-resolution image segmentation that combines object detection in low-resolution images and segmentation in high-resolution images.** We extend our preprocessing methods to create an end-to-end trainable network that combines low-resolution object localization with high-resolution image segmentation. We allow network convergence in two key ways. First, we use a shared architecture between the localization and segmentation subnetworks which allows transfer learning. Secondly, we ensure gradient flow throughout the network by passing information from one subnetwork to the other. We show that training this end-to-end network increases the robustness of both the localization and segmentation stages.

4. **A new method of embedding depth information in two-dimensional convolutional neural network input data.** Acknowledging the data efficiency limitations of 3D networks for volumetric data like CT scans, we develop a method to embed depth information into 2D slices by adding a normalized z-coordinate channel to each slice. We show that this allows effective segmentation of CT images with slice-based 2D networks.

The effectiveness of these methods is validated across various medical imaging modalities, including CT scans, microscopy, dermatoscopy, and colonoscopy, showing

not only enhanced data efficiency but also achieving state-of-the-art segmentation results in certain tasks. The techniques introduced are versatile, suitable for a broad spectrum of medical imaging fields, and can serve as general preprocessing steps for any convolutional neural network-based segmentation architecture.

The research presented in this thesis has resulted in the publication of four research papers in scientific journals (all as first author) and five papers presented at international scientific conferences (of which four as first author).

Samenvatting

Medische beeldsegmentatie speelt een centrale rol in verschillende medische toepassingen, van chirurgische planning tot diagnose en onderzoek. Over het algemeen steunt medische beeldsegmentatie sterk op neurale netwerken om nauwkeurige segmentatie te bereiken bij verschillende beeldmodaliteiten zoals CT, MRI, microscopie, dermatoscopie en andere. De effectiviteit van neurale netwerken wordt echter aanzienlijk beïnvloed door de beschikbaarheid en kwaliteit van training data, die vaak moeilijk te verkrijgen is vanwege de dure en langdurige acquisitie, de invasiviteit van sommige technieken, grote bestandsgroottes en wettelijke uitdagingen. De arbeidsintensieve en deskundige aard van het annoteren van medische beelden voor segmentatie maakt deze uitdagingen nog groter, waardoor het bijzonder moeilijk is om grote datasets van hoge kwaliteit samen te stellen voor het trainen van segmentatiemodellen voor medische beelden.

Statistische leertheoretische principes geven aan dat complexere medische beeldsegmentatie neurale netwerken met een groot aantal parameters vereist voor effectieve segmentatie. Dit vereist op zijn beurt grotere steekproeven om overfitting te voorkomen. In de medische beeldvorming is er echter een gebrek aan datasets van dergelijke omvang. Daarom is er een grote behoefte aan gegevensefficiënte segmentatiemethoden die betrouwbare resultaten kunnen leveren met beperkte training data.

In dit proefschrift presenteren we een overzicht van segmentatiemethoden voor medische afbeeldingen en bestaande strategieën om hun gegevensefficiëntie te verbeteren. Daarnaast stellen we verschillende nieuwe methoden voor die de segmentatietaak vereenvoudigen, waardoor convolutionele neurale netwerken nauwkeurige segmentatie kunnen uitvoeren met minder parameters en, bij uitbreiding, kleinere steekproeven. Onze methoden zijn gericht op het transformeren van de segmentatiegrens in een representatie die met minder parameters gemodelleerd kan worden. We doen dit door domeinkennis en traditionele beeldverwerkingstechnieken te gebruiken om gunstige beeldtransformaties te identificeren. De parameters van de beeldtransformaties worden dynamisch geselecteerd voor elk beeld met behulp van neurale netwerken, waardoor de segmentatietaak wordt opgesplitst in twee gemakkelijkere hanteerbare fasen: een eerste ruwe lokalisatie van het doelobject gevolgd door de segmentatie van een vereenvoudigde weergave van het beeld.

Concreet gebruiken we twee neurale netwerken om segmentatietaken uit te voeren. Het eerste netwerk richt zich op het ruwweg lokaliseren van het doelobject, waarbij Gaussiaanse verdelingen gecentreerd op het object of eenvoudige segmentatiekaarten, als initiële richtlijnen worden gebruikt. We construeren een functie die, gegeven deze ruwe lokalisatie, parameters produceert voor de beeldtransformatie om de segmentatietaak te vereenvoudigen. Het besluitvormingsproces voor de functie en de beeldtransformaties is gebaseerd op zowel domeinkennis als empirische resultaten. Om bijvoorbeeld elliptische objecten efficiënter te segmenteren passen we de polaire transformatie toe, met de polaire oorsprong in het midden van het object. Het beeld ondergaat de beeldtransformatie volgens de resulterende parameters en wordt voorbereid voor het tweede neurale netwerk, dat dan gedetailleerde segmentatie uitvoert. Dit tweede netwerk wordt specifiek getraind op getransformeerde beelden met behulp van parameters verkregen uit segmentatiemaskers ter referentie.

Door het tweede netwerk te trainen op getransformeerde beelden, vereenvoudigen we de segmentatiegrens en kunnen we dus netwerken gebruiken met minder parameters. Netwerken met minder parameters vereisen minder gegevens om te trainen, en dus leidt onze aanpak van nature tot een toename in gegevensefficiëntie. Onze resultaten tonen een vergelijkbare of verbeterde nauwkeurigheid bij gebruik van minder data of labels in verschillende medische beeldvormingstoepassingen, zoals segmentatie van de lever en epicardiaal vetweefsel uit CT-scans, huidlaesies, poliepen uit colonoscopiebeelden en cellen uit microscopische beelden.

De twee neurale netwerken kunnen afzonderlijke netwerken zijn of ze kunnen na verbinding end-to-end worden getraind. Een opmerkelijk voordeel van deze aanpak is dat de ruwe lokalisatie- en fijne segmentatienetwerken dezelfde veelgebruikte architectuur voor medische beeldsegmentatie kunnen delen. Dit maakt transferleren tussen de twee netwerken eenvoudig, wat de trainingstijd verkort en de prestaties verbetert. Bovendien kunnen de netwerken worden voorgetraind met behulp van bestaande beschikbare datasets of netwerken voor medische beeldsegmentatie.

Om de uitdaging van gegevensefficiëntie in medische beeldsegmentatie aan te gaan, introduceert dit proefschrift de volgende originele wetenschappelijke bijdragen:

1. **Een nieuwe biomedische beeldsegmentatiemethode gebaseerd op polaire transformatie voorbewerking met een geleerd middelpunt.** We introduceren een nieuwe voorbewerkingstechniek voor medische beelden met elliptisch gevormde objecten. Door een polaire transformatie op de afbeelding toe te passen, vereenvoudigen we cirkelvormige beslissingsgrenzen in lineaire, waardoor segmentatie eenvoudiger wordt. Een belangrijke bijdrage is de ontwikkeling van een neuraal netwerk om de optimale oorsprong voor de polaire transformatie te identificeren, waardoor de segmentatieprestaties verbeteren en het gebruik van minder complexe netwerken mogelijk wordt.

2. **Een verbeterde methode voor het verkleinen van de invoerafbeelding in neurale netwerken met behulp van saillante bijgesneden afbeeldingen.** Voortbouwend op de inzichten van modelgedreven polaire transformaties, stellen we een modelgedreven techniek om bij te snijden voor om de invoerafmetingen van neurale netwerken te minimaliseren zonder fijne details op te offeren. Door het doelobject te lokaliseren in een gedownsampled beeld en geïdentificeerde interessegebieden te extraheren uit een beeld met hoge resolutie, behouden we nauwkeurige segmentatie met kleinere inputs voor het netwerk. Aangezien de complexiteit van het model toeneemt met de grootte van de afbeelding, verbetert deze reductie van de inputgrootte de gegevensefficiëntie.

3. **Een nieuwe neurale netwerkarchitectuur voor beeldsegmentatie met hoge resolutie die objectdetectie in beelden met lage resolutie en segmentatie in beelden met hoge resolutie combineert.** We breiden onze voorbewerkingsmethoden uit om een end-to-end trainbaar netwerk te creëren dat objectlokalisatie in lage resolutie combineert met beeldsegmentatie in hoge resolutie. We maken netwerkconvergentie op twee belangrijke manieren mogelijk. Ten eerste gebruiken we een gedeelde architectuur tussen de lokalisatie- en segmentatie subnetwerken, wat transferleren mogelijk maakt. Ten tweede zorgen we ervoor dat de gradiënt door het netwerk stroomt door informatie van het ene subnetwerk door te geven aan het andere. We laten zien dat het trainen van dit end-to-end netwerk de robuustheid van zowel de lokalisatie- als de segmentatiefase verhoogt.

4. **Een nieuwe methode om diepte-informatie te incorporeren in de data aan de ingang van tweedimensionale convolutionele neurale netwerken.** Met het oog op

de beperkingen van de gegevensefficiëntie van 3D-netwerken voor volumetrische gegevens zoals CT-scans, ontwikkelen we een methode om diepte-informatie te incorporeren in 2D-doorsneden door een genormaliseerde z-coördinaat toe te voegen aan elke doorsnede. We demonstreren dat dit effectieve segmentatie van CT-beelden met 2D netwerken op basis van doorsneden mogelijk maakt.

De effectiviteit van deze methoden wordt gevalideerd voor verschillende modaliteiten in medische beeldvorming, waaronder CT-scans, microscopie, dermatoscopie en colonoscopie, waarbij niet alleen een verbeterde gegevensefficiëntie wordt aangetoond, maar ook state-of-the-art segmentatieresultaten voor bepaalde taken. De geïntroduceerde technieken zijn veelzijdig, geschikt voor een breed spectrum van medische beeldvorming en kunnen dienen als algemene voorbewerkingsstappen voor elke op convolutionele neurale netwerken gebaseerde segmentatiearchitectuur.

Het onderzoek gepresenteerd in dit proefschrift resulteerde in de publicatie van vier onderzoekspapers in wetenschappelijke tijdschriften (allemaal als eerste auteur) en vijf papers die op internationale wetenschappelijke conferenties werden gepresenteerd (waarvan vier als eerste auteur).

Sažetak

Segmentacija medicinske slike igra ključnu ulogu u raznim medicinskim primjenama od kirurškog planiranja preko dijagnostike do istraživanja. Taj se proces često oslanja na neuronske mreže za točnu segmentaciju raznih modaliteta slike kao što su CT, MRI, mikroskopija, dermatoskopija i druge. Međutim, uspješnost ovih modela je značajno uvjetovano dostupnosti i kvalitetom podataka za trening, koje je često teško prikupiti zbog visokih vremenskih i financijskih zahtjeva dobivanja slika, invazivnosti nekih procedura medicinskog slikanja, velikih datoteka i regulatornih zahtjeva vezanih uz medicinske slike.

Količina truda i stručnosti koju zahtjeva označavanje medicinskih slika dodatno pogoršava navedene izazove, što čini prikupljanje velikih visokokvalitetnih skupova podataka za treniranje neuronskih mreža posebnom zahtjevnim. Zbog ovog nedostatka podataka potrebno je razviti podatkovno učinkovite modele segmentacije koji mogu dati pouzdane rezultate s ograničenim brojem uzoraka.

Iz područja statistike i teorije statističkog učenja, postavljena je teorija da, kako zadaci segmentacije medicinskih slika postaju kompleksniji, oni zahtijevaju neuronske mreže sa sve više parametara za uspješnu segmentaciju. Taj zahtjev za povećanjem parametara tada zahtjeva više uzoraka kako bi se izbjegla prenaučenost modela. Međutim, u medicinskim slikama nema dovoljno podatkovnih skupova s velikim brojem uzoraka. U ovom doktorskom radu je predstavljen pregled metoda segmentacije medicinskih slika i postojeće strategije poboljšanja podatkovne učinkovitosti tih metoda.

Predstavljene su razne nove metode koje pojednostavljuju zadatak segmentacije, što omogućuje treniranje neuronskih mreža s manjim brojem parametara te time manje potrebnih uzoraka. Metode se zasnivaju na transformiranju segmentacijske granice u oblik koji se može modelirati manjim brojem parametara. To se postiže korištenjem znanja iz domene i tradicionalnim tehnikama obrade slike kako bi se identificirale korisne transformacije slike. Razvijene su neuronske mreže koje dinamički određuju parametre tih transformacija za svaku sliku. Time se zadatak segmentacije razlomi na dva lakša zadatka: inicijalna gruba lokalizacija ciljanog objekta koja je popraćena segmentacijom pojednostavljene reprezentacije slike.

Uspješnost predstavljenih metoda procijenjena je u raznim modalitetima medicinske slike, uključujući CT, mikroskopiju, dermatoskopiju i kolonoskopiju. Pokazana su poboljšanja ne samo u podatkovnom učinkovitosti nego su i za neke segmentacijske zadatke postignuti rezultati bolji od onih objavljenih u dosadašnjoj literaturi. Predstavljene metode su fleksibilne i koriste za širok spektar tipova medicinskih slika. Mogu se koristiti kao opći koraci pretprocesiranja slika za bilo koju arhitekturu segmentacijskih konvolucijskih neuronskih mreža.

Konkretnije, koriste se dvije neuronske mreže za učinkovito rješavanje problema segmentacije. Prva se mreža fokusira na grubo lokaliziranje ciljanog objekta koristeći kao oznake Gaussovu distribuciju centriranu na objektu ili standardne segmentacijske mape. Zatim se konstruira funkcija koja za danu grubu lokalizaciju daje parametre transformacije slike, a čija je svrha pojednostavljivanje zadatka segmentacije. Proces odlučivanja o korištenoj funkciji i transformaciji slike zasniva se na znanju iz domene i empirijski utvrđenim rezultatima. Primjerice, kako bi se učinkovitije segmentirali eliptični

objekti koristi se polarna transformacija slike, i to tako da je polarno ishodište pozicionirano u središtu objekta. Ulazna slika prolazi kroz transformaciju slike prema dobivenim parametrima kao priprema za drugu neuronsku mrežu koja obavlja detaljnu segmentaciju. Ta je druga mreža posebno trenirana na transformiranim slikama koristeći parametre koji su dobiveni od ručno označenih segmentacijskih maski.

Treniranjem druge mreže na transformiranim slikama pojednostavljuje se granica segmentacije, te se time mogu koristiti mreže s manje parametara za zadatak segmentacije. Mreže s manje parametara trebaju manje podataka za treniranje, tako da ovaj pristup prirodno dovodi do povećanja učinkovitosti podataka. Rezultati pokazuju usporedivu ili poboljšanu točnost korištenjem manjeg uzorka ili manjeg broja oznaka u različitim aplikacijama medicinskih slika, kao što su segmentacija jetre i epikardijalnog masnog tkiva iz CT skeniranja, kožnih lezija, polipa iz kolonoskopskih slika i stanica iz mikroskopskih slika.

Spomenute dvije mreže mogu biti zasebne neuronske mreže ili se mogu povezati i zajednički trenirati. Značajna je prednost ovog pristupa to što mreže za grubu lokalizaciju i finu segmentaciju mogu dijeliti istu često korištenu arhitekturu za segmentaciju medicinskih slika. To omogućuje jednostavan prijenos znanja između dviju mreža, ubrzavajući vrijeme obuke i poboljšavajući točnost. Osim toga, mreže se mogu unaprijed trenirati uz pomoć postojećih dostupnih skupova podataka ili neuronskih mreža za segmentaciju medicinskih slika.

U svrhu poboljšanja podatkovne učinkovitosti segmentacije medicinskih slika, ovaj doktorski rad predstavlja sljedeće izvorne znanstvene doprinose:

1. **Nova metoda segmentacije biomedicinskih slika temeljena na pretprocesiranju polarnom transformacijom s naučenim ishodištem.** Predstavljena je nova tehnika pretprocesiranja medicinskih slika, posebice onih koje sadrže objekte eliptičnih oblika. Primjenom polarne transformacije kružne se granice odluke pojednostavljuju u linearne, što čini segmentaciju jednostavnijom. Razvijena je neuronska mreža da identifikaciju optimalnog ishodišta polarne transformacije. Poboljšana je kvaliteta segmentacije i omogućeno korištenje manje kompleksnih neuronskih mreža.

2. **Poboljšanje metode za smanjenje veličine ulaznih slika u neuronske mreže obrezivanjem i korištenjem značajnih dijelova slike.** Motivirano uvidima korištenjem polarne transformacije, predložena je tehnika obrezivanja slika navođena modelom u svrhu smanjenja veličine ulaznih slika u neuronsku mrežu bez gubitka detalja. Precizna je segmentacija zadržana usprkos smanjenju veličine ulazne slike time što se ciljani objekt lokalizira u slici niske rezolucije te se obrezuje regija od interesa iz slike visoke rezolucije. Pokazano je da ovaj pristup povećava podatkovnu učinkovitost jer kompleksnost modela raste s veličinom ulazne slike.

3. **Nova arhitektura neuronske mreže za segmentaciju slika visoke rezolucije koja kombinira detekciju objekata na slikama niske rezolucije i segmentaciju objekata na slikama visoke rezolucije.** Predstavljene metode pretprocesiranja slika su dodatno unaprijeđene tako što se razvila cjelovita neuronska mreža koja se može u cijelosti trenirati koja kombinira detekciju objekta na slici niske rezolucije i segmentira objekt na obrezanom dijelu slike visoke rezolucije. Mreži je omogućena konvergencija na dva načina. Prvo, koristi se ista arhitektura mreže za grubu i detaljnu segmentaciju, što omogućuje prijenosno učenje. Drugo, informacije se prosljeđuju od grube do detaljne segmentacijske podmreže, što osigurava tok gradijenta kroz cijelu mrežu. Pokazano je da treniranje ove cjelovite mreže povećava robusnost i grubog i detaljnog stadija segmentacije.

4. **Nova metoda ugradnje dubinskih informacija u ulazne podatke dvodimenzionalne konvolucijske neuronske mreže.** S obzirom na ograničenja u podatkovnoj učinkovitosti 3D neuronskih mrežama za segmentaciju volumetričnih slika kao što je CT, razvijena je metoda ugradnje dubinskih informacija u 2D presjeke slike proširenjem presjeka dodatnim kanalom koji predstavlja normaliziranu z koordinatu. Pokazano je da ovo omogućuje uspješnu segmentaciju CT slika s 2D mrežama temeljenih na presjecima.

Svaka se od predstavljenih metoda može koristiti kao opći korak pretprocesiranja neovisno o korištenoj arhitekturi konvolucijske neuronske mreže. Ovi doprinosi prikazuju mogućnosti općenitog pojednostavljenja zadataka segmentacije medicinskih slika i poboljšanja podatkovne učinkovitosti s raznim modalitetima i zadacima medicinskih slika.

Kao rezultat istraživanja predstavljenog u ovom doktorskom radu objavljeno je četiri rada u međunarodnim znanstvenim časopisima (svi kao prvi autor) i pet radova na međunarodnim znanstvenim skupovima (četiri od kojih kao prvi autor).

Contents

Acknowledgements **V**

Abstract **VII**

Samenvatting **XI**

Sažetak **XV**

1 Introduction **1**
1.1 Contributions 2
1.2 List of Publications 3
1.2.1 Publications in Scientific Journals 3
1.2.2 Publications in Scientific Conferences 4
1.3 Organization of the Thesis 4
References 6

2 Neural Network-Based Segmentation of Biomedical Images **9**
2.1 Common Types of Biomedical Images 9
2.1.1 3D Modalities 9
2.1.2 2D Modalities 13
2.2 Image Segmentation: From Images to Segmentation Maps 14
2.2.1 Traditional Image Processing Methods 15
2.2.2 Machine Learning 18
2.3 Deep Learning-Based Segmentation Methods 19
2.3.1 Neural Network Training, Validation and Testing 20
2.3.2 Encoders and Decoders 21
2.3.3 Convolutional Neural Networks 22
2.4 CNN Architectures for Medical Image Segmentation 26
2.4.1 Fully Convolutional Network (FCN) 26
2.4.2 U-Net and Its Variants 27
2.4.3 Mask R-CNN 30
2.4.4 Other Notable Segmentation CNNs 31
2.5 Fully Connected Transformers for Medical Image Segmentation 32
2.5.1 Attention 32
2.5.2 Positional Encoding 33
2.5.3 Adapting Transformers to Image Segmentation 34
References 36

3 Data Efficiency in Neural Network-Based Image Segmentation **41**
3.1 Transfer Learning 44
3.1.1 Simple Transfer Learning 44
3.1.2 Domain Adaptation 45
3.1.3 Semi-Supervised and Self-Supervised Learning 46

3.2 Synthetic Data . . . 47
3.3 Regularization . . . 47
3.4 Conclusion . . . 48
References . . . 49

4 Data Efficiency via Model-Driven Preprocessing 51
4.1 Motivation: Using The Polar Transform for Preprocessing . . . 51
4.2 Model-driven Preprocessing . . . 53
4.2.1 Training Model-Driven Preprocessing Networks . . . 56
4.2.2 Related Work . . . 57
4.3 Model-Driven Polar Transform Using Centerpoint Prediction . . . 57
4.3.1 Methodology . . . 58
4.3.2 Centerpoint prediction . . . 59
4.3.3 Experiments . . . 63
4.3.4 Results and Discussion . . . 65
4.4 Supporting Multiple Transformations of an Image . . . 71
4.4.1 Methodology . . . 73
4.4.2 Results and Discussion . . . 75
4.5 Conclusion . . . 77
References . . . 79

5 Reducing Model Input Sizes with Model-Driven Crops 83
5.1 Model-Driven Image Cropping . . . 84
5.1.1 Related work . . . 87
5.1.2 Results . . . 89
5.2 An End-to-End Extension of Model-Driven Image Cropping . . . 97
5.2.1 End-to-end Network Architecture and Training . . . 97
5.2.2 Experiments in Clinical Dermatological Image Segmentation . . . 98
5.3 Conclusion . . . 103
References . . . 104

6 Adding Depth Information to 2D U-Nets for CT Segmentation 107
6.1 Epicardial Adipose Tissue Segmentation . . . 108
6.2 Related Work . . . 109
6.3 Dataset Description . . . 109
6.4 Methodology . . . 110
6.4.1 Data Preprocessing . . . 111
6.4.2 Model Training . . . 111
6.5 Experiments and Results . . . 112
6.6 Conclusion . . . 113
References . . . 115

7 Conclusion 119

8 Curriculum Vitae 121

1 Introduction

Medical image segmentation, the process of delineating one region of the image such as cancerous tissues from the rest of the image, is a crucial step in computer-assisted medical image analysis. Whether the ultimate goal is surgical planning [Sel+02], diagnosis [Dev+21], performing measurements [Sob+19] or doing population-level research [Bas+10], segmentation is often the first step in understanding 2D and 3D medical images.

Neural networks have become the standard tool to achieve biomedical image segmentation in almost all problem areas including, among others, segmenting organs or specific tissues from CT, MRI, or X-ray images [Ant+22]; cells from microscopic images [Edl+21]; and skin lesions from dermatoscopic images [Cod+19].

The results of these methods are highly dependent on the quantity and quality of the training data [Cho+16]. However, obtaining medical imaging data is challenging due to several reasons. Firstly, capturing medical images is costly both in terms of time and finances. For instance, MRI and CT scanning can take 30 minutes or more and require equipment that is inaccessible to large parts of the world. Secondly, such data is often large in terms of file size and stored in complex systems, increasing the friction of sharing and using the data. Finally, some jurisdictions define medical images as personally identifiable data [Lot+20] and thus require explicit consent for their secondary use to train neural networks. While valid and understandable, these patient privacy concerns can limit the use of already existing large databases in medical institutions.

After obtaining a medical image, the image needs to be labeled with high-quality delineations of the target region. Such labeling is often done through a tedious and time-consuming process where multiple experts manually draw curves or polygons on the image. While some labeling methods make this process easier [Lut+19], each image still needs to be checked by an expert in the field. For challenging problems, this often requires highly trained and experienced specialists.

These challenges make collecting large medical image segmentation datasets infeasible. Therefore, to improve performance and robustness, there is a need to develop data-efficient segmentation methods. Data efficiency sometimes referred to as sample efficiency, measures how well a model performs with respect to the amount of training data. Data-efficient models achieve good results given a small amount of data. Here, the amount of data is determined by both sample size and the number of features inside each sample.

This thesis will demonstrate that as the complexity of segmentation tasks increases, neural networks require a greater number of parameters to achieve effective segmentation. However, the need for more parameters necessitates a larger sample size of training data to develop an accurate model. This presents a significant challenge in medical image segmentation, where there is a need for high-parameter models but insufficient available training data. To address this issue, we introduce various methods designed to mitigate the gap between the need for complex models and the scarcity of training data.

The central idea behind the methods presented in this thesis is simplifying the segmentation problem. By reducing the complexity, neural networks can operate with fewer

parameters, consequently lowering the demand for large sample sizes. This simplification is achieved by applying domain knowledge and established image processing techniques to identify image transformations that simplify the segmentation boundary into one that can be modeled using fewer parameters. Neural networks are then employed to dynamically determine the most effective transformation parameters for each image, thereby dividing the complex task of segmentation into two simpler steps: initial rough localization of the object and subsequent segmentation of a simplified image representation based on this localization.

We evaluate our methods on a large variety of medical imaging modalities including CT, microscopy, dermatoscopy, and colonoscopy. The results demonstrate improvements in data efficiency across a wide array of medical image segmentation tasks while achieving state-of-the-art results for some of them. All of the methods presented here are generally applicable across different medical imaging domains and can be used as preprocessing steps independent of the underlying convolutional neural network architecture.

1.1 Contributions

This thesis aims to develop new methods of achieving data efficiency in medical image segmentation. In particular, we propose the following original contributions to the scientific literature:

A new biomedical image segmentation method based on polar transform preprocessing with a learned center point.

We propose a method for preprocessing medical images where the objects of interest are elliptically shaped, as is often the case for organ or skin lesion segmentation. Motivated by the intuition that a polar transform transforms circular decision boundaries into linear ones, we show that this can be used to simplify the decision boundary of segmentation models. As a beneficial transformation depends on the selected origin, we construct a neural network to predict the center points of the objects that need to be segmented. Following this preprocessing, we train a segmentation network on polar-transformed images and observe improvements in segmentation performance, allowing for the use of networks with reduced parameters or smaller datasets compared to existing studies. This contribution is explained in more detail in Chapter 4 and has been preliminarily published in a journal article [Ben+21], with additional extensions of the method presented in a conference paper [Ben+22b].

An improved method of reducing the input image size in neural networks using salient image crops.

Knowing that reducing network input sizes generally benefits data efficiency and training times and building on the insights from our polar transform approach, we propose model-driven image cropping as a strategy to decrease neural network input sizes. We use a neural network with a regular segmentation architecture that roughly localizes a target object within a downsampled image. The rough localization is then used to extract high-resolution crops of the target objects in the original image resolution. These cropped images are then segmented using a second specially trained network, enabling precise delineation while working with significantly reduced network input sizes compared to the original image dimensions. This method is presented in Chapter 5 and published in a journal paper [Ben+23].

A new neural network architecture for high-resolution image segmentation that combines object detection in low-resolution images and segmentation in high-resolution images.

We expand upon our model-driven preprocessing techniques to make it end-to-end trainable. We do so by connecting a subnetwork that performs learned cropping to a segmentation subnetwork, connected via a cropping layer. The shared architecture between the two subnetworks allows us to use transfer learning to help the subnetworks converge. Additionally, pass information between the subnetworks to ensure gradient flow throughout the entire network. This integrated approach allows for the fine-tuning of the network as a whole, enhancing the robustness of each subnetwork to the potential inaccuracies of the other. We present this contribution in Chapter 5. This contribution was published in a conference paper [BHG24].

A new method of embedding depth information in two-dimensional convolutional neural network input data.

In the thesis, we show that using larger images necessitates higher-capacity models, which negatively impacts data efficiency. Particularly for volumetric data like CT scans, training 2D neural networks often proves more data-efficient than their 3D counterparts. Addressing this, we introduce a straightforward method to incorporate depth information into 2D slices of CT images for use in fully convolutional neural networks. Assuming each slice is located at some z-axis value in the scan, we append the normalized z-coordinate (0 being the bottom and 1 at the top of each scan cropped to the same region) as an additional channel to each image slice, enhancing the network's ability to interpret depth, which, as we demonstrate, leads to improved segmentation outcomes. Details of this method are discussed in Chapter 6 and have been published in a conference paper [BHG21].

1.2 List of Publications

1.2.1 Publications in Scientific Journals

- [Ben+21] Marin Benčević, Irena Galić, Marija Habijan, and Danilo Babin. "Training on Polar Image Transformations Improves Biomedical Image Segmentation". In: *IEEE Access* 9 (2021), pp. 133365–133375. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3116265

- [Ben+22a] Marin Benčević, Irena Galić, Marija Habijan, and Aleksandra Pižurica. "Recent Progress in Epicardial and Pericardial Adipose Tissue Segmentation and Quantification Based on Deep Learning: A Systematic Review". In: *Applied Sciences* 12.10 (May 2022), p. 5217. ISSN: 2076-3417. DOI: 10.3390/app12105217

- [Ben+23] Marin Benčević, Yuming Qiu, Irena Galić, and Aleksandra Pižurica. "Segment-Then-Segment: Context-Preserving Crop-Based Segmentation for Large Biomedical Images". In: *Sensors* 23.2 (Jan. 2023), p. 633. ISSN: 1424-8220. DOI: 10.3390/s23020633

- [Ben+24] Marin Benčević, Marija Habijan, Irena Galić, Danilo Babin, and Aleksandra Pižurica. "Understanding Skin Color Bias in Deep Learning-Based Skin Lesion Segmentation". In: *Computer Methods and Programs in Biomedicine* 245 (Mar. 2024), p. 108044. ISSN: 01692607. DOI: 10.1016/j.cmpb.2024.108044

1.2.2 Publications in Scientific Conferences

- [BHG21] Marin Benčević, Marija Habijan, and Irena Galić. "Epicardial Adipose Tissue Segmentation from CT Images with A Semi-3D Neural Network". In: *2021 International Symposium ELMAR*. Zadar, Croatia: IEEE, Sept. 2021, pp. 87–90. ISBN: 978-1-66544-437-8. DOI: 10.1109/ELMAR52657.2021.9550936

- [Ben+22b] Marin Benčević, Marija Habijan, Irena Galić, and Danilo Babin. "Using the Polar Transform for Efficient Deep Learning-Based Aorta Segmentation in CTA Images". In: *2022 International Symposium ELMAR*. Sept. 2022, pp. 191–194. DOI: 10.1109/ELMAR55880.2022.9899786

- [Ben+22c] Marin Benčević, Marija Habijan, Irena Galić, and Aleksandra Pižurica. "Self-Supervised Learning as a Means to Reduce the Need for Labeled Data in Medical Image Analysis". In: *2022 30th European Signal Processing Conference (EUSIPCO)*. Belgrade, Serbia: IEEE, Aug. 2022, pp. 1328–1332. ISBN: 978-90-827970-9-1. DOI: 10.23919/EUSIPCO55093.2022.9909542

- [Ver+23] Nicolas Vercheval, Marin Benčević, Dario Mužević, Irena Galić, and Aleksandra Pižurica. "Counterfactual Functional Connectomes for Neurological Classifier Selection". In: *2023 31st European Signal Processing Conference (EUSIPCO)*. Helsinki, Finland: IEEE, Sept. 2023, pp. 1050–1054. ISBN: 978-94-6459-360-0. DOI: 10.23919/EUSIPCO58844.2023.10289859

- [BHG24] Marin Benčević, Marija Habijan, and Irena Galić. "Crop-Guided Neural Network Segmentation of High-Resolution Skin Lesion Images". In: *MOSTART2024: International Conference on Digital Transformation in Education and Artificial Intelligence Applications*. Mostar, Bosnia and Herzegovina, Apr. 2024

1.3 Organization of the Thesis

Aside from this introductory chapter, the thesis is organized into two background and theory chapters followed by three chapters that present the main novel contributions of this thesis, followed by a conclusion at the end.

Chapter 2 offers a detailed look at various medical imaging modalities and their technical details for neural network training. It lays the groundwork for image segmentation, with a focus on segmentation convolutional neural networks. The chapter concludes with a review of medical image segmentation techniques, from traditional image processing techniques to recent neural network-based methods.

Chapter 3 delves into the principles underpinning data efficiency from the fields of statistics and learning theory, establishing core dependencies between model complexity, sample size, and segmentation error. These principles guide the methods presented in the remainder of the thesis. This chapter also provides a synopsis of different strategies for improving data efficiency in neural network-based image segmentation.

In Chapter 4, we show a general overview of the model-driven preprocessing methods described in this thesis. It then shows a specific implementation using the polar transform as the main transformation used to improve data efficiency and shows empirical studies of the method across various medical image modalities. These methods were published in [Ben+21] and [Ben+22b].

Following this, Chapter 5 expands on the model-driven preprocessing concept, focusing on a method to reduce the input size for neural networks. We

achieve this by predicting salient image crops that are used to crop target objects from a high-resolution image. The chapter describes results published in [**bencevicSegmentthenSegmentContextPreservingCropBased2023**] and [BHG24].

Chapter 6 presents a small detour from the model-driven preprocessing approach as it is a standalone method of preprocessing 3D images to be more easily segmented by 2D neural networks. This is achieved by adding the CT slice depth as a separate channel of a 2D slice image, as published in [BHG21].

The thesis concludes with the Conclusion, summarizing key findings and contributions made throughout the study and offering an overview of the research outcomes in a wider context of medical image segmentation.

References

[Ant+22] Michela Antonelli, Annika Reinke, Spyridon Bakas, Keyvan Farahani, Annette Kopp-Schneider, et al. "The Medical Segmentation Decathlon". In: *Nature Communications* 13.1 (July 2022), p. 4128. ISSN: 2041-1723. DOI: 10.1038/s41467-022-30695-9. (Visited on 09/28/2023).

[Bas+10] Gorka Bastarrika, Jordi Broncano, U. Joseph Schoepf, Florian Schwarz, Yeong Shyan Lee, et al. "Relationship Between Coronary Artery Disease and Epicardial Adipose Tissue Quantification at Cardiac CT". In: *Academic Radiology* 17.6 (June 2010), pp. 727–734. ISSN: 10766332. DOI: 10.1016/j.acra.2010.01.015. (Visited on 09/27/2023).

[Ben+21] Marin Benčević, Irena Galić, Marija Habijan, and Danilo Babin. "Training on Polar Image Transformations Improves Biomedical Image Segmentation". In: *IEEE Access* 9 (2021), pp. 133365–133375. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3116265.

[Ben+22a] Marin Benčević, Irena Galić, Marija Habijan, and Aleksandra Pižurica. "Recent Progress in Epicardial and Pericardial Adipose Tissue Segmentation and Quantification Based on Deep Learning: A Systematic Review". In: *Applied Sciences* 12.10 (May 2022), p. 5217. ISSN: 2076-3417. DOI: 10.3390/app12105217.

[Ben+22b] Marin Benčević, Marija Habijan, Irena Galić, and Danilo Babin. "Using the Polar Transform for Efficient Deep Learning-Based Aorta Segmentation in CTA Images". In: *2022 International Symposium ELMAR*. Sept. 2022, pp. 191–194. DOI: 10.1109/ELMAR55880.2022.9899786.

[Ben+22c] Marin Benčević, Marija Habijan, Irena Galić, and Aleksandra Pižurica. "Self-Supervised Learning as a Means to Reduce the Need for Labeled Data in Medical Image Analysis". In: *2022 30th European Signal Processing Conference (EUSIPCO)*. Belgrade, Serbia: IEEE, Aug. 2022, pp. 1328–1332. ISBN: 978-90-827970-9-1. DOI: 10.23919/EUSIPCO55093.2022.9909542.

[Ben+23] Marin Benčević, Yuming Qiu, Irena Galić, and Aleksandra Pižurica. "Segment-Then-Segment: Context-Preserving Crop-Based Segmentation for Large Biomedical Images". In: *Sensors* 23.2 (Jan. 2023), p. 633. ISSN: 1424-8220. DOI: 10.3390/s23020633.

[Ben+24] Marin Benčević, Marija Habijan, Irena Galić, Danilo Babin, and Aleksandra Pižurica. "Understanding Skin Color Bias in Deep Learning-Based Skin Lesion Segmentation". In: *Computer Methods and Programs in Biomedicine* 245 (Mar. 2024), p. 108044. ISSN: 01692607. DOI: 10.1016/j.cmpb.2024.108044.

[BHG21] Marin Benčević, Marija Habijan, and Irena Galić. "Epicardial Adipose Tissue Segmentation from CT Images with A Semi-3D Neural Network". In: *2021 International Symposium ELMAR*. Zadar, Croatia: IEEE, Sept. 2021, pp. 87–90. ISBN: 978-1-66544-437-8. DOI: 10.1109/ELMAR52657.2021.9550936.

[BHG24] Marin Benčević, Marija Habijan, and Irena Galić. "Crop-Guided Neural Network Segmentation of High-Resolution Skin Lesion Images". In: *MOSTART2024: International Conference on Digital Transformation in Education and Artificial Intelligence Applications*. Mostar, Bosnia and Herzegovina, Apr. 2024.

[Cho+16] Junghwan Cho, Kyewook Lee, Ellie Shin, Garry Choy, and Synho Do. *How Much Data Is Needed to Train a Medical Image Deep Learning System to Achieve Necessary High Accuracy?* Jan. 2016. arXiv: 1511.06348 [cs]. (Visited on 09/28/2023).

[Cod+19] Noel Codella, Veronica Rotemberg, Philipp Tschandl, M. Emre Celebi, Stephen Dusza, et al. *Skin Lesion Analysis Toward Melanoma Detection 2018: A Challenge Hosted by the International Skin Imaging Collaboration (ISIC)*. Mar. 2019. arXiv: 1902.03368 [cs]. (Visited on 09/28/2023).

[Dev+21] Sindhu Devunooru, Abeer Alsadoon, P. W. C. Chandana, and Azam Beg. "Deep Learning Neural Networks for Medical Image Segmentation of Brain Tumours for Diagnosis: A Recent Review and Taxonomy". In: *Journal of Ambient Intelligence and Humanized Computing* 12.1 (Jan. 2021), pp. 455–483. ISSN: 1868-5137, 1868-5145. DOI: 10.1007/s12652-020-01998-w. (Visited on 09/27/2023).

[Edl+21] Christoffer Edlund, Timothy R. Jackson, Nabeel Khalid, Nicola Bevan, Timothy Dale, et al. "LIVECell—A Large-Scale Dataset for Label-Free Live Cell Segmentation". In: *Nature Methods* 18.9 (Sept. 2021), pp. 1038–1045. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-021-01249-6. (Visited on 09/28/2023).

[Lot+20] Eyal Lotan, Charlotte Tschider, Daniel K. Sodickson, Arthur L. Caplan, Mary Bruno, et al. "Medical Imaging and Privacy in the Era of Artificial Intelligence: Myth, Fallacy, and the Future". In: *Journal of the American College of Radiology* 17.9 (Sept. 2020), pp. 1159–1162. ISSN: 15461440. DOI: 10.1016/j.jacr.2020.04.007. (Visited on 09/28/2023).

[Lut+19] Brendon Lutnick, Brandon Ginley, Darshana Govind, Sean D. McGarry, Peter S. LaViolette, et al. "An Integrated Iterative Annotation Technique for Easing Neural Network Training in Medical Image Analysis". In: *Nature Machine Intelligence* 1.2 (Feb. 2019), pp. 112–119. ISSN: 2522-5839. DOI: 10.1038/s42256-019-0018-3. (Visited on 09/28/2023).

[Sel+02] D. Selle, B. Preim, A. Schenk, and H.-O. Peitgen. "Analysis of Vasculature for Liver Surgical Planning". In: *IEEE Transactions on Medical Imaging* 21.11 (Nov. 2002), pp. 1344–1357. ISSN: 0278-0062. DOI: 10.1109/TMI.2002.801166. (Visited on 09/27/2023).

[Sob+19] Zahra Sobhaninia, Shima Rafiei, Ali Emami, Nader Karimi, Kayvan Najarian, et al. "Fetal Ultrasound Image Segmentation for Measuring Biometric Parameters Using Multi-Task Deep Learning". In: *2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*. Berlin, Germany: IEEE, July 2019, pp. 6545–6548. ISBN: 978-1-5386-1311-5. DOI: 10.1109/EMBC.2019.8856981. (Visited on 09/27/2023).

[Ver+23] Nicolas Vercheval, Marin Benčević, Dario Mužević, Irena Galić, and Aleksandra Pižurica. "Counterfactual Functional Connectomes for Neurological Classifier Selection". In: *2023 31st European Signal Processing Conference (EUSIPCO)*. Helsinki, Finland: IEEE, Sept. 2023, pp. 1050–1054. ISBN: 978-94-6459-360-0. DOI: 10.23919/EUSIPCO58844.2023.10289859.

2 Neural Network-Based Segmentation of Biomedical Images

Image segmentation is the process of defining the borders of regions or objects on an image. For instance, delineating the liver from its surrounding tissue in a CT scan, separating each of a cell's organelles in a microscopic image, or determining the border between healthy and unhealthy tissue in a dermatoscopic image. This chapter introduces the different types of biomedical images as well as the process of segmenting them. We will first provide an overview of the world of biomedical images and its diverse modalities and imaging techniques. We will touch on the technical details of how these images are stored and used in segmentation algorithms. Then, we will more formally describe the process of image segmentation from traditional segmentation methods to the most current deep learning methods.

Biomedical imaging covers a wide range of techniques used in both biology and medicine [Wal+19]. This includes complex modalities like CT scans and simpler ones like photographs of a patient's skin or plants. Because of this variety, a method designed for one modality might not work for another without changes. So, understanding each modality's specifics is crucial when considering segmentation methods. The next section will provide an overview of different biomedical imaging modalities, focusing on those most relevant to this thesis.

2.1 Common Types of Biomedical Images

Broadly, we may classify biomedical images into ones that have three and two spatial dimensions. Images with three spatial dimensions include modalities such as CT and MRI. Examples of image modalities with two spatial dimensions, among others, include X-ray imaging, most microscopic images, and dermatoscopic images. Images from both of these categories can also include additional dimensions such as time. It is worth noting that biomedical images do not always fit neatly into distinct categories. While our classification provides a general overview, there are emerging techniques that merge different modalities [Str+02], blurring the classification boundaries presented here.

2.1.1 3D Modalities

It is hard to overstate the importance of 3D modalities such as CT and MRI in contemporary medical practice, scientific advancements, and patient outcomes [Wal+19]. These techniques offer medical professionals a window into the patient's body, allowing much safer and more reliable diagnosis, treatment, and surgical planning.

The foundational principle behind these modalities revolves around emitting a signal capable of penetrating tissues. This signal interacts variably with different tissue types. After transmission through the body, the residual signal is detected and used to create an image. By repeating this process across various planes of the body and leveraging computational algorithms, a 3D volume of the patient's anatomy is constructed. From

there, these modalities are often stored as voxel-based 3D files, where a voxel is the smallest 3D unit equivalent to a pixel in 2D images. They are usually stored alongside patient data such as age, sex, imaging parameters, and other relevant information.

In the subsequent sections, we will delve deeper into the specifics of the two predominant 3D biomedical imaging modalities: CT and MRI.

Computed Tomography (CT)

Computed tomography is an X-ray-based imaging technique where different planes of the subject are captured and then reconstructed into a 3D image using a process called tomography [Gol07a]. CT, as opposed to MRI, uses ionizing radiation that can be harmful in large doses. As such, the benefit of each scan should be weighed against potential hazards. Notably, there is a relationship between radiation dose and image quality. The radiation dose has to be carefully balanced to achieve the required image quality without subjecting the patient to undue radiation [Gol07b].

This balance between radiation dose and image quality has implications for segmentation methods. Low-dose and high-dose CT images visually differ — enough to cause issues in the segmentation model's performance across these two domains [Li+23]. High-quality images result in better neural network models, but their availability is much more limited than low-dose scans.

Another way to enhance the quality of CT images is the use of contrast agents. Typically administered intravenously, these agents are designed to have a high intensity on the resulting CT image. A common example of this is CT angiography where contrast is used to make it easier to delineate blood vessels from surrounding tissue [Gol07a].

CT images are generally stored as $W \times H \times D$ matrices, with each element indicating the grayscale intensity of a voxel (*volumetric pixel*) within the 3D volume. In contrast with regular pixels, voxels can have different lengths along each dimension and usually are not cubes [Mot+21]. The voxel's depth (along the z-axis) is known as slice thickness, and it is chosen based on the task at hand. Thinner slices increase the spatial detail in the image making it easier to see minute details. However, thinner slices generally also increase the noise in the image. Slice thickness is selected to maintain a tradeoff between spatial resolution and level of noise. The width and height of the voxel are governed by the field of view and the scanner itself.

Voxel values in CT images are typically stored as 12-bit signed integers. These values represent the attenuation of the X-ray as it passes through various tissues. Denser structures such as bones attenuate the radiation more strongly than fat tissue and thus result in higher intensity values.

To maintain consistency across scans and machines, the attenuation values in a CT are linearly transformed [FO11]. Specifically, air is assigned a value of -1000, while water is attributed a value of 0, under standard temperature and pressure conditions. Mathematically, this transformation is expressed as

$$\mathrm{HU}(\mu) = 1000 \cdot \frac{\mu - \mu_{\text{water}}}{\mu_{\text{water}} - \mu_{\text{air}}}, \tag{2.1}$$

where μ is the attenuation value of a given voxel while μ_{water} and μ_{air} are attenuation values of water and air, respectively. This transformation is called the **Hounsfield unit** (HU) scale. Calibration according to this scale ensures that the attenuation of any tissue is always relative to water's attenuation. As a result, values remain standardized across different images, CT machines, imaging parameters, and institutions. The approximate Hounsfield unit values corresponding to various tissues are shown in Table 2.1.

Table 2.1 Approximate Hounsfield values of various tissues [FO11; KLG16].

Tissue	Hounsfield value
Fat	-30 to -70 HU
Muscle	20 to 40 HU
Bone	1000 HU
Blood	13 to 50 HU

The Hounsfield unit values play a pivotal role in the analysis of CT scans. Beyond facilitating the identification of specific tissues or structures, variations in Hounsfield values of the same tissues across patients or scans can hold significant diagnostic implications. For example, clotted blood exhibits a higher HU value than its unclotted counterpart, establishing the Hounsfield value as a potential marker for intracranial hemorrhage [KLG16]. In another context, the mean HU value of epicardial fat can be indicative of myocardial infarction [Mah+17].

The range of gray levels in a CT scan is far wider than what the human eye can discern. Therefore, it is standard practice for professionals to employ a technique called "windowing" when reviewing scans [BM17]. In essence, windowing compresses the value range by applying two specific thresholds (t_1, t_2), as defined by

$$y(x) = \begin{cases} t_1, & \text{if } x \leq t_1 \\ x, & \text{if } t_1 < x < t_2 \\ t_2, & \text{if } x \geq t_2 \end{cases}, \tag{2.2}$$

where x is a given HU value. This allows CT visualization software to visually stretch the remaining range of gray values, thus enhancing the differences in attenuation, as seen in Figure 2.1. This technique is not limited to human experts; it finds applications in image segmentation as well. It is common to window CT image inputs into segmentation neural networks. For example, if segmenting fat, it is often beneficial to discard all voxels outside of the fat tissue range [Ben+22], reducing the number of voxels that the network needs to process.

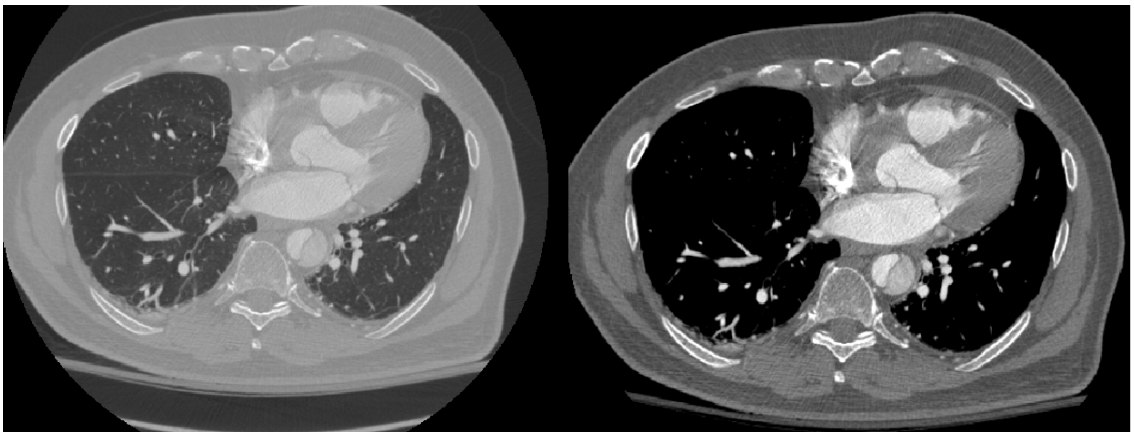


Figure 2.1: A cardiac CTA in its full range (left) and windowed (right). [Rad+22]

Magnetic Resonance Imaging (MRI)

Much like CT, MRI produces voxel-based images that visualize the insides of an object. The process of obtaining an MRI image can be separated into several steps [Gro+15]:

1. The machine applies a strong magnetic field to the subject. This causes protons in the body to align in the same direction — parallel to the z-axis.
2. Next, the machine emits a brief radio frequency pulse, perturbing a small portion of the aligned protons and causing them to deviate from their uniform alignment.
3. After the pulse is turned off, the protons gradually return back to alignment. As they realign, the movement of their positive charge induces an electric current in a coil inside the MRI machine. In contrast to CT, which gauges the attenuation of X-rays, MRI measures this induced current.

The time to reach the equilibrium state depends on the specific tissue type, so the measured current allows the differentiation of tissues in the image. The equilibrium state is achieved through two independent processes: T1 and T2. T1 measures the time it takes the protons to reach equilibrium longitudinal alignment, while T2 measures the time to regain its equilibrium transverse alignment. Different tissues return to their equilibrium states at varying rates for T1 and T2, enabling two different types of MRI images: T1- or T2-weighted. For example, water exhibits a long T1 time, while fat has a shorter one. This means that in T1-weighted images, fat appears brighter than water, while the opposite is true in T2-weighted images. This can be seen in Figure 3.1.

The relative differences in T1 and T2 images make each more suitable for imaging specific tissues [Boa+22]. For instance, T1-weighted images allow for identifying fatty tissue or detecting liver lesions, while T2-weighted images are more effective for identifying white matter lesions and edemas.

The voxels in an MRI, much like a CT, are generally rectangular solids and can be different in width, height, and depth. The image resolution is determined by several factors, including the scanner's field of view, the MRI machine itself, and specific imaging parameters. Generally, increasing resolution leads to higher levels of noise and a longer acquisition time [Mac96]. Consequently, an optimal balance between resolution,

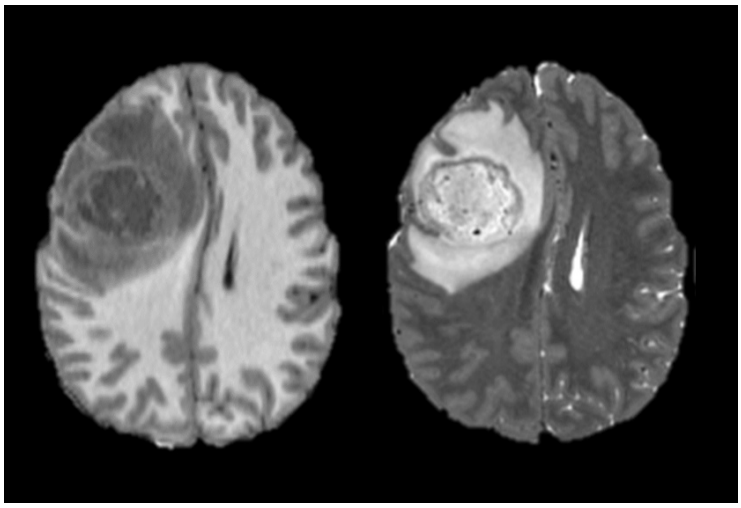


Figure 2.2: An example T1-weighted MRI (left) and a T2-weighted MRI (right) showing a diffuse glioma. [Cal+22]

noise, and scan time must be found, tailored to the tissue being imaged and the specific diagnostic objective.

MRI has several advantages over CT. For one, MRI usually offers superior contrast resolution (the minimal detectable difference between tissues) when imaging soft tissues [Cha+87]. This has established MRI as the gold standard for a large number of diagnostic procedures. Much like with CT scans, MRI contrast can be further enhanced using contrast agents. Furthermore, as MRI uses magnetic fields instead of ionizing radiation, it poses no radiation-induced risks to patients.

However, MRI has its own set of limitations. MRI scans typically take longer than CT scans, which can be uncomfortable for patients, particularly those who are claustrophobic or have mental and cognitive disabilities [TSD08]. MRI is contraindicated for individuals with certain non-removable magnetic objects, like coronary pacemakers and specific implants [SSC07]. A notable drawback is the lack of standardization in MRI values across different scans and machines [Wah+21]. Contrary to CT scans, an MRI voxel's intensity is not consistent across different scans and can only be reliably interpreted in relation to adjacent tissue in the same scan.

2.1.2 2D Modalities

3D modalities, while offering a detailed view of a subject, often entail time-consuming and infeasible procedures in common clinical use. 2D modalities such as X-ray and diagnostic ultrasound are often quicker and more readily available. This ease and speed of 2D modalities contribute to the large amount of publicly available datasets in 2D modalities compared to 3D ones. For example, analyzing X-ray images is one of the most active fields in computerized medical image analysis research [Ngu+20; Irv+19].

X-ray imaging

X-ray imaging, also known as radiography, can be thought of as the 2D equivalent of a CT as it also relies on ionizing radiation [Sei04]. To obtain an image, X-rays are emitted on one side of the subject and captured on the opposite side. The intensity of the pixels corresponds to the attenuation of the emitted radiation. Denser materials appear with a higher intensity on the resulting image due to their high attenuation.

When capturing an X-ray image, an expert manually positions the generator. The relative positioning of this generator and the subject directly influences the image's magnification and field of view. If the subject is closer to the detector than the generator, it will appear magnified on the resulting image. Such magnification makes the scale on an X-ray image non-standard and precludes the possibility of making objective length and area measurements.

The expert also determines various parameters that ultimately change the quality and quantity of the X-ray beams. The quality measures the ability of an X-ray to penetrate tissue and is proportional to the X-ray energy level. Quantity, on the other hand, measures the number of photons constituting the beam. A high beam quality is advantageous for imaging denser tissues, like bones, or traversing larger volumes. However, high-quality beams reduce the contrast in soft tissues. Owing to the variability introduced by these parameters, X-ray image intensities lack the uniform standardization found in CT scans. Consequently, there is no standardized unit for gauging X-ray intensity across images.

Dermatological images (clinical and dermatoscopic)

The application of deep learning models to dermatology, especially skin lesion analysis, has gained significant momentum. This is driven in part by organizations such as the International Skin Imaging Collaboration (ISIC) which curates large dermatological datasets [Rot+21]. These datasets consist of two primary types of dermatological images: dermatoscopic and clinical. Clinical images are regular photographs of skin lesions, while dermatoscopic images are captured with a device called a digital epiluminescence dermatoscope. A dermatoscope consists of a camera attached to a magnifying lens with a built-in light source. It allows capturing detailed and magnified images of a skin lesion while filtering out skin reflections.

The primary application of deep learning in dermatological images is for classification, e.g. predicting whether a lesion is benign or not or detecting the type of skin disease. However, by precisely outlining the boundary of a skin lesion, segmentation techniques can yield more consistent and objective descriptors of the lesion, aiding classification algorithms [Rot+21].

Microscopy

In biomedicine, one of the predominant uses of computer vision is in segmenting, analyzing, and quantifying microscopic images [Kha+18]. This covers a broad array of tasks in digital pathology, from identifying cancerous cells and segmenting nuclei to quantifying white blood cell counts.

Publicly available microscopic images of cells are abundant due to how frequently they are captured and that they do not contain any personally identifiable information. However, the large size of these images can present a challenge. Often spanning multiple megapixels, they exceed the capacity of current deep-learning models and machines. Consequently, to make them more manageable, these images are typically divided into smaller patches, downscaled, or processed using a coarse-to-fine approach [Jha+21].

As can be seen, both 2D and 3D biomedical images are exceedingly diverse in their appearance, use, technical details, and format. Yet, many segmentation techniques prove versatile enough to be applied across different modalities. In the next section, we will provide a general outline of how image segmentation works, highlighting commonly used methods.

2.2 Image Segmentation: From Images to Segmentation Maps

Image segmentation is the process of categorizing each pixel (or voxel) of an image into one of several predefined classes. Consider a 3-class segmentation scenario for CT images, with classes being liver, liver tumor, and background. Each of these classes is assigned a distinct numeric identifier, termed the **class label**. For illustration, the labels could be '0', '1', and '2' for the background, liver, and tumor respectively. The segmentation process outputs an image identical in dimensions to the original but with each pixel's value corresponding to the class label of that position in the original image. As an example, pixels corresponding to the liver will all have the value '1'. This resulting image is called a **segmentation map** since it, like a map, delineates important regions on the original image.

In medical image segmentation, it is common to extract only a single tissue type against the backdrop, a technique known as **binary segmentation**. Additionally, multi-class segmentation challenges can be decomposed into multiple binary segmentation tasks, with each class having its individual class-vs-background segmentation map.

Therefore, one can frame any segmentation problem as a set of binary segmentation problems.

Mathematically, given a set of K classes, and an input d-dimensional image of N channels $I(A)$, $I \in \mathbb{R}^N$, $A \in \mathbb{N}^d$ where A is the location of each voxel, the segmentation map $M : \mathbb{N}^d \to \mathbb{R}^K$ maps each pixel location to a vector of class probabilities:

$$M(A) = (\ \Pr(C_1 \,|\, I(A)),\ \Pr(C_2 \,|\, I(A)),\ \cdots,\ \Pr(C_K \,|\, I(A))\ ), \tag{2.3}$$

where $\Pr(C_i \mid I(A))$ is the probability that the voxel $I(A)$ contains an object of class C_i. Expressed this way, the segmentation map is a K-channel image of the same size as $I(A)$. Each channel of the image corresponds to a probability map of finding an object of a given class at a given voxel location.

In binary segmentation, where $K = 2$, the classes can be denoted as C_1 for the background and C_2 for the target object. Given this, for all voxel locations A, the relation $\Pr(C_2 \,|\, I(A)) = 1 - \Pr(C_1 \,|\, I(A))$ holds true. Therefore $M(A)$ simplifies to a scalar value $M(A) = \Pr(C_2 \,|\, I(A))$. This representation is very common in medical image segmentation problems such as segmenting organs, cell nuclei, and skin lesions, among others.

Often, the next step in the process is to binarize the segmentation map $M(A)$. Voxels corresponding to the target object are set to '1', while the background is marked as '0'. In such cases, $M(A)$ is frequently termed a **segmentation mask**. In computer vision, "mask" refers to a binary image $M_{01}(A) \in \{0, 1\}$ that hides (masks) regions in another image $I(A)$, resulting in a masked image $I_m(A) = I(A)M_{01}(A)$. Within the context of deep learning-based image segmentation, the terms "segmentation map" and "segmentation mask" are often used interchangeably.

Having laid the foundation for the mathematical framework of image segmentation, we can now transition to exploring specific segmentation techniques. The next section presents an overview of common segmentation methods, from traditional ones based on heuristics to complex model-based approaches prevalent today.

2.2.1 Traditional Image Processing Methods

Despite the popularity of deep learning, traditional techniques, rooted in fundamental image characteristics, still play a vital role in modern medical image segmentation. Traditional techniques are often used as methods for pre-processing and data augmentation, as well as refining deep learning model outputs. As we will show later in the dissertation, traditional methods can increase the robustness and data efficiency of deep learning-based segmentation models.

Rather than relying on data-intensive training phases, these techniques often operate deterministically, using explicit algorithms that manipulate image characteristics. They directly use image properties such as intensity and texture, combined with heuristic strategies that draw from empirical observations and domain knowledge. Heuristics are best-practice rules derived from previous samples and experiences. For instance, the longest component in the bone intensity range of an X-ray image is usually the femur. These methods provide a clear, interpretable pathway to segmentation.

However, traditional methods are usually developed with a very specific application in mind and are hard to translate to other tasks and domains without significant changes. They are also sensitive to parameter selection and properties of the image such as intensity level. Helpfully, since they do not use a learning component their limitations can be known ahead of time. Their validity can also be confirmed using fewer samples than is the case for learning-based methods.

Image Thresholding

Thresholding is a fundamental technique that isolates regions in an image based on a specified range of intensities. In essence, it removes or retains portions of the image where the intensity either falls outside or within a given threshold.

To illustrate the use of thresholding, consider an example of segmenting adipose tissue on a CT scan. As mentioned earlier in this chapter, voxel intensities on CT scans quantify a tissue's X-ray radiation attenuation as measured in Hounsfield units. Typically, fatty tissue lies between -250 HU and -30 HU. Thresholding the entire scan to this range segments fatty tissue from all other tissues on the scan with no need for additional complex models. For preprocessing, Hounsfield unit thresholding is an efficient way to discard irrelevant voxels, allowing the rest of the segmentation process to focus on fewer voxels.

Thresholding can be also used to greatly simplify a model's task using domain knowledge. Take, for instance, the segmentation of epicardial fat, which refers to the fat in proximity to the heart wall. Epicardial fat is sparsely distributed in a complex shape inside the pericardium. Yet, the pericardium itself has a smooth, elliptical shape and is comparatively easy to segment. By first segmenting the pericardium and subsequently thresholding the pericardium region to the fatty tissue range, we can find epicardial fat without having to segment its complex shape directly [Ben+22].

Region Growing Techniques

Region growing is a voxel-based image segmentation method [AB94]. Beginning from a designated seed voxel, the method assesses neighboring pixels in successive steps. If these pixels meet a specified criterion, often related to pixel values or textures, they are integrated into the region. This process continues until all image pixels are assessed. Despite being a straightforward method, region growing is very sensitive to the selected seed point and may struggle with nuanced transitions between regions. This process is shown in Figure 2.3.

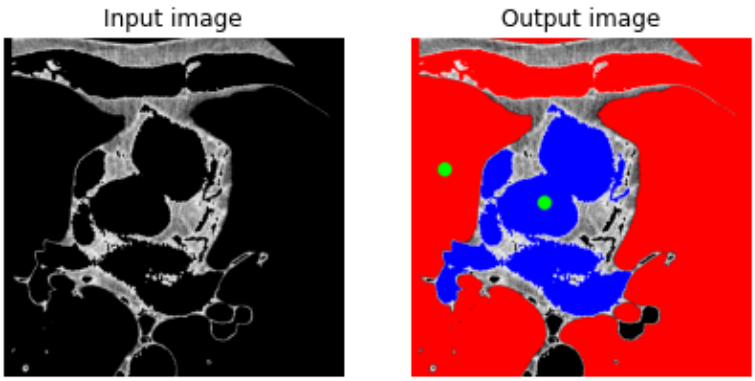


Figure 2.3: A demonstration of region growing for delineating the internal and external areas of the pericardium on a CT slice, set to the adipose tissue intensity range. The left image is the original input, and the right image depicts the segmented outcome with the heart exterior in red and the interior in blue. The green dots represent the manually chosen seed points initiating the region-growing technique. [Ben+22]

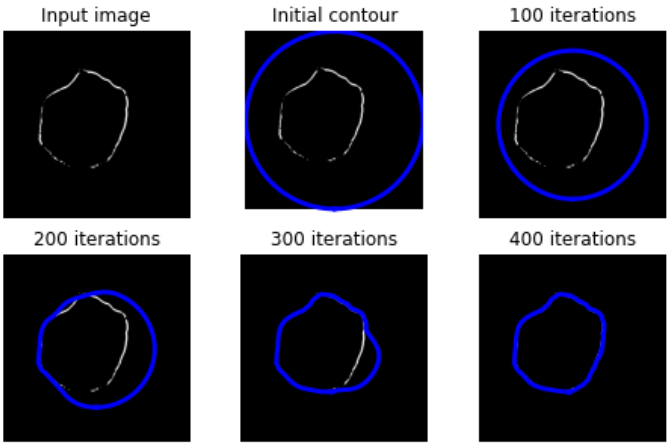


Figure 2.4: A demonstration of employing active contours to complete the absent segments of the pericardium line, displayed in white. The contour, illustrated in blue, starts as a complete circle surrounding the image. With every iteration, the contour adapts more closely to the image's shape. [Ben+22]

Active Contours or Snakes

Active contours, often termed "snakes", are segmentation methods that employ dynamic curves to outline image parts [KWT88]. The process involves tightening a preliminary curve around an object iteratively until it conforms to the object's shape. The adaptation is guided by an energy function that evaluates the curve's smoothness and proximity to edges. A practical depiction of active contours is displayed in Figure 2.4.

Atlas-Based Segmentation

Atlas-based methods, differing from contour-based ones, leverage the spatial relationships among identified structures in an image [Roh+05]. First, a template image is selected and an expert creates an atlas (a segmentation map) by manually segmenting and labeling structures in the template image. Due to anatomical variations, multiple representative images are often merged to produce a template image. The atlas can then be employed to segment new images using a registration algorithm. Image registration is an optimization problem where one image, called the moving image, is deformed to best align with a target image according to some scoring function. The scoring function usually uses the distance between heuristic-based landmarks in the target and moving images to determine how well the two images align. In atlas-based segmentation, a new moving image is deformed to be aligned with the template image that was used to construct the atlas. The atlas can then be used as a segmentation map for the moving image, as it is now aligned with the atlas. The atlas-based registration process is illustrated in Figure 2.5.

A disadvantage of this approach is that the process can lead to a complete failure to segment the image if the target and moving images are too dissimilar. Therefore, atlas-based segmentation has fallen out of favor due to the emergence of deep learning-based methods. However, recently there has been significant progress in image registration and atlas-based segmentation using deep learning-based techniques [Sin+22]. These approaches offer good potential for merging traditional and newer approaches will be discussed later in the chapter.

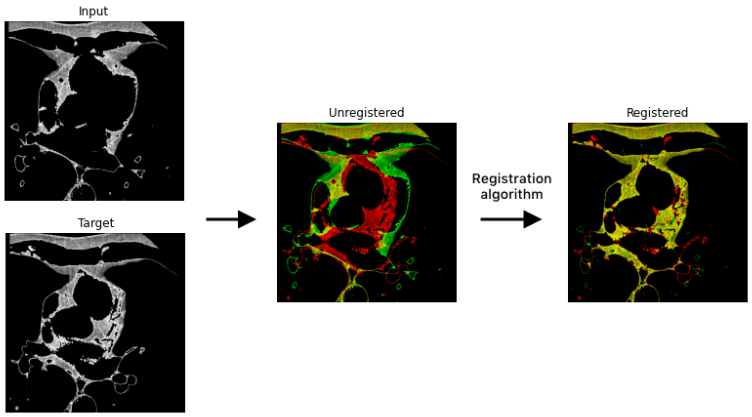


Figure 2.5: A schematic representation of the registration procedure. Initially, input and target images are chosen. Throughout the registration phase, the input image (shown in green) undergoes deformation to align with the fixed target image (shown in red). [Ben+22]

2.2.2 Machine Learning

While deep learning generally falls within the machine learning umbrella term, in this dissertation the term "machine learning" will be used to refer to techniques that are not based on deep neural networks. This encompasses methods such as support vector machines, random forests, and other statistical learning techniques that rely on manually engineered image features.

In this context, image segmentation can be viewed as a voxel-wise classification problem expressed as

$$H(x;\theta) = (\ \Pr(C_0\,|\,x),\ \Pr(C_1\,|\,x),\ \cdots,\ \Pr(C_{K-1}\,|\,x)\ ), \tag{2.4}$$

where H is a classification function parameterized by θ, which maps an input vector x of voxel-level features to the probability that the voxel contains the class C_i. The features are manually constructed based on domain knowledge and represent each pixel and its surrounding region. Commonly, these include the pixel's intensity, mean intensity of the area, image moments, and other information deemed relevant for the classification. The classifier is trained to minimize a predefined loss function by feeding each pixel's features to the classifier and comparing the output to the ground truth output. This is presented visually in Figure 2.6.

Another common approach is to divide the image into smaller patches and then employ a machine learning classifier to categorize each patch into one of the predefined K classes. The classifications are then fused together to form a segmentation map. Both patch- and voxel-based machine learning approaches can be more data efficient than deep learning-based approaches [Ben+22], but they often lack the ability to model complex features and dependencies.

It is clear that these traditional techniques have inspired and paved the way for more complex deep learning methods — while still playing a vital role as parts of segmentation pipelines today. In the next section, we will provide an overview of how deep learning builds on traditional machine learning to achieve more powerful models that can segment very complex regions.

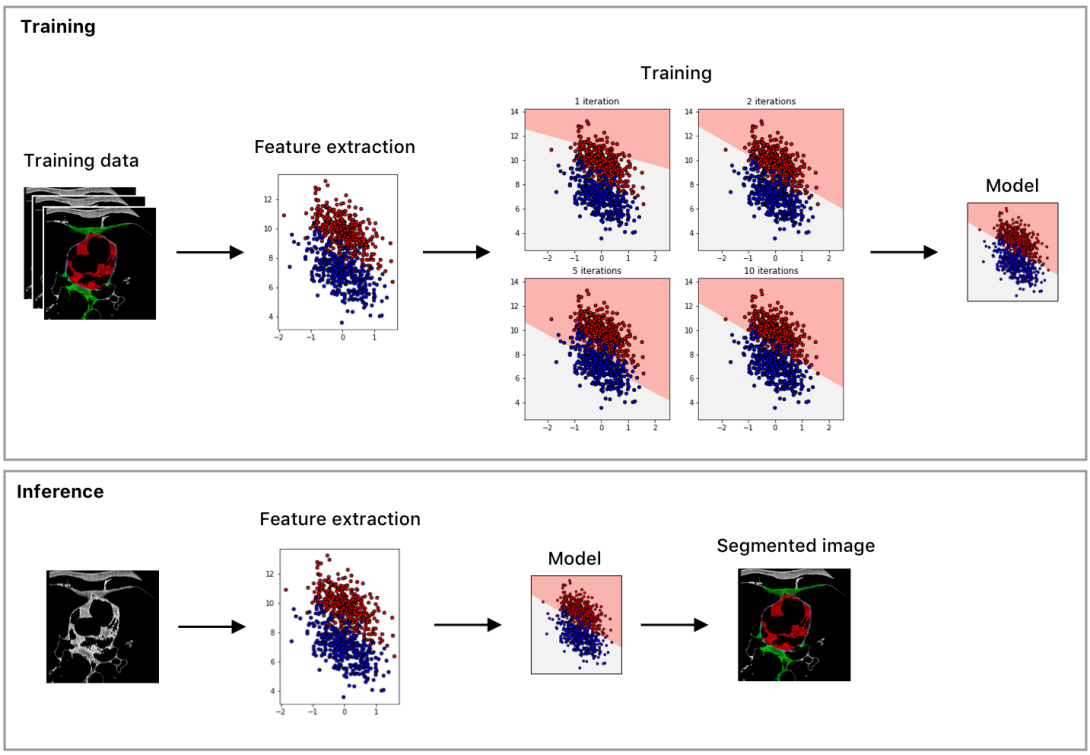


Figure 2.6: A schematic of a supervised linear classifier in a machine learning workflow. The upper section illustrates the training phase. Here, features are color-coded according to their known class from training data, depicted in red and blue. The parameters of the decision boundary, which demarcates the zones of the two classes (highlighted in light red and grey), are determined during training. The lower part of the diagram depicts the inference stage. In this phase, features are extracted from new images, and the trained model is employed to classify each pixel in the image. [Ben+22]

2.3 Deep Learning-Based Segmentation Methods

Machine learning encompasses a broad range of techniques for building statistical models, which includes deep learning — a subset of machine learning that focuses on using artificial neural networks. Artificial neural networks consist of simple nodes called **neurons**. Each neuron is a non-linear function of the sum of its inputs. This non-linear function is called the **activation function**. The neurons are all arranged in a graph where the outputs of one set of neurons are connected to the input of another set of neurons. Each connection has an associated **weight** and **bias** parameter. The value on the connection is multiplied by the weight and the bias is added before it is fed into the subsequent neuron. The exact configuration of the graph, including the number of neurons and how they are connected, is determined by hand and is referred to as the **neural network architecture**. Typically neurons are arranged into **layers**, where neurons of one layer connect only to the next layer. Current deep-learning networks usually have between ten and 100 layers. A typical neural network architecture can be seen in Figure 2.7.

Mathematically, a neuron is a function that maps some input vector to a scalar value. First, the dot product between the input vector $x = [x_0 \; x_1 \; \cdots \; x_{n-1}]$ and the weights

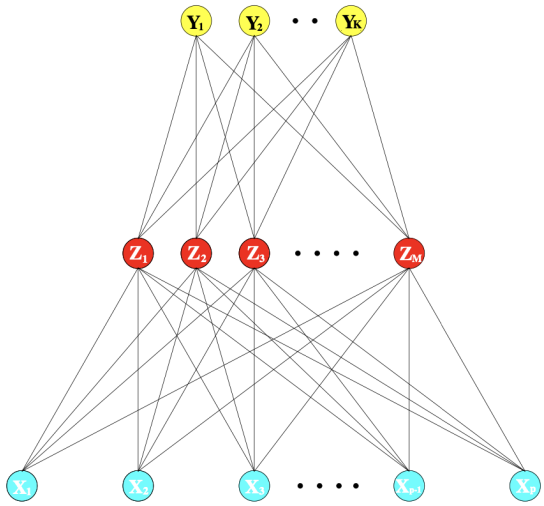


Figure 2.7: A small neural network with three layers, wherein each layer is connected to every neuron in the next layer. [HTF09]

column vector w is calculated and added with the bias vector b:

$$z(x;w,b) = \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \end{bmatrix} \begin{bmatrix} w_0 \\ w_1 \\ \cdots \\ w_{n-1} \end{bmatrix} + \begin{bmatrix} b_0 & b_1 & \cdots & b_{n-1} \end{bmatrix}. \tag{2.5}$$

Then, all of the inputs are summed and the neuron output is produced using the activation function g:

$$f(x;w,b) = g\left(\sum_{i=0}^{n-1} z(x;w,b)_i\right). \tag{2.6}$$

The parameters b and w associated with each neuron can be consolidated into a large matrix, denoted as θ. This matrix is called the **parameters of the network**. Current deep learning networks usually have multiple millions of parameters.

One example of an activation function is the rectified linear unit (ReLU) function:

$$ReLU(x) = \begin{cases} x, & \text{if } x > 0, \\ 0, & \text{otherwise.} \end{cases}. \tag{2.7}$$

ReLU can be seen as a simple thresholding function that sets all negative values to zero. Despite its simplicity, ReLU is one of the most prevalent activation functions in neural networks.

2.3.1 Neural Network Training, Validation and Testing

Initially, the parameters of a neural network are often initialized with random values. These values are then fine-tuned during a procedure called **training**, which iteratively adjusts the parameters to minimize a **loss function**. The loss function measures how well the network is performing its task by comparing its output to known correct values. For example, in image segmentation, the loss function would measure the similarity between the network's segmentation map and a hand-labeled counterpart. Through a process

called **backpropagation**, each parameter is updated in the direction that will decrease the loss. This is repeated multiple times for each image in a **training dataset**.

However, there is a caveat. The training dataset, much like any statistical data, is a sampling of the "real world" data from some unknown distribution. Depending on the size and quality of the dataset, it is possible that the training dataset does not represent the true distribution of the data. Moreover, the network can learn to produce correct solutions for each training image individually instead of learning general patterns in the data. This is called **overfitting** — the model excels on the training data but fails on unseen data. To detect and mitigate this, two additional datasets are introduced: the **validation** and **test** datasets.

The **validation dataset** serves a critical role during model development. After training, the model is assessed using this dataset to inform decisions about its architecture, preprocessing, and other facets. During the development cycle, the network is constantly modified and re-evaluated on the validation dataset. Since adjustments to the model are based on its performance on the validation dataset, there's a risk of inadvertently tailoring the model to the specific distribution of the validation data. To counteract this potential bias, a **test dataset** (often termed a hold-out dataset) is used. This dataset is reserved exclusively for a final evaluation, offering the least biased estimate of the model's real-world performance.

The training, validation, and testing datasets are created before the model development process. Usually, they are sampled randomly from a larger dataset with ratios such as 80%, 10% and 10% for the training, validation, and testing datasets, respectively.

In the next chapter, we will further discuss overfitting in neural networks. For now, we will move on to describing neural network architectures in more detail.

2.3.2 Encoders and Decoders

Machine and deep learning-based segmentation methods can be conceptualized as a two-stage process. The first stage, termed the **encoder** stage, is a function $En : \mathbb{R}^{D \times C} \rightarrow \mathbb{R}^{n \times m \times d}$. In other words, a D-dimensional image with C channels is mapped to a tensor commonly referred to as a feature map. This feature map has an arbitrary size and depth determined by the network architecture. The encoder effectively compresses the image by distilling salient information into a smaller representation, the feature map. In deep learning parlance sometimes this is referred to as the **backbone** of the network.

The output of the encoder then goes to the second stage called the **decoder** or the **head**. The decoder is a function $De : \mathbb{R}^{n \times m \times d} \rightarrow \mathbb{R}^{D \times K}$ that maps a given three-dimensional feature map tensor of size $n \times m \times d$ into a two-dimensional segmentation map of size $D \times K$. Each pixel value of the segmentation map corresponds to the probability of finding the target object on that pixel.

Given an image I, the segmentation process can then be written as

$$M = (En(\theta_{En}) \circ De(\theta_{De}))(I), \tag{2.8}$$

where En and De are functions parameterized by a θ_{En} and θ_{De}, respectively. In conventional machine learning, En is not a trainable function. Instead, θ_{En} consists of hand-selected parameters to extract pre-selected features from the image. The value of θ_{De}, on the other hand, is determined by minimizing a loss function. In deep learning, both θ_{En} and θ_{De} are determined by minimizing a loss function.

Thus, the difference between traditional machine learning and deep learning is in the encoder stage. Deep learning utilizes neural networks to define features, whereas traditional machine learning uses handcrafted functions.

The separation of segmentation models into encoder and decoder stages is a crucial aspect of current research in deep learning. This separation allows for independent improvement of both stages and an easy combination of different encoders and decoders. For instance, a classification and segmentation network could use the same encoder architecture. While the classifier would utilize a simpler decoder to map input features to a class probability vector, the segmentation model would employ a more complex decoder to produce a segmentation map. However, both models would use the exact same encoder. In fact, there are models such as Mask R-CNN [He+17] that use two parallel decoders, one for segmentation and another for object detection, both connected to the same encoder. Throughout the rest of this thesis, when describing neural network architectures, we will describe them in terms of their encoder and decoder and how the two are connected.

2.3.3 Convolutional Neural Networks

Convolutional neural networks have profoundly impacted computer vision, establishing deep learning as the prevailing approach for complex computer vision tasks. The foundational element of a CNN is its convolutional layer, which uses a convolution-like operation in place of the conventional neurons mentioned earlier in this chapter.

Convolution

Generally, convolution is a mathematical operation between two functions. In the context of this thesis, however, we will focus on discrete 2D convolution between two square images, as that is most relevant for image segmentation. Convolution is denoted as

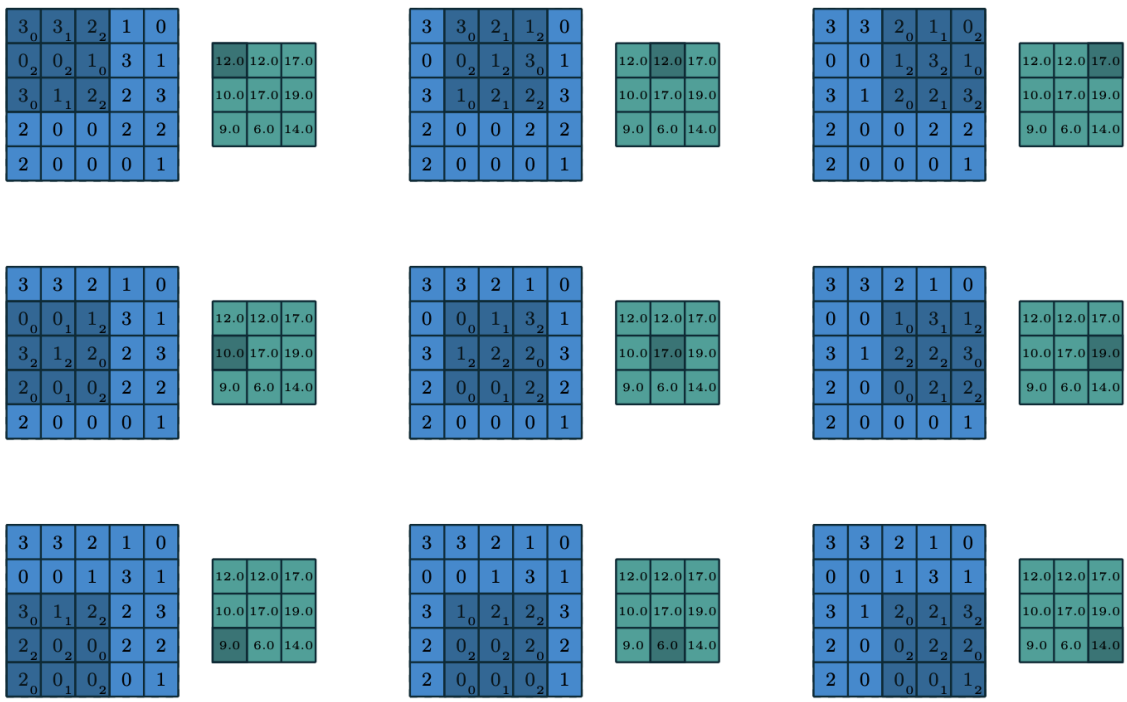


Figure 2.8: A visual depiction of a convolution procedure step-by-step. In each step, the kernel slides over the image (shown in blue). The overlapping elements between the kernel and the image are multiplied and then summed to produce a value in the resultant image (shown in green). [DV18]

$I(A) \star k(B)$ where $I(A)$ is an image $I(A) \in \mathbb{R}^{W \times H}$ and $k(B)$ is a matrix $k(B) \in \mathbb{R}^{w \times h}$ indexed by locations $B \in \mathbb{N}^2$ for called the **convolutional kernel**. At position $A = (a_x, a_y)$, the convolution operation is defined as

$$(I \star k)(a_x, a_y) = \sum_{j=1}^{h} \sum_{i=1}^{w} I(a_x - i, a_y - j) k(i, j). \tag{2.9}$$

Explained differently, the resulting image is produced by sliding the kernel over the input image pixel by pixel. At each pixel location, values where the kernel and the image overlap are multiplied, and all of the products are summed together to form the corresponding pixel's value in the output image. This is shown visually in Figure 2.8.

While mathematically a simple operation, convolution is exceedingly powerful and can produce almost endless transformations of an image. It is most commonly used for filtering — a convolution can elegantly find patterns in the image and increase their intensity in the image. One such example is the convolution with a kernel called the Prewitt operator:

$$I_y = I \star \begin{bmatrix} -1 & 0 & 1 \\ -1 & 0 & 1 \\ -1 & 0 & 1 \end{bmatrix}. \tag{2.10}$$

When convolved with this kernel, the resulting image has high-intensity pixels in regions where vertical edges are present, and low intensity everywhere else. This can be seen in Figure 2.9.

Vertical edges necessarily have to have a large jump in values going from left to right or right to left. Otherwise, there would be no perceptible edge. This kernel takes advantage of that fact to accentuate parts of the image where there is such a jump. It does this by replacing each pixel with the difference between the pixels on its left and its right.

This process happens as follows. For each pixel of the input image, the kernel is placed such that it is centered on that pixel. This means that the values of the pixel as well as its neighbors above and below are all multiplied by zero. The neighbors on the left are multiplied by -1, and the ones on the right are multiplied by 1. Summed together, the result represents the sum of the values on the right of the pixel, minus the sum of the values on the left.

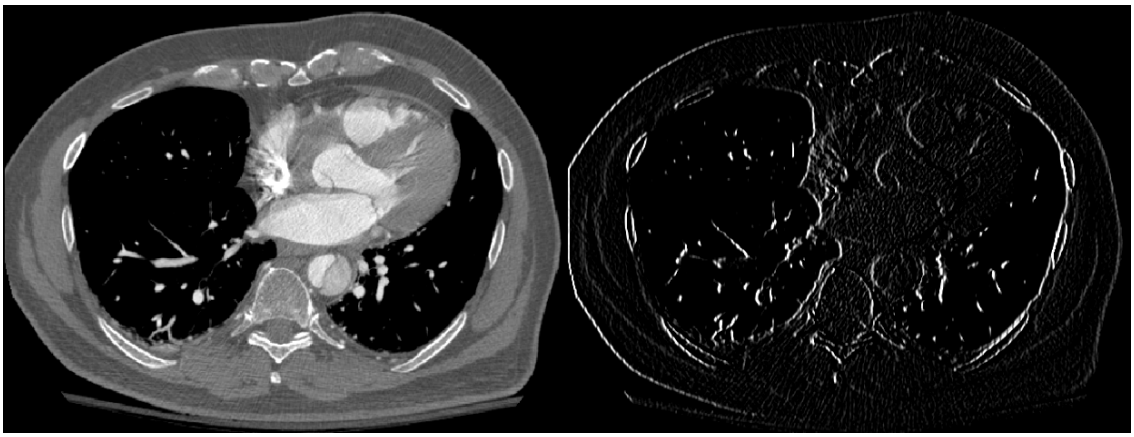


Figure 2.9: An example of an input image (left) convolved with the Prewitt operator (right). Note that vertical edges are accentuated in the convolution result.

To illustrate this, let us consider 1×3 region of the image where no vertical edges are present, skipping the two pixels on the borders during the convolution:

$$\begin{bmatrix}128 & 130 & 136\end{bmatrix} \star \begin{bmatrix}-1 & 0 & 1\end{bmatrix} = \begin{bmatrix}-1 \times 128 + 0 \times 130 + 1 \times 136\end{bmatrix} = \begin{bmatrix}8\end{bmatrix}$$

This section of the image does not contain a vertical edge, so the convolution result is a relatively low value. In a standard image with values in $[0, 255)$, 8 would appear almost completely black.

However, consider some section of the image where a vertical edge is indeed present (again skipping the two pixels on the border of the region):

$$\begin{bmatrix}63 & 66 & 132\end{bmatrix} \star \begin{bmatrix}-1 & 0 & 1\end{bmatrix} = \begin{bmatrix}-1 \times 64 + 0 \times 66 + 1 \times 132\end{bmatrix} = \begin{bmatrix}68\end{bmatrix}$$

The value is now much larger due to the difference between the left and right sides of the image. This example demonstrates how a relatively simple kernel can capture complex features of an image.

Beyond edge detection, there are many commonly used convolution kernels to perform tasks such as blurring, sharpening, or denoising images. A CNN can leverage the power of the convolution by stringing together sequences of intricate kernels to match complex patterns in the image.

Convolutional Layers

A CNN operates by passing an image through a sequence of convolutions. The output from one convolution serves as the input for the next. Interspersed between these convolutions are non-linear transformations of the outputs. This layered approach enables the network to successively find more and more intricate features by combining simpler ones. For instance, combining edges into corners, corners into shapes, and ultimately detecting objects from shapes. The addition of non-linearity drastically increases the complexity of features the network is able to detect and combine.

More technically, in a CNN, convolutional layers are connected to one another similarly to how fully connected layers are connected in fully connected neural networks. Each convolutional layer performs a set number of convolutions, denoted by n. The results are stored in an n-channeled **feature map**. Each channel corresponds to a convolution with a distinct kernel. Hence, a convolutional layer contains n unique kernels. The values of the kernels are the parameters of the layer, optimized during training to minimize a loss function like weights and biases of standard neurons. Finally, a convolutional layer also applies a non-linear activation function to the resulting feature map.

It is worth noting that the operation within a convolutional layer differs from a standard 2D discrete convolution. Here, both the kernel and the input are three-dimensional and have the same depth, but the kernel is usually much smaller in width and height. Since the kernel has the same width as the input, during the operation the kernel only slides along the width and height dimensions. The output of each sliding window step is still a single scalar since all overlapping elements in the sliding window volume are multiplied and summed together. This means that the output of the whole operation is a two-dimensional image. This is visualized in Figure 2.10.

Pooling Layers

Most CNN-based encoders follow a pattern of gradually reducing the width and height of the feature maps while increasing their depth. Depth is increased by giving each successive convolutional layer more kernels. Since each channel represents the result of

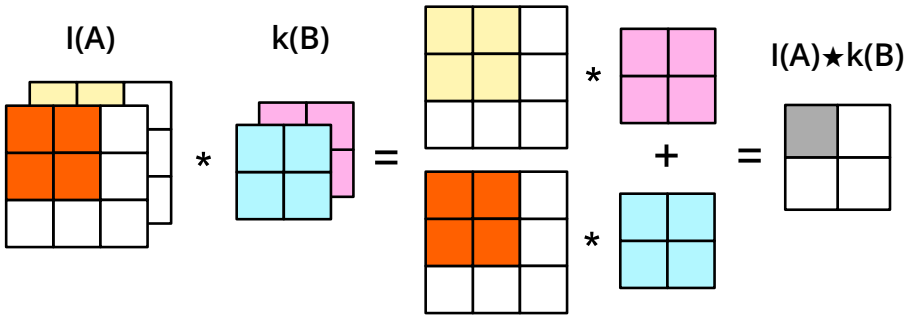


Figure 2.10: A view of one step of a single convolution operation inside a CNN layer. The layer performs multiple convolutions, each with a different kernel that has an equal number of channels as the input image. In each step, the whole kernel slides over the width and height of the image, and the overlapping channels are multiplied together and summed to produce a single output value. The output of the convolution is one channel of a n-channel image where n is the number of different kernels in the layer.

a convolution with a specific kernel, having more kernels directly increases the output depth. To reduce the width and height, **pooling layers** are employed. These layers downsample the image, often by averaging pixel values within a defined neighborhood. This downsampling serves dual purposes in a CNN. Firstly, it compresses the image by gradually removing spatial information and retaining relevant semantic information. Secondly, the network is gradually able to build up large complex features by combining small general features. Such an architecture is shown in Figure 2.11.

To illustrate this, consider a contrived example of a CNN encoder trained to detect grapes on a vine. The first few layers might focus on simple operations such as edge detection, requiring a large amount of spatial information. The next layer could use the edge information to detect circles. This layer does not need as much spatial detail since the edges are already identified. A third layer might then combine the locations of the circles to detect a cluster of grapes. By this stage, the spatial information is even less important, with the overall arrangement of circles being the primary focus.

A convolutional decoder essentially reverses the encoder process. Instead of adding channels, a convolutional decoder successively removes channels up to and upsamples the image. By the end, the output matches the input image's width and height, and the

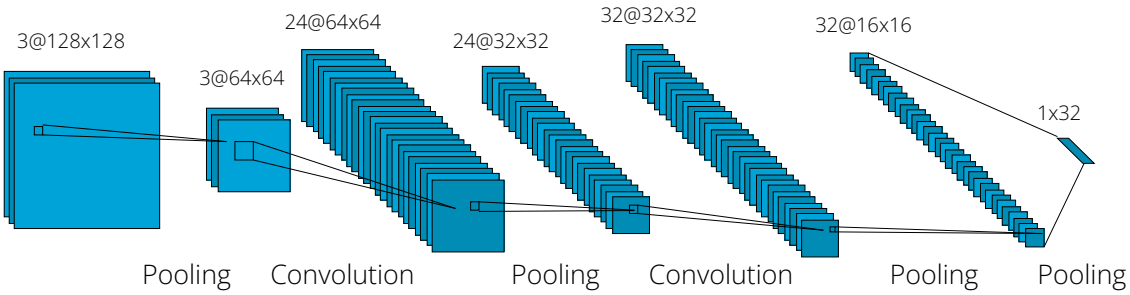


Figure 2.11: A typical architecture of a CNN encoder. The encoder consists of consecutive convolutional and pooling layers that gradually increase the feature map depth and decrease its width and height. The result is a map of features that tells the decoder what features are on the image but does not provide much spatial information about the location of those features. [Lec+98]

number of output channels equals the number of classes.

One way in which upsampling is implemented in a convolutional decoder is with a **transposed convolutional layer**. These layers are very common in segmentation neural networks and allow for dynamic, learned upsampling of the image. Regular convolutions cannot increase the image dimension width and height. To overcome this, a transposed convolution produces an output matrix for each step of the sliding window process, instead of just a scalar value. For instance, a transposed convolution with a 2×2 kernel produces 4 new values for each pixel in the input image. This is done by performing scalar multiplication of the whole kernel and the pixel it is sliding over during each sliding window step. All of the sliding window results are joined as a grid to form a final, larger, image.

To better understand this, consider an example of upscaling a 2×2 2D input image m with a 2×2 kernel k. The transposed convolution can be calculated as follows:

$$m \star^T k = \begin{bmatrix} m_{11}k_{11} & m_{11}k_{12} & 0 \\ m_{11}k_{21} & m_{11}k_{22} & 0 \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & m_{12}k_{11} & m_{12}k_{12} \\ 0 & m_{12}k_{21} & m_{12}k_{22} \\ 0 & 0 & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ m_{21}k_{11} & m_{21}k_{12} & 0 \\ m_{21}k_{21} & m_{21}k_{22} & 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & 0 \\ 0 & m_{22}k_{11} & m_{22}k_{12} \\ 0 & m_{22}k_{21} & m_{22}k_{22} \end{bmatrix}$$

This method can be similarly applied to images and kernels of any size. While transposed convolution is often referred to as *deconvolution*, in this thesis, we will avoid using the term to prevent confusion with the meaning of deconvolution outside of the context of deep learning.

Segmentation models that use only convolutional and pooling layers are called **fully convolutional models** and are one of the most widely used types of models for medical image segmentation. In the following section, we will describe these and other commonly used CNN architectures in medical image segmentation.

2.4 CNN Architectures for Medical Image Segmentation

Deep learning's application in medical image segmentation has predominantly seen the use of convolutional neural networks. This field has witnessed a rapid evolution, giving rise to a diverse range of architectures. While some of these architectures were initially designed for general image classification and segmentation tasks, others have been specifically tailored to overcome challenges specific to medical imaging. In this section, we will highlight the most influential CNN architectures that have shaped medical image segmentation, presented in their chronological development.

2.4.1 Fully Convolutional Network (FCN)

The fully convolutional network (FCN), one of the more straightforward contemporary CNN architectures for image segmentation, was introduced in [LSD15]. As mentioned earlier, convolutional encoders gradually downsample the image while increasing the number of channels in the feature maps. There is then a need to reverse the process and decode the features into an image of equal width and height as the original input image. FCN achieves this using convolutional layers and upscaling.

In FCN, the output of the encoder is interpreted as a coarse heatmap of detected features. An output segmentation map needs to be constructed based on these features and

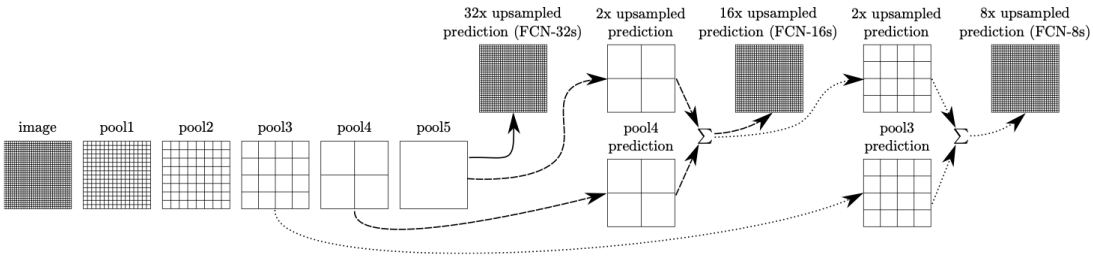


Figure 2.12: A diagram of how FCN forms predictions based on the output of different encoder layers. Encoder layers are shown on the left and the grid represents the coarseness of the feature map. The maps are combined at three different levels to produce three predictions. Each prediction is compared with the ground truth during training, but for inference only the 8x upsampled prediction is used. [LSD15]

have a number of channels equal to the number of classes. FCN does this by using a convolutional layer with 1×1 kernels — one kernel for each class. The resulting convolutions combine the features depth-wise to form a class prediction for each location of the image. However, the resulting segmentation map has a very low spatial resolution. To be useful, it needs to be upsampled using transposed convolution to the original image resolution.

As noted earlier, the encoder's output feature map is only a rudimentary representation of spatial details. To inject additional spatial information into the final segmentation, FCN combines predictions from multiple encoder layers to form the final prediction. This is done by upsampling 1×1 convolution results from different encoder layers to a uniform size and summing them together. The composite prediction is then upsampled to the original image resolution. This process is shown in Figure 2.12.

2.4.2 U-Net and Its Variants

U-Net [RFB15] is one of the most influential and commonly used architectures in medical image segmentation. Aside from being a standard baseline model, a well-tuned U-Net network with data preprocessing has reached state-of-the-art performance in various segmentation challenges across different domains and modalities [Ise+21].

At its core, U-Net follows a straightforward architectural pattern, comprised of a fully convolutional encoder and decoder with only convolutional and pooling layers. The encoder and decoder have a symmetrical structure: the encoder gradually decreases the width and height of the feature maps while increasing the number of channels, while the decoder does the opposite using upsampling or transposed convolutions. While FCN uses only one 1×1 convolutional layer and upsampling to form a prediction, U-Net uses a sequence of several convolutional and upsampling layers to gradually build a segmentation map. This progressive structure allows U-Net to produce more detailed feature maps.

Earlier we mentioned that spatial information is important for the precise localization of the features. Like FCN, U-Net also uses the outputs of different encoder layers to maintain spatial information in the decoder but does so in a different way. In U-Net, the output of every encoder layer is added to the input of its corresponding decoder layer on the other side of the network. This gives each decoder layer concurrent access to information about which features are on the image and where they are located. U-Net's architecture can be seen in Figure 2.13.

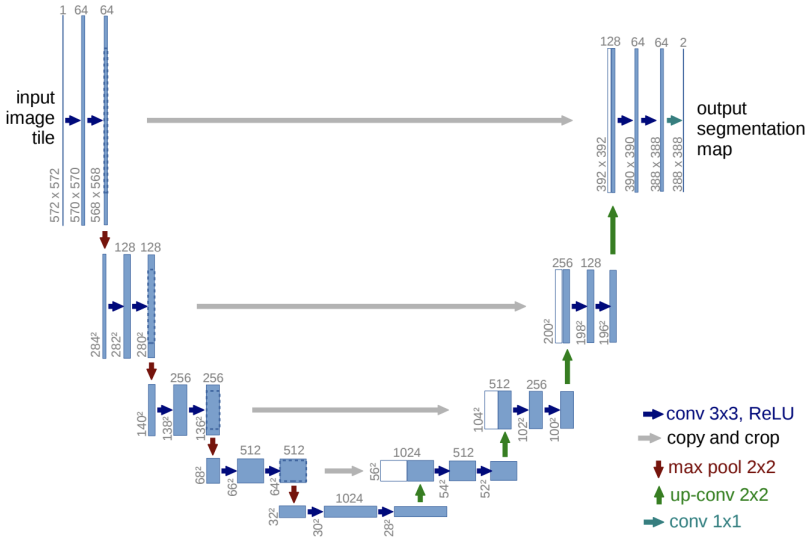


Figure 2.13: A diagram of the U-Net model. The output of each layer of the encoder is concatenated to the input of its corresponding layer in the decoder. [RFB15]

Because of its symmetrical structure and skip connections, U-Net is often visualized in the shape of the letter U, giving it its name. U-Net is only one of a class of networks that are sometimes called U-shaped networks. Aside from being a state-of-the-art architecture for medical image segmentation, U-Net is also widely used as a component of influential deep learning models in other domains such as image generation [Rom+21] and registration [Sin+22].

nnU-Net

U-Net is a powerful baseline model that can be carefully tuned using various heuristics and combined with beneficial data preprocessing to achieve state-of-the-art results, even when compared to much more complex models. However, tuning the parameters of the model and preprocessing is a laborious process that requires experise and empirical research. nnU-Net [Ise+21] aims to automate this tuning process by producing an optimal data pre- and postprocessing pipeline and a U-Net-based architecture for a given dataset. This is achieved through a set of heuristics, rules, and optimization techniques.

The nnU-Net framework begins by calculating key parameters of the data distribution, such as intensity distribution and median image size. Based on these parameters, a set of predefined rules determine the network's architecture and training strategy, including the number of layers, training parameters, and preprocessing approach.

Using the determined parameters, nnU-Net trains three models: one using 2D slices of the data, another using the entire 3D volume, and the third using a cropped 3D region of interest. During inference, outputs from these models are combined through preprocessing to form an ensemble prediction, enhancing the overall accuracy and robustness of the segmentation. This process is shown in Figure 2.14.

nnU-Net has demonstrated exceptional performance, establishing itself as a state-of-the-art solution in a variety of CT and MRI segmentation tasks. nnU-Net is a hugely important tool for both academia and industry, as it allows for creating well-tuned baseline models with very little manual intervention or technical expertise.

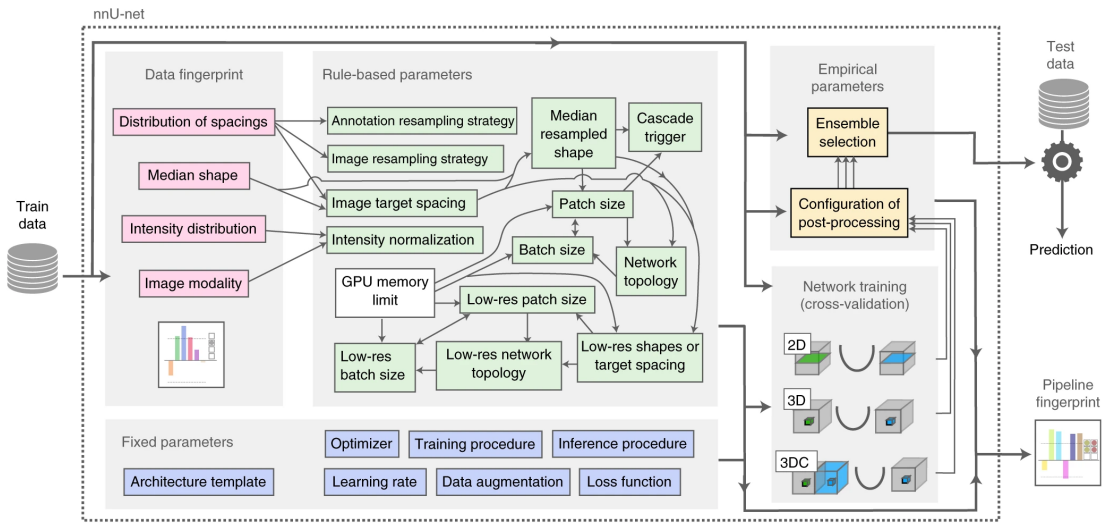


Figure 2.14: A diagram of the nnU-Net procedure of creating a training configuration. [Ise+21]

Furthermore, nnU-Net exemplifies the critical role of preprocessing and parameter tuning in improving model performance, particularly when working with small datasets. This is the cornerstone of this thesis: improving data efficiency through preprocessing. The success of nnU-Net serves as a validation of this approach.

U-Net++

In standard U-Net, skip connections link encoder layer outputs to decoder layer inputs through basic concatenation operations. This approach leaves room for more intricate integration of the decoder and encoder outputs. Additionally, U-Net only merges features at corresponding levels of the encoder and decoder. An enhancement to this could be to allow each decoder layer access to features from multiple encoder layers. U-Net++ [Zho+19] takes advantage of these opportunities with a flexible architectural design.

In U-Net++, skip connections are reimagined as a network of convolutional layers. This setup connects each encoder layer to its respective decoder layer through multiple convolutional layers. This enhances the information fed into the skip convolutional layers. Additionally, the outputs of these skip connections are further upscaled and fed into the subsequent decoder layers. This design results in a significantly more complex network architecture. Here, each decoder layer not only receives features from its direct encoder counterpart but also from all deeper encoder layers. This multifaceted connection system is visually represented in Figure 2.15.

In U-Net++, skip connections are reimagined as a network of convolutional layers instead of simple concatenation. Moreover, outputs from each encoder layer (except the first) are upscaled and then supplied as additional inputs to the skip connections of the preceding layer. The outputs of the skip connections themselves are also upscaled and given to the next corresponding decoder layer. Thus, each decoder layer not only receives features from its direct encoder counterpart but also from all deeper encoder layers. This architecture is presented in Figure 2.15.

There are several advantages to U-Net++ over U-Net. Each decoder layer has access to multi-scale features of the input image and thus can produce more accurate outputs. Conceptually, U-Net++ can be viewed as an ensemble of U-Nets with varying depths,

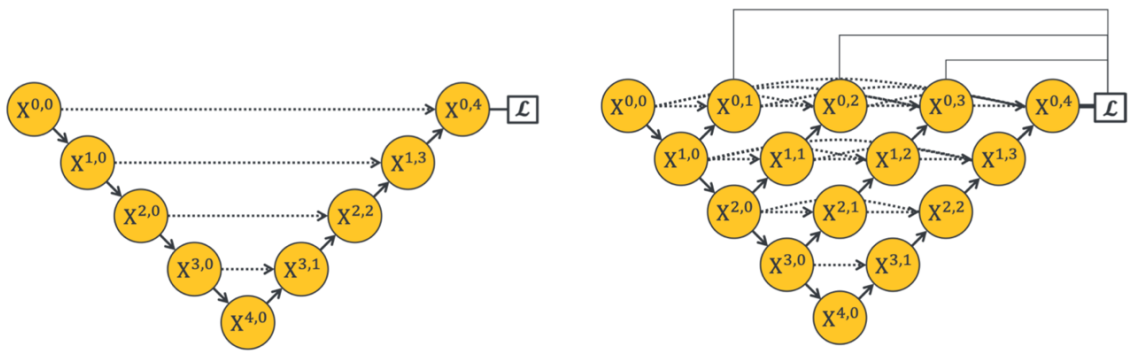


Figure 2.15: A comparison between U-Net (left) and U-Net++ (right). Each node in the graph represents a convolutional layer. The dashed arrows represent skip connections, while full arrows are downsampling or upsampling operations. [Zho+19]

which helps in faster convergence compared to a standard U-Net of similar depth. Further, instead of finding the optimal U-Net depth, due to the flexibility of neural networks, U-Net++ learns the optimal depth during training. This makes U-Net++ more flexible, requiring fewer design decisions. However, U-Net++ has a higher parameter count and does not necessarily enhance data efficiency. In some cases, its performance benefits can be matched or surpassed by a well-preprocessed and tuned U-Net [Ise+21].

2.4.3 Mask R-CNN

Earlier in the chapter we mentioned that CNN-based segmentation networks generally consist of encoder and decoder stages. Segmentation architectures such as U-Net employ a decoder that progressively upsamples the feature maps to construct a segmentation map. This differs from how object detection decoders work.

Object detection aims to find the location of an object in an image. Instead of classifying each pixel as in segmentation, object detection typically outputs the coordinates of a bounding box surrounding the object. In technical terms, the output is not an image but rather a vector of four numbers for each object. As such, the decoders in object detection networks are usually much shallower and employ fully connected layers instead of convolutional layers.

Despite these differences, both object detection and segmentation networks share similar encoder designs. Mask R-CNN [He+17] takes advantage of this to combine object detection and segmentation into a single network. In Mask R-CNN, a standard CNN encoder generates a feature map from an input image. This map is then given to two parallel decoders: one for segmentation and the other for object detection. The segmentation decoder, akin to those in standard segmentation networks, uses convolutional layers and upsampling to generate a segmentation map for each object. The detection decoder, in contrast, employs downsampling and fully connected layers to output the bounding box and class label of each object. The Mask R-CNN architecture can be seen in Figure 2.16.

Consequently, for every detected object, Mask R-CNN provides a bounding box, a class label, and a segmentation mask. The network is trained using a composite loss function, optimizing for segmentation, detection, and classification simultaneously.

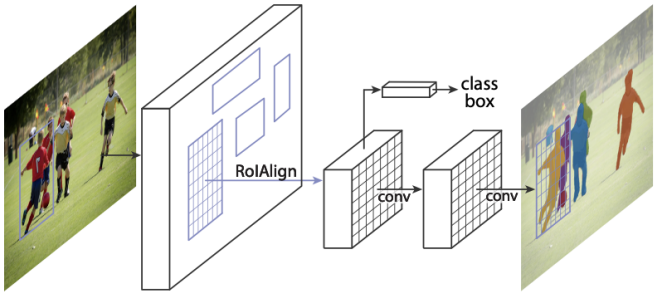


Figure 2.16: A diagram of the Mask R-CNN architecture. Two parallel decoder branches are used to achieve segmentation and object detection simultaneously. [He+17]

The integration of these tasks in Mask R-CNN takes advantage of synergies between them. The information learned for classifying an object is relevant to its detection. Similarly, by using a bounding box the segmentation decoder can focus on the relevant portion of the image and be invariant to the scale of the object on the image. This joint learning approach enhances performance across all tasks compared to training separate networks for each task. However, Mask R-CNN's complexity can pose challenges, particularly when trained on very small datasets, where simpler segmentation-focused architectures like U-Net might be more effective.

2.4.4 Other Notable Segmentation CNNs

A central challenge in CNN-based segmentation is balancing spatial information with feature complexity. As the network increases the feature map depth and decreases width and height, spatial information is erased but more complex features can be detected on the image. Effective segmentation requires both elements. While U-Net addresses this with skip connections, other architectures use different methods to integrate multi-scale features.

One approach involves pyramid pooling, as utilized in PSPNet [Zha+17]. Initially, PSPNet uses a standard encode to create a feature map. This map is then rescaled to various sizes, and each rescaled map undergoes processing through a series of convolutional layers. The resulting outputs from these layers are resized to a uniform spatial dimension and concatenated. This combined feature map is then fed into a decoder to produce the final segmentation map.

Pyramid pooling is also employed by DeepLabV3 [Che+17] albeit in a different way. Like in PSPNet, the encoder first creates a feature map. This feature map then goes through a series of parallel layers that each use an atrous convolution. In atrous convolution, the kernel is dilated by introducing gaps between its values. During each sliding window step, the gaps are ignored and the result is convolved regularly for non-empty kernel values. This technique enables covering a larger area of the image without increasing kernel parameters. Each parallel layer uses different gap sizes in the kernel, creating multi-scale features. These features are upscaled, merged, and further decoded into a segmentation map.

MA-Net [Fan+20] takes a different route by incorporating an attention mechanism for combining multi-scale features. Rather than directly concatenating in skip connections like U-Net, MA-Net introduces a multi-scale attention block. This block aims to capture the interdependencies of different features to focus on more salient ones before they are concatenated to the decoder layer input. Additionally, the final decoder layer

is processed through a position-wise attention block to highlight significant features in specific image locations. MA-Net has shown performance improvements in liver tumor segmentation on CT images compared to U-Net and U-Net++.

2.5 Fully Connected Transformers for Medical Image Segmentation

So far most of this chapter dealt with CNNs. However, another class of networks — transformers [Vas+17] — has recently gained traction and even surpassed the segmentation quality of CNNs for many tasks. The key to the transformer's impressive performance is the ability to support very deep neural networks. Empirical observations suggest that with enough data, network performance tends to increase with the depth of the network. Hence, transformers have become state-of-the-art in tasks where a large amount of data is available. Notably, current transformers used in natural language processing contain more than 100 layers and trillions of parameters, orders of magnitude more than CNN-based models.

Transformers first became broadly used in the field of natural language processing and achieved groundbreaking results in text generation, machine translation, and understanding natural language. Broadly, transformers employ a combination of regular (non-convolutional) fully connected layers and attention mechanisms, which are the key to their success. Attention allows a layer to learn the interdependencies between parts of its input or across different input values. In linguistic contexts, this leads to a more nuanced understanding of syntactic relationships, enabling the network to learn concepts such as the interplay between subjects and objects or the likelihood of certain adjectives preceding specific nouns.

2.5.1 Attention

For each element x_i of an input sequence of length n, an attention mechanism aims to produce a vector $v_i = (v_{i1}, v_{i2}, \cdots, v_{in})$ where v_{ij} measures how dependent x_i is on x_j. In a natural language scenario, if we consider a sentence as our input sequence, this mechanism can be envisioned as going word by word, and for each word producing relevance scores for every other word in the sentence. For instance, in the sentence "Susan has a brown dog that she loves.", the word "she" should have a high relevance score for "Susan", and a low score for "brown" since it is not pertinent to encoding the meaning of "she" in this sentence.

Mathematically, attention can be expressed as a series of matrix multiplications. First, each input sequence element is transformed into a query vector. This transformation represents a learned encoding by the network, converting an input element into a lower-dimension vector. The query serves to identify the current input element in the sequence. Let X be an input matrix where each row is a new element of the sequence. Let W_Q be a matrix of weights learned by the network that encodes each element of the input sequences into a vector. The query matrix Q can then be calculated as

$$Q = W_Q X, \tag{2.11}$$

where each row Q_i represents the query for X_i. Remember that the goal of attention is to score the relevance between an input element X_i and some other element X_j of the same input sequence. Having encoded each element X_i in the form of Q_i, we need to similarly encode each X_j in a different matrix K where K_j is called the **key** for some input sequence

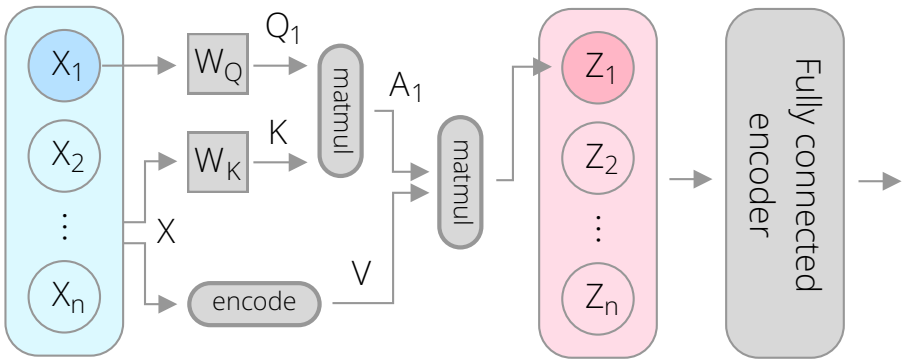


Figure 2.17: A visualization of a single encoder layer in a transformer network. This shows the encoding process for one element of the input sequence.

element X_j. We can do so as

$$K = W_K X, \tag{2.12}$$

where W_K is another weight matrix that the model optimizes during training. Given K and Q, we can compute the relevance scores for every combination of input elements x_i and x_j as

$$A = QK^T. \tag{2.13}$$

Here, A_{ij} represents the relevance score between x_j and x_i. It is important to note that this is only one of many ways the matrix A can be calculated. In practice, A is scaled to be in $[0, 1]$ or transformed in some other way.

Finally, to get the output Z of the self-attention layer, we weigh each element of the original input sequence X by the attention scores A:

$$Z = AV, \tag{2.14}$$

Z consists of X elements weighted by how relevant they are to the encoding of the whole sequence. Visually, this process for one element of the input sequence can be seen in Figure 2.17.

Encoder and decoder layers in a transformer include an attention layer followed by a fully connected layer. In the encoder, the attention layers are generally *self-attention*, where both the key and the query come from the input sequence. In contrast, the decoder layers employ both self-attention and **encoder-decoder attention**. In the latter, the keys and values come from the encoded features of the last layer of the encoder. Finally, each attention layer performs Z calculation multiple times using different W and K weights, and the weighted values are concatenated together. This design enables the network to simultaneously learn and apply various types of attention within each individual layer.

2.5.2 Positional Encoding

The position of each word in a sentence carries important information. The sentence "The yellow notebook is on top of the red notebook." has a different meaning than "The red notebook is on top of the yellow notebook.", even though both sentences contain the same words. To capture this in transformers, the position of each input element is encoded and included as part of the input.

Each input sequence element index is mapped to a matrix via a function $f : \mathbb{Z}^+ \to \mathbb{R}^d$ where d is the dimensionality of the layer's input. Mapped that way, the positional

encoding can simply be concatenated to the layer's input with no additional changes in the network. While it is possible for this encoding function to be represented as neural network layers and learned during training, a common approach is to use a predefined, fixed function. For instance, in citeattnAllYouNeed, given an element index pos, the i-th element of the positional encoding is calculated as

$$PE(pos, i) = \begin{cases} \sin(pos/1000^{2i/d}), & i \text{ is even} \\ \cos(pos/1000^{2i/d}), & i \text{ is odd} \end{cases}. \tag{2.15}$$

The choice of positional encoding functions is guided by their numerical characteristics, ensuring they are efficiently processed by the neural network.

2.5.3 Adapting Transformers to Image Segmentation

Convolutional Neural Networks (CNNs) have traditionally dominated image processing due to their efficiency in handling large inputs. Consider a 128×128 pixel image which, when flattened into a row vector, results in 16,384 elements. In a fully connected network, where each input element connects to every output element, this translates into $16,384^2$ parameters for just one layer. Even if the image were to be heavily downscaled and ignore the loss of information this would create, managing such a network on current GPUs is infeasible.

Transformers that use fully connected layers face a challenge when applied to image data due to this very issue. The key solution, as utilized in the Vision Transformer (ViT) [Dos+21], is treating an image as a sequence of small patches, allowing the image to be processed similarly to text.

ViT, primarily designed for image classification, begins by partitioning each image into uniform square patches (e.g. 32×32 pixels). The patches, ordered starting from the top-left of the image, form an input sequence X. Each patch is then flattened to a row vector and given to a fully connected layer to encode it as a d-dimensional vector. Then, a d-dimensional positional encoding is added to the encoding as a function of the patch index. The resultant matrix serves as the input for the encoder, which operates identically to a standard transformer network. This architecture can be seen visually in Figure 2.18.

While ViT originally used a classification decoder, it is possible to combine the ViT encoder with a segmentation decoder to obtain a segmentation map. There are several ways this has been achieved. For instance, the Segmentation Transformer (SETR) [Zhe+21] introduces two distinct CNN-based segmentation decoders connected to a ViT encoder. The first variant follows a conventional convolutional decoder design, where the encoder's output goes through a series of upsampling and convolutional layers, producing a segmentation map in the original image resolution. The second decoder variant is similar to the way FCN works — outputs from various encoder layers are resized to a uniform resolution, merged, and then decoded into a segmentation map.

Later transformer-based segmentation networks modified the ViT encoder as well. For instance, the authors of the Swin Transformer [Liu+21] argue that neural networks for computer vision need to be built with translation and scale invariance in mind. Swin Transformer starts with small patches, similar to ViT, but in each successive encoder layer neighboring patches are merged. Rather than computing self-attention within individual patches, Swin uses non-overlapping windows of multiple patches to perform self-attention. This ensures that Swin's computational demand increases linearly with image size, a significant improvement over ViT's exponential growth. A key aspect of

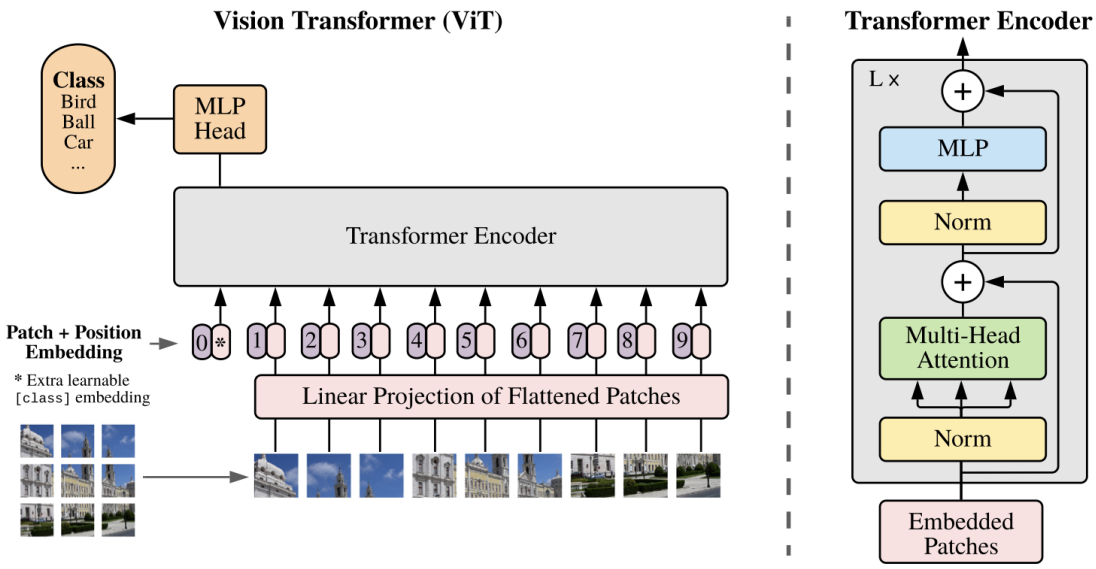


Figure 2.18: The ViT architecture for image classification. [Dos+21]

Swin is that each successive layer shifts the window by half of the window size. This allows self-attention between two different windows and more global information about the image. Finally, Swin's encoder output is fed into a DeepLab-like decoder to produce the segmentation map.

Transformers have demonstrated exceptional performance in image segmentation, particularly in large datasets and benchmarks with upwards of 20,000 natural images [Liu+21; Zhe+21; Che+22]. However, their advantage is less pronounced in smaller datasets. This aligns with observations from [Vas+17] — convolutional layers have properties that are naturally a good fit for extracting information out of images. Transformers need to compensate for their lack of convolutions by using more layers. However, as we will see in the next chapter, there is a relationship between the number of parameters of a network and the number of samples required to train that network.

References

[AB94] R. Adams and L. Bischof. "Seeded Region Growing". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 16.6 (1994), pp. 641–647. DOI: 10.1109/34.295913.

[Ben+22] Marin Benčević, Irena Galić, Marija Habijan, and Aleksandra Pižurica. "Recent Progress in Epicardial and Pericardial Adipose Tissue Segmentation and Quantification Based on Deep Learning: A Systematic Review". In: *Applied Sciences* 12.10 (May 2022), p. 5217. ISSN: 2076-3417. DOI: 10.3390/app12105217.

[BM17] Yahya Baba and Andrew Murphy. "Windowing (CT)". In: *Radiopaedia.Org*. Radiopaedia.org, Mar. 2017. DOI: 10.53347/rID-52108. (Visited on 06/18/2024).

[Boa+22] Mateus Boaventura, Jaume Sastre-Garriga, Aran Garcia-Vidal, Angela Vidal-Jordana, Davide Quartana, et al. "T1/T2-weighted Ratio in Multiple Sclerosis: A Longitudinal Study with Clinical Associations". In: *NeuroImage: Clinical* 34 (2022), p. 102967. ISSN: 22131582. DOI: 10.1016/j.nicl.2022.102967. (Visited on 06/18/2024).

[Cal+22] Evan Calabrese, Javier E. Villanueva-Meyer, Jeffrey D. Rudie, Andreas M. Rauschecker, Ujjwal Baid, et al. "The University of California San Francisco Preoperative Diffuse Glioma MRI (UCSF-PDGM) Dataset". In: *Radiology: Artificial Intelligence* 4.6 (Nov. 2022), e220058. ISSN: 2638-6100. DOI: 10.1148/ryai.220058. arXiv: 2109.00356 [cs, eess]. (Visited on 10/05/2023).

[Cha+87] Alfred E. Chang, Yvedt L. Matory, Andrew J. Dwyer, Suvimol C. Hill, Mary E. Girton, et al. "Magnetic Resonance Imaging Versus Computed Tomography in the Evaluation of Soft Tissue Tumors of the Extremities:" in: *Annals of Surgery* 205.4 (Apr. 1987), pp. 340–348. ISSN: 0003-4932. DOI: 10.1097/00000658-198704000-00002. (Visited on 06/18/2024).

[Che+17] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: 1706.05587 [cs.CV].

[Che+22] Zhe Chen, Yuchen Duan, Wenhai Wang, Junjun He, Tong Lu, et al. "Vision Transformer Adapter for Dense Predictions". In: *arXiv preprint arXiv:2205.08534* (2022). arXiv: 2205.08534.

[Dos+21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, et al. "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *International Conference on Learning Representations*. 2021.

[DV18] Vincent Dumoulin and Francesco Visin. *A Guide to Convolution Arithmetic for Deep Learning*. Jan. 2018. arXiv: 1603.07285 [cs, stat]. (Visited on 10/19/2023).

[Fan+20] Tongle Fan, Guanglei Wang, Yan Li, and Hongrui Wang. "MA-Net: A Multi-Scale Attention Network for Liver and Tumor Segmentation". In: *IEEE access : practical innovations, open solutions* 8 (2020), pp. 179656–179665. DOI: 10.1109/ACCESS.2020.3025372.

[FO11] R. Fosbinder and D. Orth. *Essentials of Radiologic Science*. Wolters Kluwer Health/Lippincott Williams & Wilkins, 2011. ISBN: 978-0-7817-7554-0.

[Gol07a] L. W. Goldman. "Principles of CT and CT Technology". In: *Journal of Nuclear Medicine Technology* 35.3 (Sept. 2007), pp. 115–128. ISSN: 0091-4916. DOI: 10.2967/jnmt.107.042978. (Visited on 06/18/2024).

[Gol07b] L. W. Goldman. "Principles of CT: Radiation Dose and Image Quality". In: *Journal of Nuclear Medicine Technology* 35.4 (Dec. 2007), pp. 213–225. ISSN: 0091-4916. DOI: 10.2967/jnmt.106.037846. (Visited on 06/18/2024).

[Gro+15] Vijay P.B. Grover, Joshua M. Tognarelli, Mary M.E. Crossey, I. Jane Cox, Simon D. Taylor-Robinson, et al. "Magnetic Resonance Imaging: Principles and Techniques: Lessons for Clinicians". In: *Journal of Clinical and Experimental Hepatology* 5.3 (Sept. 2015), pp. 246–255. ISSN: 09736883. DOI: 10.1016/j.jceh.2015.08.001. (Visited on 06/18/2024).

[He+17] Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. Venice: IEEE, Oct. 2017, pp. 2980–2988. ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.322. (Visited on 10/09/2023).

[HTF09] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. Springer Series in Statistics. New York, NY: Springer New York, 2009. ISBN: 978-0-387-84857-0 978-0-387-84858-7. DOI: 10.1007/978-0-387-84858-7. (Visited on 10/21/2023).

[Irv+19] Jeremy Irvin, Pranav Rajpurkar, Michael Ko, Yifan Yu, Silviana Ciurea-Ilcus, et al. *CheXpert: A Large Chest Radiograph Dataset with Uncertainty Labels and Expert Comparison*. Jan. 2019. arXiv: 1901.07031 [cs, eess]. (Visited on 10/02/2023).

[Ise+21] Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. "nnU-Net: A Self-Configuring Method for Deep Learning-Based Biomedical Image Segmentation". In: *Nature Methods* 18.2 (Feb. 2021), pp. 203–211. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-020-01008-z.

[Jha+21] Aadarsh Jha, Haichun Yang, Ruining Deng, Meghan E. Kapp, Agnes B. Fogo, et al. "Instance Segmentation for Whole Slide Imaging: End-to-End or Detect-Then-Segment". In: *Journal of Medical Imaging* 8.01 (Jan. 2021). ISSN: 2329-4302. DOI: 10.1117/1.JMI.8.1.014001. (Visited on 10/02/2023).

[Kha+18] Arif Ul Maula Khan, Angelo Torelli, Ivo Wolf, and Norbert Gretz. "AutoCellSeg: Robust Automatic Colony Forming Unit (CFU)/Cell Analysis Using Adaptive Image Segmentation and Easy-to-Use Post-Editing Techniques". In: *Scientific Reports* 8.1 (May 2018), p. 7302. ISSN: 2045-2322. DOI: 10.1038/s41598-018-24916-9. (Visited on 10/16/2023).

[KLG16] Shervin Kamalian, Michael H. Lev, and Rajiv Gupta. "Computed Tomography Imaging and Angiography – Principles". In: *Handbook of Clinical Neurology*. Vol. 135. Elsevier, 2016, pp. 3–20. ISBN: 978-0-444-53485-9. DOI: 10.1016/B978-0-444-53485-9.00001-5. (Visited on 09/29/2023).

[KWT88] Michael Kass, Andrew Witkin, and Demetri Terzopoulos. "Snakes: Active Contour Models". In: *International Journal of Computer Vision* 1.4 (Jan. 1988), pp. 321–331. DOI: 10.1007/bf00133570.

[Lec+98] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-Based Learning Applied to Document Recognition". In: *Proceedings of the IEEE* 86.11 (Nov./1998), pp. 2278–2324. ISSN: 00189219. DOI: 10.1109/5.726791. (Visited on 10/05/2023).

[Li+23] Ming Li, Jiping Wang, Yang Chen, Yufei Tang, Zhongyi Wu, et al. "Low-Dose CT Image Synthesis for Domain Adaptation Imaging Using a Generative Adversarial Network With Noise Encoding Transfer Learning". In: *IEEE Transactions on Medical Imaging* 42.9 (Sept. 2023), pp. 2616–2630. ISSN: 0278-0062, 1558-254X. DOI: 10.1109/TMI.2023.3261822. (Visited on 06/18/2024).

[Liu+21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, et al. "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows". In: *arXiv preprint arXiv:2103.14030* (2021). arXiv: 2103.14030.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully Convolutional Networks for Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2015, pp. 3431–3440.

[Mac96] Albert Macovski. "Noise in MRI". In: *Magnetic Resonance in Medicine* 36.3 (Sept. 1996), pp. 494–497. ISSN: 0740-3194, 1522-2594. DOI: 10.1002/mrm.1910360327. (Visited on 06/18/2024).

[Mah+17] Amir Abbas Mahabadi, Bastian Balcer, Iryna Dykun, Michael Forsting, Thomas Schlosser, et al. "Cardiac Computed Tomography-Derived Epicardial Fat Volume and Attenuation Independently Distinguish Patients with and without Myocardial Infarction". In: *PLOS ONE* 12.8 (Aug. 2017). Ed. by Marc W. Merx, e0183514. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0183514. (Visited on 09/29/2023).

[Mot+21] Margherita Mottola, Stephan Ursprung, Leonardo Rundo, Lorena Escudero Sanchez, Tobias Klatte, et al. "Reproducibility of CT-based Radiomic Features against Image Resampling and Perturbations for Tumour and Healthy Kidney in Renal Cancer Patients". In: *Scientific Reports* 11.1 (June 2021), p. 11542. ISSN: 2045-2322. DOI: 10.1038/s41598-021-90985-y. (Visited on 06/18/2024).

[Ngu+20] Ha Q. Nguyen, Khanh Lam, Linh T. Le, Hieu H. Pham, Dat Q. Tran, et al. *VinDr-CXR: An Open Dataset of Chest X-rays with Radiologist's Annotations*. 2020. arXiv: 2012.15029 [eess.IV].

[Rad+22] Lukas Radl, Yuan Jin, Antonio Pepe, Jianning Li, Christina Gsaxner, et al. "AVT: Multicenter Aortic Vessel Tree CTA Dataset Collection with Ground Truth Segmentation Masks". In: *Data in Brief* 40 (Feb. 2022), p. 107801. ISSN: 23523409. DOI: 10.1016/j.dib.2022.107801. (Visited on 10/05/2023).

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. May 2015. arXiv: 1505.04597 [cs]. (Visited on 10/06/2023).

[Roh+05] Torsten Rohlfing, Robert Brandt, Randolf Menzel, Daniel B. Russakoff, and Calvin R. Maurer. "Quo Vadis, Atlas-Based Segmentation?" In: *Handbook of Biomedical Image Analysis*. Springer US, 2005, pp. 435–486. DOI: 10.1007/0-306-48608-3_11.

[Rom+21] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. *High-Resolution Image Synthesis with Latent Diffusion Models*. 2021. arXiv: 2112.10752 [cs.CV].

[Rot+21] Veronica Rotemberg, Nicholas Kurtansky, Brigid Betz-Stablein, Liam Caffery, Emmanouil Chousakos, et al. "A Patient-Centric Dataset of Images and Metadata for Identifying Melanomas Using Clinical Context". In: *Scientific Data* 8.1 (Jan. 2021), p. 34. ISSN: 2052-4463. DOI: 10.1038/s41597-021-00815-z. (Visited on 10/02/2023).

[Sei04] J. Anthony Seibert. "X-Ray Imaging Physics for Nuclear Medicine Technologists. Part 1: Basic Principles of x-Ray Production". In: *Journal of Nuclear Medicine Technology* 32.3 (Sept. 2004), pp. 139–147. ISSN: 0091-4916.

[Sin+22] Matthew Sinclair, Andreas Schuh, Karl Hahn, Kersten Petersen, Ying Bai, et al. "Atlas-ISTN: Joint Segmentation, Registration and Atlas Construction with Image-and-Spatial Transformer Networks". In: *Medical Image Analysis* 78 (May 2022), p. 102383. ISSN: 13618415. DOI: 10.1016/j.media.2022.102383. (Visited on 10/03/2023).

[SSC07] A. Stecco, A. Saponaro, and A. Carriero. "Patient Safety Issues in Magnetic Resonance Imaging: State of the Art". In: *La radiologia medica* 112.4 (June 2007), pp. 491–508. ISSN: 0033-8362, 1826-6983. DOI: 10.1007/s11547-007-0154-4. (Visited on 06/18/2024).

[Str+02] Gary Strangman, Joseph P. Culver, John H. Thompson, and David A. Boas. "A Quantitative Comparison of Simultaneous BOLD fMRI and NIRS Recordings during Functional Brain Activation". In: *NeuroImage* 17.2 (Oct. 2002), pp. 719–731. ISSN: 10538119. DOI: 10.1006/nimg.2002.1227. (Visited on 06/18/2024).

[TSD08] Susan Thorpe, Paul M. Salkovskis, and Antonia Dittner. "Claustrophobia in MRI: The Role of Cognitions". In: *Magnetic Resonance Imaging* 26.8 (Oct. 2008), pp. 1081–1088. ISSN: 0730725X. DOI: 10.1016/j.mri.2008.01.022. (Visited on 06/18/2024).

[Vas+17] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, et al. "Attention Is All You Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, et al. Vol. 30. Curran Associates, Inc., 2017.

[Wah+21] Kareem A. Wahid, Renjie He, Brigid A. McDonald, Brian M. Anderson, Travis Salzillo, et al. "Intensity Standardization Methods in Magnetic Resonance Imaging of Head and Neck Cancer". In: *Physics and Imaging in Radiation Oncology* 20 (Oct. 2021), pp. 88–93. ISSN: 24056316. DOI: 10.1016/j.phro.2021.11.001. (Visited on 06/18/2024).

[Wal+19] Justine Wallyn, Nicolas Anton, Salman Akram, and Thierry F. Vandamme. "Biomedical Imaging: Principles, Technologies, Clinical Aspects, Contrast Agents, Limitations and Future Trends in Nanomedicines". In: *Pharmaceutical Research* 36.6 (Apr. 2019). ISSN: 1573-904X. DOI: 10.1007/s11095-019-2608-5.

[Zha+17] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. "Pyramid Scene Parsing Network". In: *CVPR*. 2017.

[Zhe+21] Sixiao Zheng, Jiachen Lu, Hengshuang Zhao, Xiatian Zhu, Zekun Luo, et al. "Rethinking Semantic Segmentation from a Sequence-to-Sequence Perspective with Transformers". In: *CVPR*. 2021.

[Zho+19] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation". In: *IEEE Transactions on Medical Imaging* (2019).

3 Data Efficiency in Neural Network-Based Image Segmentation

In this chapter, we will introduce the concept of data or sample efficiency, as well as how it relates to various factors such as model complexity, problem complexity and sample size. In this thesis, we define data efficiency as a measure of the amount of training data needed for a model to adequately perform its task. While somewhat loosely defined, this term is widely used in the deep learning literature [Nik+22]. In reinforcement learning, the term "sample efficiency" is commonly used to denote the same concept [MMP22]. However, in the context of deep learning models, both the number of samples as well as the number of features in each sample matter to how well a model performs. Therefore, we will prefer the term data efficiency to encompass both the number of samples and features.

Above all, the accuracy of neural networks is governed by three factors: problem complexity, model complexity, and sample size. Each of these factors contributes to accuracy in different ways and they have complex interdependent relationships. We need to understand these relationships to grasp the concept of data efficiency. In this chapter, we will show how the three factors impact model error by examining neural networks as function approximators.

Consider a machine learning scenario where we estimate the 10-year risk of heart disease based on age. Here, some underlying function maps age to heart disease risk, based on a complex physical process that is almost impossible to understand and model. Our only recourse is to collect a set of data points and try to approximate the underlying function. Our goal is to iteratively search for a function that minimizes some criterion that measures the discrepancy between the function and our collected data points.

One can construct an infinite number of functions that fit a set of data points equally well under a selected criterion. Therefore, to derive a single solution, we have to choose a finite class of functions to evaluate. One of the simplest classes of functions we could use is lines — leading to a simple linear regression. A line is defined by two parameters, its slope θ_1 and intersect θ_0:

$$f(x) = \theta_1 x + \theta_0. \tag{3.1}$$

In this equation, $f(x)$ represents our regression model. The objective is to determine the optimal value of $\theta = \{\theta_1, \theta_2\}$ that most accurately represents the collected data pairs (x_i, y_i). This can be achieved by minimizing the squared distance between the line and each data point:

$$\min_{\theta} \sum_{i=1}^{N} |y_i - f(x_i)|^2. \tag{3.2}$$

We will conduct a simulated experiment. We will construct an underlying function as a cosine function with an added Gaussian noise:

$$y(x) = -(\cos(0.01\pi x) - 1) \cdot 0.4 + Z, Z \sim \mathcal{N}(0, 0.1), \tag{3.3}$$

where $\mathcal{N}(0, 0.1)$ is the normal distribution with a mean of zero and a standard deviation of 0.1 from which Z is sampled. We will generate several (x_i, y_i) pairs and apply the previously discussed linear model and criterion to fit a line through these points. In real-world conditions, the underlying function and the nature of the measurement noise remain unknown. However, for this simulation, knowing these details is useful to demonstrate the divergence between the fitted function and the underlying one. Figure 3.1 illustrates the fitted line using varying numbers of data points.

The simulation with only three points clearly demonstrates a mismatch between the approximated and underlying functions. This discrepancy is known as **overfitting**. Overfitting occurs when the model is overly influenced by the specific sample of data, particularly the noise, and fails to generalize to new, unseen examples. As the number of samples increases, the approximated function aligns more closely with the underlying one. However, even with 20 measurements, there is a notable difference between the approximated and underlying functions. The underlying function is a non-linear cosine function. No matter how many data points we use, a line cannot perfectly approximate

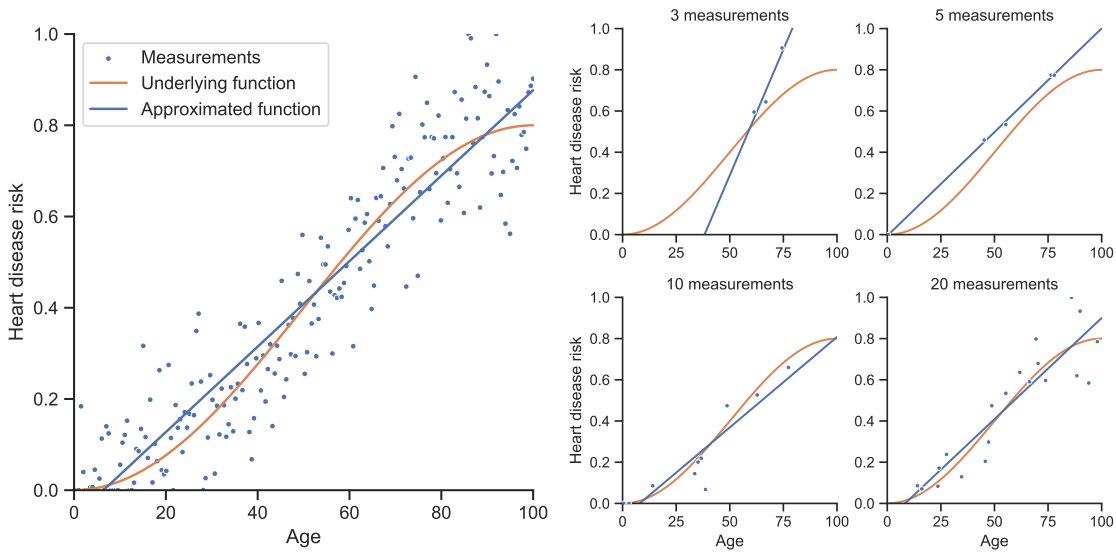


Figure 3.1: A simulated example of heart disease risk prediction using simple linear regression. The plots on the right show how the approximated function depends on the sample size.

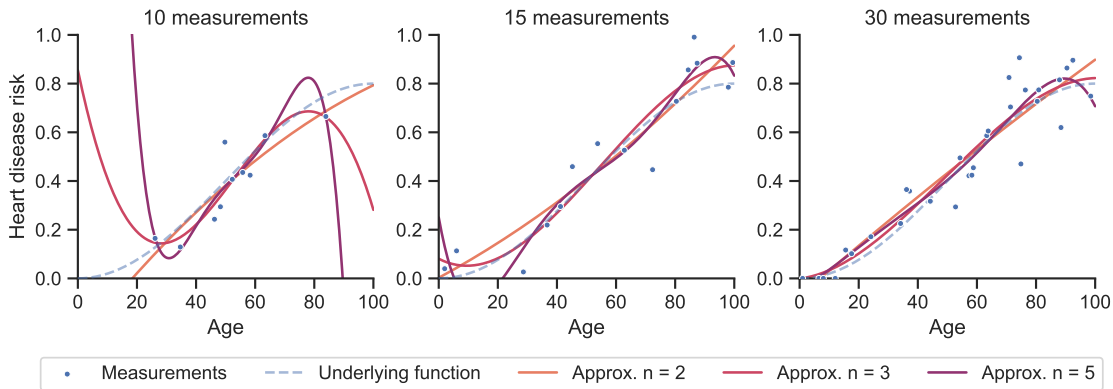


Figure 3.2: Fitting a polygonal function of various degrees on three different sample sizes.

this non-linear function. This is an example of **underfitting**, where the model lacks sufficient parameters to capture the complexity of the underlying function.

To address underfitting, a more complex model is required, meaning we need a broader class of functions for optimization. Our current regression can be viewed as fitting a polynomial function to the dataset. Initially, we have employed polynomials of degree 1 — lines. We can generalize our model to an n-degree polynomial:

$$f(x) = \theta_n x^n + \theta_{n-1} x^{n-1} + \cdots + \theta_1 x + \theta_0. \tag{3.4}$$

To increase our function class, we will choose a larger polynomial degree and again minimize the mean squared distance to obtain θ_n through θ_0. By increasing the degree we are increasing the complexity of the model and the model will be able to represent more complex underlying functions. We can observe what happens when we choose degrees 2, 3, or 5 in Figure 3.2.

In the case with 10 measurements, we observe that the approximated polynomial with $n = 2$ follows the underlying function somewhat well. However, as we increase the degree the function quickly begins to overfit. This experiment demonstrates that the likelihood of overfitting increases with model complexity. Yet, we need complex models to approximate complex functions.

This inherent conflict between overfitting and underfitting is known as the **bias–complexity tradeoff** (alternatively, the bias—variance tradeoff). Let us investigate this problem more formally. We can decompose the error of a deep learning model into two components [SB14]. Consider a trained model f_θ with optimal parameter values θ from a class of functions $\mathcal{F}$. Let there be a function $L(f)$ that measures the error between any model $f \in \mathcal{F}$ and the underlying function we are approximating. The total error of the trained model can be expressed as:

$$L(f_\theta) = \epsilon_{app} + \epsilon_{est} \text{ where: } \epsilon_{app} = \min_{f \in F} L(f),\ \epsilon_{est} = L(f_\theta) - \epsilon_{app}, \tag{3.5}$$

where ϵ_{app} is the **approximation error** and ϵ_{est} is the **estimation error**. The approximation error is the residual error when using the best possible approximator within our chosen class of functions $\mathcal{F}$. In other words, the error that remains given the best possible model parameters. For instance, in the earlier function approximation example with a 1st-degree polynomial, even with infinite data, there would always be some discrepancy in approximation compared to the underlying function. The approximation error depends entirely on the class of functions we use and can only be reduced by adopting a different class, such as polynomials of a higher degree. The remaining estimation error depends on the quality of our parameters θ. It is possible to have a class of functions capable of perfectly approximating the underlying function but still fail to precisely identify the optimal function within that class.

In our previous experiment, we observed that the estimation error is influenced by both the sample size and the complexity of the model. To formally analyze this, we can explore the concept of **sample complexity**. Sample complexity, denoted as $n(\epsilon, \delta)$ for chosen $\epsilon, \delta \in (0, 1)$ measures the number of samples needed for a model f chosen from a finite class of functions $\mathcal{F}$ to be **probably approximately correct**. This means that, across different samplings of data points, there is a probability of $1 - \delta$ that the model is correct within some margin of error ϵ, i.e. $L(f) \leq \epsilon$. Assuming there exists some unknown $f' \in \mathcal{F}$ for which $L(f') = 0$, we can say that the approximation error is zero. Then, it can be shown [SB14] that:

$$n(\epsilon, \delta) \geq \frac{\log(|\mathcal{F}|\delta)}{\epsilon}. \tag{3.6}$$

We can see that an increased function class size (for instance, by choosing a higher-degree polynomial) necessitates a larger sample size n to probably approximately correctly approximate the underlying function. Thus, we conclude that the estimation error can be reduced either by adding more data samples or by decreasing the size of the function class we are optimizing over.

In medical image segmentation, we have an unfortunate stalemate between the three factors governing estimation and approximation errors. The complexity of medical image segmentation demands a large number of parameters, which in turn increases the estimation error. Ideally, this error could be mitigated by adding more data samples, but in the realm of medical imaging, acquiring ample data is often impractical.

To reduce the estimation error, we are left with reducing the size of the function class. We can do so in two ways. The first involves reducing the parameters of the model itself, shrinking the class of functions, and allowing the model to more easily find the optimal function. This usually requires transforming the data such that it can be easily segmented with a less complex model, or using domain-specific knowledge to reframe the problem. The second method involves initializing the model with parameters that are already near-optimal. In this case, the model starts close to the optimal function, needing only to search within a local space around these parameters. This is typically achieved by first optimizing the model on similar data for a related task that involves learning features pertinent to segmentation, followed by further optimization for the specific segmentation task.

In the subsequent sections, we will delve into specific techniques from these two approaches, providing examples of their application in neural network-based medical image segmentation.

3.1 Transfer Learning

Consider the task of segmenting liver tumors in CT scans. Suppose we have a limited dataset of labeled liver tumor regions, but a more extensive collection of labeled liver regions. In such a scenario, the first step would be to train a neural network for liver segmentation. The rationale here is that most of the parameters learned for liver segmentation will be relevant and close to those needed for liver tumor segmentation. Therefore, when we are developing the liver tumor segmentation network, we copy the values of the parameters learned in the liver segmentation network. We can then say that the network was **pre-trained** or **initialized** on a liver segmentation dataset and **fine-tuned** on the liver tumor segmentation dataset.

In this thesis, we refer to this basic technique as "simple transfer learning". There are many other more complex ways of achieving pre-training and fine-tuning. In this section, we will present an overview of some of these methods.

3.1.1 Simple Transfer Learning

Simple transfer learning is a prevalent technique in medical image segmentation. Typically, segmentation neural networks in this field are initialized with weights trained on the ImageNet dataset [Den+09]. ImageNet comprises a collection of natural images, which significantly differ in distribution from medical images. As a result, initializing with ImageNet-trained weights does not usually enhance accuracy but it does make the network converge faster. This is because such initialization 'skips' the phase of learning fundamental image features common across various types of images, including medical ones. Figure 3.3 illustrates this process.

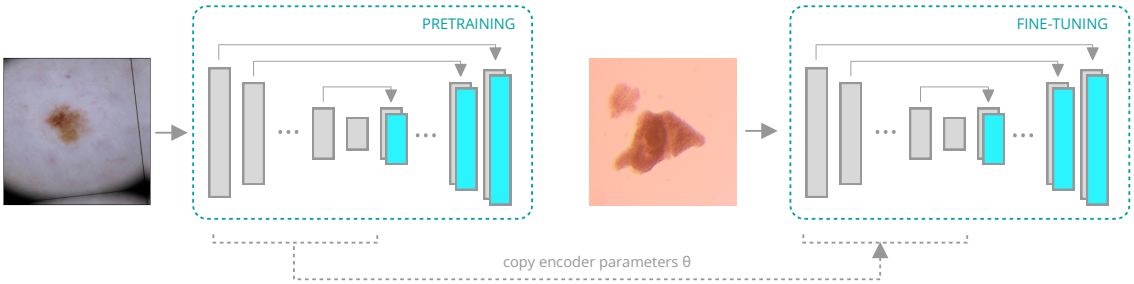


Figure 3.3: An overview of the simple transfer learning procedure. First, a model is pretrained on some related segmentation task. Then, part of the trained model's weights are copied to a new model that is then fine-tuned on the target segmentation task.

Recent datasets, like Wasserthal et al. [Was+23], comprising over 1000 subjects, are increasingly utilized for transfer learning, improving the accuracy of downstream tasks [MON23; Myr+23]. While these approaches can result in impressive performance improvements, their application is constrained to fields where ample, similar data is accessible. In more specialized modalities or segmentation problems, the knowledge learned during pre-training may not be adequate for improving downstream accuracy. A potential solution to bridge this gap involves adapting the pre-trained model to the specific target domain, a topic we'll explore in the following section.

3.1.2 Domain Adaptation

Neural networks often struggle to generalize across different datasets, even within similar domains [TE11]. This is referred to as the "domain shift" problem. Overcoming this problem could lead to an increase in data efficiency since a model trained on some similar large dataset can perform well on a smaller target dataset. Various domain adaptation techniques have been developed to tackle this challenge.

One class of methods aims to explicitly penalize domain shift by including an estimate of domain shift in the loss function. For instance, Liu et al. [Liu+18] first train a network on a source domain with abundant data. This network is then adapted to perform segmentation on a target domain with limited data. During adaptation, the loss function includes a term for the maximum mean discrepancy between the encoded feature vectors of the source and target domains. Minimizing this term forces the encoder to maintain consistent feature representations rather than shifting the distribution of the encoded feature maps.

Another approach involves reducing the network's ability to distinguish between source and target domains, thereby removing information that leads to domain shift. Ganin and Lempitsky [GL15] add a classification head to the network to identify the input's domain. Typically, networks adjust their parameters to minimize loss during each iteration. However, in [GL15], the gradients from the domain classifier are inverted so the network updates its parameters to make it harder to differentiate between the two datasets. This method is illustrated in Figure 3.4.

Domain adaptation has been effectively applied to medical image segmentation, as demonstrated in the work of Bermudez-Chacon et al. [Ber+18]. They utilized a U-Net-like architecture with dual encoder branches, each processing images from a different domain, paired with a shared decoder. Their loss function incorporates a maximum mean discrepancy term, comparing the decoder's final feature maps from inputs of both

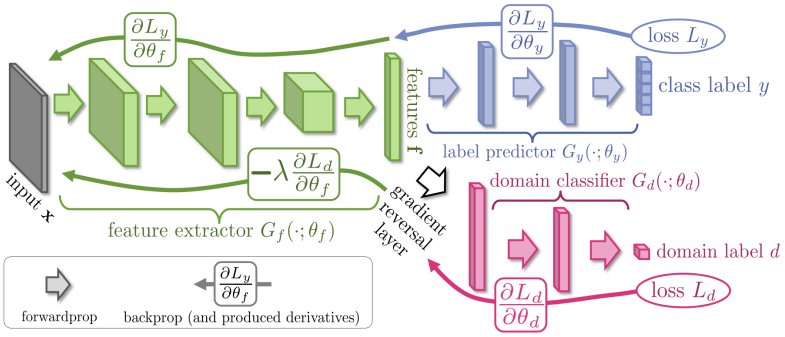


Figure 3.4: A diagram of the domain adaptation approach in Ganin and Lempitsky [GL15]. The gradients of the domain classification head which are applied to the encoder are reversed during backpropagation.

domains. This method was employed for segmenting mitochondria and synapses in microscopic images.

Although domain adaptation techniques enhance transfer learning performance, they generally require labeled data from the source domain. There is growing interest in methods that can utilize unlabeled data to identify general features in images from the target domain regardless of the specific task. We'll explore these methods in the following section.

3.1.3 Semi-Supervised and Self-Supervised Learning

Semi-supervised learning combines both labeled and unlabeled data to train neural networks. One such approach is **self-supervised learning**, where a feature extractor can be trained in a completely unsupervised manner using a large number of images. Then, that feature extractor can be employed via transfer learning to train a network for some specific task such as segmentation. One common strategy for this involves a **pretext task**, where the network is trained on a task for which solutions can be automatically generated from unlabeled data, using automatically generated labels to train the network. A pretext task could be, for instance, solving jigsaw puzzles [NF16]. An image is divided into 3×3 tiles, the tiles are randomly shuffled and fed into the network. The network is tasked with reordering these tiles to reconstruct the original image. Solving this task forces the network to learn to identify salient features such as recognizing a single object across multiple tiles. Such an approach is visualized in Figure 3.5.

Recently, a more common approach to self-supervised learning is **contrastive learning**. This approach focuses on training the encoder to minimize the distance between

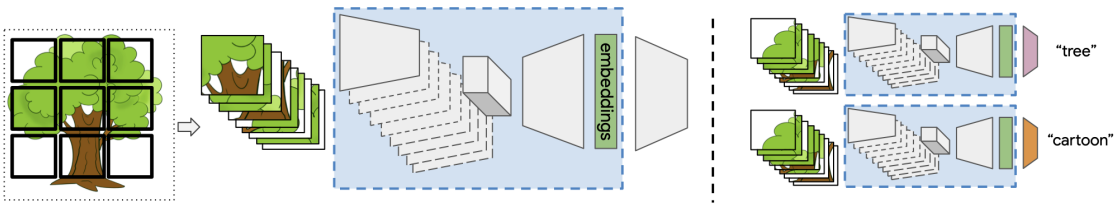


Figure 3.5: A self-supervised learning approach using shuffling image tiles as a pretext task presented by Carr et al. [Car+21]. The trained feature encoder learns to extract relevant features and its parameters are transferred to a model trained to perform the downstream task.

feature vectors of similar (or positive) examples while maximizing the distance between dissimilar (or negative) examples. Positive examples are generated in an unsupervised manner, often by applying random augmentations to an image to create two variants. The goal is for the feature vectors of these variants to be closely aligned. This teaches the network to recognize similar objects in a way that is invariant to the augmentations. Notable implementations of contrastive learning include SimCLR [Che+20] and MoCo [He+19].

3.2 Synthetic Data

The data efficiency methods we've discussed so far revolve around pre-training networks with unlabeled or similar data. An alternative strategy involves generating synthetic data that mimics the target distribution. There are various techniques for this, each offering different levels of complexity and generation quality. A straightforward and commonly used method for expanding dataset size is **data augmentation**.

Data augmentation refers to the application of random transformations to input images, thereby artificially increasing their diversity. These transformations can be simple ones such as adjusting brightness or contrast, flipping the image horizontally, or more intricate, involving complex random deformations of the image. Generally, the choice of transformations is governed by domain knowledge — the network is forced to disregard transformations that we know it should be invariant to. Data augmentation is a standard practice in segmentation models, often employed alongside other data efficiency techniques. However, its implementation typically relies on simple deformation or transformation methods, which limits its capacity to create truly diverse samples.

Recently, the field of deep learning-based image generation, especially generative adversarial networks and diffusion models, has sparked a large interest in generating synthetic medical images. **Generative adversarial networks** [Goo+14] consist of two neural networks, a generator and a discriminator, that are trained simultaneously. The generator creates synthetic images intended to be indistinguishable from real images, while the discriminator learns to differentiate between the two. The process continues until the generator produces images so convincing that the discriminator cannot easily distinguish them from real images. This technique has been particularly effective in generating realistic medical images for data augmentation [Shi+18].

Diffusion models [HJA20], a more recent development in the field of generative models, offer an alternative approach to image generation. These models, inspired by the physical process of diffusion, work by gradually adding noise to an image and then learning to reverse this process. The result is the generation of new images by reversing the noise addition process from a randomly sampled noise distribution. Diffusion models have been shown to generate high-quality images that can be particularly useful in medical imaging contexts [Kha+23].

3.3 Regularization

Aside from getting new data, the easiest way to reduce the estimation error is by limiting the size of the function class our model optimizes. In a traditional machine learning approach, this would be done by hand-selecting a class of functions with a lower number of parameters and then training the model. In deep learning, however, this approach is usually replaced with **regularization**, where the parameters themselves are manipulated

to simplify the model.[1] Instead of selecting the number of parameters, we can penalize the network for the complexity and size of the parameters. In other words, the loss function between a predicted model output y_{pred} and a ground-truth label y has the form:

$$L(\theta; y_{pred}, y) = L_{seg}(\theta; y_{pred}, y) + \alpha\Omega(\theta), \tag{3.7}$$

where θ are the model parameters, L_{seg} is the segmentation loss function, and $\Omega(\theta)$ is a parameter norm penalty, i.e. a measure of the complexity of the model [GBC16]. As we increase α, the strength of the regularization increases and the loss is more impacted by the complexity of the model. As the network is being trained, it will seek to both reduce $\Omega(\theta)$ as well as L_{seg}, leading to a more constrained model.

A commonly used parameter norm penalty is the L^2 regularization, also known as **weight decay** in deep learning contexts. Assuming θ consists of a set of weights w, $|w| = n$ and biases b, the L^2 norm is defined as $\Omega(w) = \sqrt{(w_1^2 + w_2^2 + \cdots + w_n^2)}$. In other words, the network is exponentially penalized for moving weights away from zero. This regularisation, in effect, forces the network to use fewer parameters.

Thus, a deep learning network can learn to use fewer parameters without necessarily changing the architecture of the network itself. However, what remains unsolved is how to transform the data such that the network can solve the problem with fewer parameters. We will discuss that in the next chapter, where we present a different approach to achieving data efficiency.

3.4 Conclusion

As we have seen, there are several approaches to increasing the data efficiency of neural network-based segmentation of medical images. This thesis focuses on using regularisation by transforming the input data in such a way as to allow the network to use fewer parameters to achieve its segmentation task. We do so by combining predictions of neural networks with more traditional approaches from image processing and data augmentation. In the next chapter, we will introduce this idea and some specific applications.

[1] Often, any method of reducing the estimation error of the model is broadly termed as regularization.

References

[Ber+18] Roger Bermudez-Chacon, Pablo Marquez-Neila, Mathieu Salzmann, and Pascal Fua. "A Domain-Adaptive Two-Stream U-Net for Electron Microscopy Image Segmentation". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. Washington, DC: IEEE, Apr. 2018, pp. 400–404. ISBN: 978-1-5386-3636-7. DOI: 10.1109/ISBI.2018.8363602. (Visited on 11/28/2023).

[Car+21] Andrew N Carr, Quentin Berthet, Mathieu Blondel, Olivier Teboul, and Neil Zeghidour. *Shuffle to Learn: Self-supervised Learning from Permutations via Differentiable Ranking*. 2021.

[Che+20] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. "A Simple Framework for Contrastive Learning of Visual Representations". In: *arXiv:2002.05709 [cs, stat]* (June 2020). arXiv: 2002.05709 [cs, stat]. (Visited on 02/10/2022).

[Den+09] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, et al. "ImageNet: A Large-Scale Hierarchical Image Database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. Miami, FL: IEEE, June 2009, pp. 248–255. ISBN: 978-1-4244-3992-8. DOI: 10.1109/CVPR.2009.5206848. (Visited on 12/01/2023).

[GBC16] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. Adaptive Computation and Machine Learning. Cambridge, Massachusetts: The MIT Press, 2016. ISBN: 978-0-262-03561-3.

[GL15] Yaroslav Ganin and Victor Lempitsky. "Unsupervised Domain Adaptation by Backpropagation". In: *Proceedings of the 32nd International Conference on International Conference on Machine Learning - Volume 37*. ICML'15. Lille, France: JMLR.org, 2015, pp. 1180–1189.

[Goo+14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, et al. *Generative Adversarial Networks*. June 2014. arXiv: 1406.2661 [cs, stat]. (Visited on 12/01/2023).

[He+19] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. "Momentum Contrast for Unsupervised Visual Representation Learning". In: *arXiv preprint arXiv:1911.05722* (2019). arXiv: 1911.05722.

[HJA20] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising Diffusion Probabilistic Models". In: *arXiv preprint arxiv:2006.11239* (2020). arXiv: 2006.11239.

[Kha+23] Firas Khader, Gustav Müller-Franzes, Soroosh Tayebi Arasteh, Tianyu Han, Christoph Haarburger, et al. "Denoising Diffusion Probabilistic Models for 3D Medical Image Generation". In: *Scientific Reports* 13.1 (May 2023), p. 7303. ISSN: 2045-2322. DOI: 10.1038/s41598-023-34341-2. (Visited on 12/01/2023).

[Liu+18] Lihang Liu, Weiyao Lin, Lisheng Wu, Yong Yu, and Michael Ying Yang. *Unsupervised Deep Domain Adaptation for Pedestrian Detection*. Feb. 2018. arXiv: 1802.03269 [cs]. (Visited on 11/28/2023).

[MMP22] Vincent Mai, Kaustubh Mani, and Liam Paull. "Sample Efficient Deep Reinforcement Learning via Uncertainty Estimation". In: *International Conference on Learning Representations*. 2022.

[MON23] MONAI Consortium. *MONAI: Medical Open Network for AI*. Zenodo. Oct. 2023. DOI: 10.5281/ZENODO.4323058. (Visited on 11/28/2023).

[Myr+23] Andriy Myronenko, Dong Yang, Yufan He, and Daguang Xu. *Automated 3D Segmentation of Kidneys and Tumors in MICCAI KiTS 2023 Challenge*. Oct. 2023. arXiv: 2310.04110 [cs]. (Visited on 11/28/2023).

[NF16] Mehdi Noroozi and Paolo Favaro. "Unsupervised Learning of Visual Representations by Solving Jigsaw Puzzles". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Vol. 9910. Cham: Springer International Publishing, 2016, pp. 69–84. ISBN: 978-3-319-46465-7 978-3-319-46466-4. DOI: 10.1007/978-3-319-46466-4_5. (Visited on 02/16/2022).

[Nik+22] Evangelos-Marios Nikolados, Arin Wongprommoon, Oisin Mac Aodha, Guillaume Cambray, and Diego A. Oyarzún. "Accuracy and Data Efficiency in Deep Learning Models of Protein Expression". In: *Nature Communications* 13.1 (Dec. 2022), p. 7755. ISSN: 2041-1723. DOI: 10.1038/s41467-022-34902-5. (Visited on 06/17/2024).

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. 1st ed. Cambridge University Press, May 2014. ISBN: 978-1-107-05713-5 978-1-107-29801-9. DOI: 10.1017/CBO9781107298019. (Visited on 12/01/2023).

[Shi+18] Hoo-Chang Shin, Neil A. Tenenholtz, Jameson K. Rogers, Christopher G. Schwarz, Matthew L. Senjem, et al. "Medical Image Synthesis for Data Augmentation and Anonymization Using Generative Adversarial Networks". In: *Simulation and Synthesis in Medical Imaging*. Ed. by Ali Gooya, Orcun Goksel, Ipek Oguz, and Ninon Burgos. Vol. 11037. Cham: Springer International Publishing, 2018, pp. 1–11. ISBN: 978-3-030-00535-1 978-3-030-00536-8. DOI: 10.1007/978-3-030-00536-8_1. (Visited on 12/01/2023).

[TE11] Antonio Torralba and Alexei A. Efros. "Unbiased Look at Dataset Bias". In: *CVPR 2011*. Colorado Springs, CO, USA: IEEE, June 2011, pp. 1521–1528. ISBN: 978-1-4577-0394-2. DOI: 10.1109/CVPR.2011.5995347. (Visited on 11/28/2023).

[Was+23] Jakob Wasserthal, Hanns-Christian Breit, Manfred T. Meyer, Maurice Pradella, Daniel Hinck, et al. "TotalSegmentator: Robust Segmentation of 104 Anatomic Structures in CT Images". In: *Radiology: Artificial Intelligence* 5.5 (Sept. 2023), e230024. ISSN: 2638-6100. DOI: 10.1148/ryai.230024. (Visited on 11/27/2023).

4 Data Efficiency via Model-Driven Preprocessing

In this chapter, we will propose a way of achieving data efficiency in neural network-based medical image segmentation different from the ones described in the previous chapter. Using this new proposed approach, we will develop a method using the polar transformation to preprocess medical images and thus improve the data efficiency of segmentation methods.

Our approach centers on preprocessing the input data using domain knowledge and traditional image-processing techniques to make the image more easily segmentable. However, instead of manually selecting transformation parameters, we will model the parameters as functions of the image, and train a neural network to estimate optimal transformation parameters. As discussed in the previous chapter, regularization enables a network to limit its number of parameters to what is necessary for the given problem's complexity. Through strategic image transformation, we aim to simplify the complexity of the segmentation task, thus allowing for efficient segmentation with fewer parameters.

We will first present some motivation to show how it is possible to transform data to be easier to model. This is followed by a detailed outline of our method, as well as an empirical study, demonstrating the effectiveness of our approach across various medical image segmentation tasks. We then show an improvement of the method to support multiple segmenting objects and another empirical study using the improved approach.

4.1 Motivation: Using The Polar Transform for Preprocessing

Our approach is centered on discovering transformations of data that enable neural networks to approximate decision boundaries with fewer parameters. This strategy is grounded in a particular interpretation of neural networks. Consider a network tasked with classifying a given image region $I(x, y)$ as class "0" or "1". All conceivable image regions $I(x, y)$ form part of a d-dimensional space, known as the input space. Here, d represents the total number of pixels in the region multiplied by the number of channels in each pixel. The neural network, through iterative optimization, seeks to transform the input space into some other n-dimensional in such a way that it becomes linearly separable by an n-dimensional hyperplane into two distinct classes. This process and its outcome are depicted in Figure 4.1.

To find the transformation, the network needs a sufficient number of parameters. More parameters allow for more complex transformations. For instance, to accurately approximate a circular decision boundary, the network needs at least three neurons in its second layer. We exemplify this in Figure 4.2.

The goal of data efficiency is to find a decision boundary given fewer data samples. As discussed in the previous chapter, this necessitates narrowing the search space of possible decision boundaries, or, in other words, reducing the number of neural network parameters. In the current and subsequent chapters, our focus will be on devising

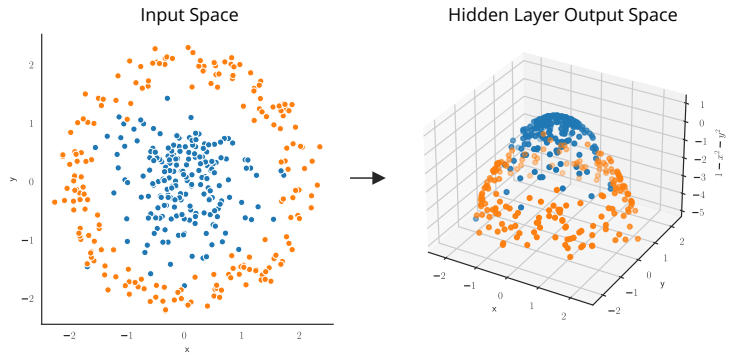


Figure 4.1: An example of how a 2-class classification neural network bends an input space (left) such that it is linearly separable with a hyperplane.

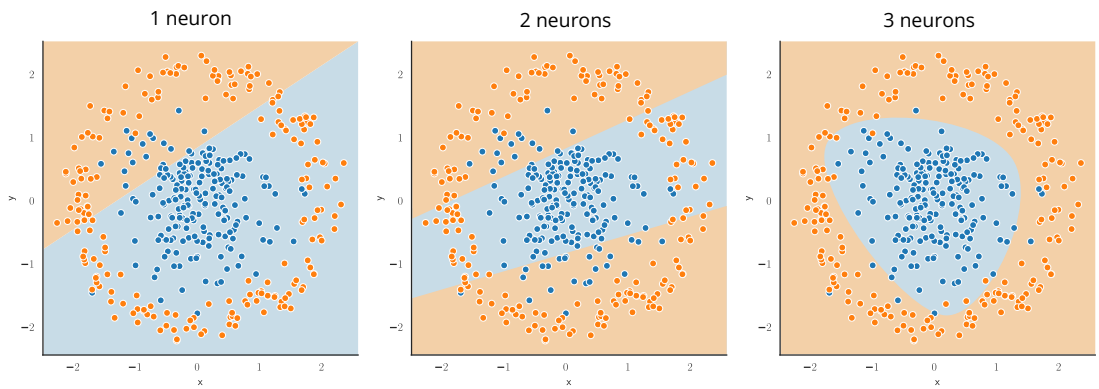


Figure 4.2: The decision boundary (shown as the background color) according to a fully connected neural network with three layers where the second layer has one, two, or three neurons.

methods to pre-process the data into an intermediary form, denoted as $\mathcal{X}_{pre}$, which is more easily separable than the original input space $\mathcal{X}$. By transforming data into $\mathcal{X}_{pre}$, the network requires fewer parameters as it has a smaller search space to find X'. This transformation process is referred to as $\mathcal{T}$.

A commonly used example of $\mathcal{T}$ is the polar transform. In the two-dimensional case, the polar transform $\mathcal{T}(X_i; C)$, defined by a chosen origin $C = [c_0, c_1]$, converts an input vector $X = [x_0, x_1]$ into a vector $\mathcal{T}_C(X) = [\theta, r]$. The transformation is as follows:

$$\begin{aligned} r =& \sqrt{(x_0 - c_0)^2 + (x_1 - c_1)^2}, \\ \theta =& \text{atan2}(x_1 - c_1, x_0 - c_0) \end{aligned} \tag{4.1}$$

where $\arctan(x, y)$ is the two-argument arcus tangent function, defined as the angle between the vector $(x, y) \in \mathbb{R}^2 \backslash \{0\}$ and the positive x-axis.

The polar transform effectively converts a circular decision boundary into a linear one. This transformation enables linear separability in our example dataset, which then requires only a single neuron in the second layer for effective classification. This is demonstrated in Figure 4.3.

However, the key to the polar transform's success in facilitating separability lies in the correct selection of the polar origin. When the origin is shifted away from the center of class "0", the decision boundary in the polar space ceases to be linear. This effect

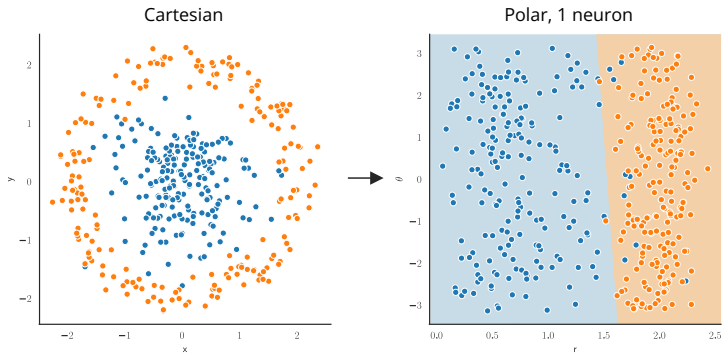


Figure 4.3: An example circular dataset transformed using the polar transform. The background color shows the decision boundary of a fully connected network with one neuron in the second layer.

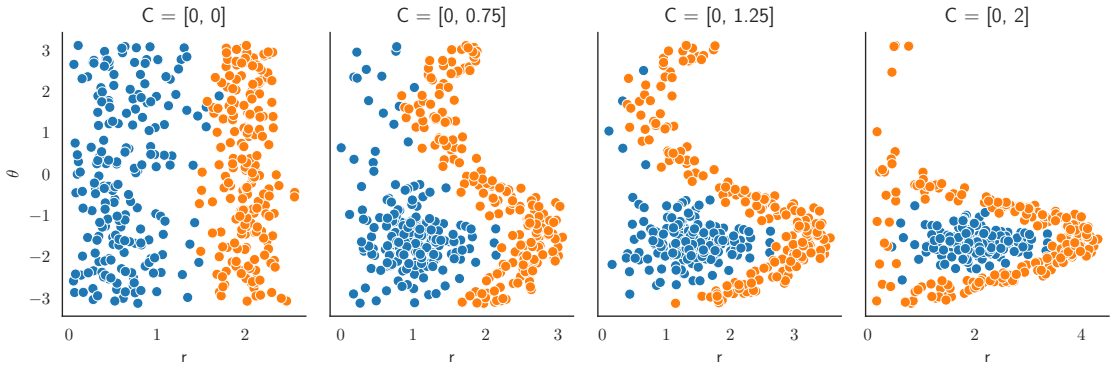


Figure 4.4: The polar transformation of the simulated data using different polar origins (C).

of origin displacement on the linearity of the decision boundary in the polar space is illustrated in Figure 4.4.

Building on the example of the polar transform, we propose a methodology where key transformation parameters, like the polar origin, are dynamically estimated using a neural network. We call this approach model-driven preprocessing, specifically tailored to enhance data efficiency in medical image segmentation. While the above examples have focused on a classification problem, if we frame a segmentation model as predicting a decision boundary between two regions of an image, we can make an argument that reducing region boundary complexity also reduces the required hypothesis space of a segmentation model.

4.2 Model-driven Preprocessing

The concept illustrated with the polar transform can be effectively applied to image segmentation. Imagine an image where pixels belonging to one class fall mostly within a circular decision boundary, with the background occupying the space outside this circle. A segmentation model tasked with this scenario needs to approximate a circular boundary to segment the image accurately. By applying the polar transform, this circular boundary is converted into a linear one, significantly simplifying the task and allowing for segmentation with a less complex model.

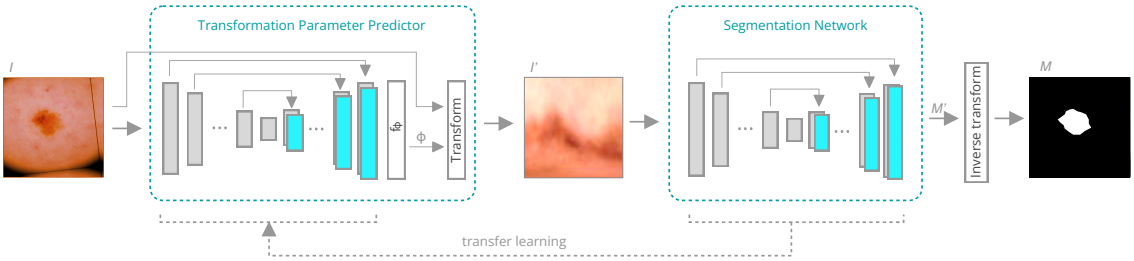


Figure 4.5: A visual summary of the model-driven preprocessing approach. First, the transformation parameter predictor network predicts a segmentation-like map, from which the transformation parameters are calculated using a transformation parameter function. Then, the input image is transformed and the transformed representation is segmented by a segmentation neural network. The final segmentation map is obtained by inverting the transformation.

We propose a general formulation of this approach, wherein images are transformed such that they can be made linearly separable using simpler transformations. We do so by carefully selecting transformations derived from traditional image processing methods. The success of these transformations hinges on the correct selection of parameters. To address this, we employ neural networks trained specifically to predict these optimal transformation parameters. The preprocessed image is then segmented using a regular neural network. Importantly, since the preprocessed image is inherently simpler to segment, the segmentation network can be designed with fewer parameters compared to a network segmenting unprocessed images. This is, in essence, a model-driven approach to preprocessing the image. A visual summary of this approach is shown in Figure 4.5. In the remainder of this section, we will describe this approach in more detail.

To preprocess the image, we select a transformation function $\mathcal{T}_\phi(I)$, where ϕ represents the transformation parameters, and I is the original image. This transformation yields a new image, I', which can be segmented with fewer parameters than the original.

The transformation function $\mathcal{T}$ can vary, encompassing affine transformations, deformation fields, thresholding techniques, and, as previously mentioned, the polar transform. The choice of $\mathcal{T}$ is contingent upon the specific segmentation task. For instance, the polar transform is advantageous for segmenting elliptical objects. When segmenting small objects, we might opt for an affine transformation that scales each object to a uniform scale. The design of $\mathcal{T}$ should be informed by domain knowledge and validated through empirical testing. An important requirement is that $\mathcal{T}$ must be invertible, allowing the resulting segmentation mask to be mapped back to the original image space. The segmentation process on the transformed image can be represented as:

$$M(I) = (\mathcal{T}_\phi^{-1} \circ F_{seg} \circ \mathcal{T}_\phi)(I), \tag{4.2}$$

where $M(I)$ is the resulting segmentation mask for image I, F_{seg} is the segmentation neural network.

The effectiveness of preprocessing is highly dependent on the appropriate choice of ϕ, as illustrated with the polar transform. It is also important to note that ϕ is specific to each image: the origin, scale, location, and distribution of objects differ across images. Acknowledging this, we propose training a model to estimate the optimal ϕ for a given image:

$$\phi(I) = F_\mathcal{T}(I; \theta_\mathcal{T}), \tag{4.3}$$

where $F_{\mathcal{T}}$ is a neural network with parameters $\theta_{\mathcal{T}}$ trained to estimate the correct parameters for $\mathcal{T}$ given an image I. In our approach, we use various types of neural networks as $F_{\mathcal{T}}$, though it is also possible to use more traditional approaches.

More precisely, $F_{\mathcal{T}}$ is structured similarly to a regular segmentation network, producing a segmentation mask $M_\phi(I)$ in its second-to-last layer. The final layer, which is a non-trainable function $\phi(I) = f_\phi(M_\phi(I))$, produces transformation parameters given a predicted segmentation map. $F_{\mathcal{T}}$ can then be expressed as:

$$F_{\mathcal{T}}(I;\theta_{\mathcal{T}}) = f_\phi(F_{\mathcal{T}seg}(I;\theta_{\mathcal{T}})), \tag{4.4}$$

where $F_{\mathcal{T}seg}$ is the segmentation sub-network producing $M_\phi(I)$. $F_{\mathcal{T}seg}$ can be trained using segmentation-like loss functions, where M_ϕ can be compared to a ground truth segmentation mask $M_{gt}(I)$.

The function f_ϕ is hand-constructed using knowledge of the transformation. For instance, we know that for the polar transform, the optimal origin lies in the center of the object. Therefore, we may use the centroid of the ground truth segmentation labels as f_ϕ. Other examples of f_ϕ for various transformations include:

- **CT Thresholding**: The transformation parameters are the maximum and minimum Hounsfield units representing a specific tissue. The function f_ϕ could use the minimum and maximum values within the predicted segmentation mask to determine these parameters.
- **Rescaling**: For rescaling transformations, the parameters are the x and y scaling factors. We can determine these factors using the predicted segmentation mask to ensure that the object attains a predefined fixed width and height.
- **Translation**: The translation parameters, in terms of x and y shifts, can be calculated by measuring the distance between the object's center and a desired reference point.

It is important to note that $M_\phi(I)$ does not have to be a conventional segmentation mask. Depending on the requirements for predicting optimal transformation parameters, we can create pseudo-labels that are more suitable for the task. For example, in cases where transformation parameters are contingent on an object's centroid, $M_\phi(I)$ could be modeled as a heatmap with its focus on the object's central point. This approach enables the network to learn to predict the centroids of objects rather than their boundaries. Consequently, f_ϕ can determine the object's centroid by locating the pixel with the highest intensity in $M_\phi(I)$.

Using this approach, data efficiency is achieved by reducing the total function class size when compared to a standard segmentation neural network. We do so by using two smaller neural networks instead of one large one. Since the decision boundaries of the preprocessed images are simplified, the segmentation network needs fewer parameters to model them. Transformation parameters often have a much smaller dimensionality than the segmentation mask, and therefore the network predicting transformation parameters can naturally use much fewer parameters.

The core idea of our approach is that $\mathcal{T}$ will simplify the segmentation problem such that a regularized segmentation network F_{seg} will be able to perform the segmentation using fewer parameters. To effectively implement this, both $F_{\mathcal{T}}$ (the network predicting transformation parameters) and F_{seg} (the segmentation network) require specialized training techniques, which will be elaborated in the forthcoming section.

4.2.1 Training Model-Driven Preprocessing Networks

As mentioned earlier, using the approach of model-driven preprocessing requires specific neural network training techniques. In our approach, two separately trained neural networks are used: the transformation parameter predictor $F_{\mathcal{T}}$ and the segmentation network F_{seg}.

Training the Transformation Parameter Predictor

$F_{\mathcal{T}}$ is designed to process original, untransformed images and output the corresponding transformation parameters, $\phi(I)$. As previously discussed, the penultimate layer of $F_{\mathcal{T}}$ generates a segmentation-like mask $M_\phi(I)$. The training data for $F_{\mathcal{T}}$ comprises pairs of images and their corresponding ground truth masks, represented as (I, M_{gt}). Given a predicted segmentation mask $M_\phi = F_{\mathcal{T}seg}(I)$ and the transformation parameter function $f_\phi(M(I))$, the transformation parameters loss $L_{\mathcal{T}}$ can be calculated as:

$$L_{\mathcal{T}}(I, M_\phi, M_{gt}) = \alpha L_\phi(f_\phi(M), f_\phi(M_{gt})) + \beta L_M(M_\phi, M_{gt}), \tag{4.5}$$

where L_ϕ measures the error in predicting the transformation parameters, while L_M is a segmentation-like loss that measures the error in predicting the segmentation mask used to calculate the transformation parameters. The coefficients α and β, which balance these two terms, are determined empirically.

Training the Segmentation Network

The segmentation network is trained using ground-truth transformation parameters (ϕ) to preprocess the input image. For every pair of image and ground truth segmentation label (I, M_{gt}), the ground truth transformation parameters are derived using:

$$\phi = f_\phi(M_{gt}) + z_\phi, \tag{4.6}$$

where f_ϕ is the transformation parameter function, identical to the one used in the parameter predictor network. The term z_ϕ represents a random noise vector matching ϕ's dimensions. This noise serves as a form of augmentation, enhancing the segmentation network's resilience to less-than-ideal ϕ predictions.

Given ϕ, the input to the segmentation network is constructed as $I' = \mathcal{T}_\phi(I)$. The network's final layer applies an inverse transformation to the output segmentation map, $\mathcal{T}_\phi^{-1}(M)$. To train the network, we use a standard segmentation loss $L_{seg}(M, M_{gt})$.

Transfer Learning

Given that the parameter predictor mirrors the structure of a segmentation network, we can utilize the same encoder and decoder in both networks, leveraging various forms of transfer learning. This approach offers several advantages:

1. Both networks employ a standard encoder, enabling the initialization of these networks with weights pre-trained on extensive medical or natural image datasets.

2. The development process can start by training a standard baseline segmentation model on untransformed data. Subsequently, the learned weights from this model can be transferred to both the segmentation and transformation parameter predictor networks. In this case, the parameter predictor already knows how to localize the objects, thereby enhancing its ability to predict optimal transformation parameters.

3. The transformation parameter predictor network's main role is to robustly transform input images. Domain shifts or unrealistic data, which are critical concerns for the segmentation network, are less problematic for the parameter predictor. This leniency allows for the use of extensive augmentation during training, further improving the network's robustness and adaptability.

Having laid out the theoretical framework of this approach, the remainder of this chapter will detail a specific application of this methodology, demonstrated through an empirical study across various medical imaging datasets. This study aims to showcase the practical effectiveness and adaptability of the model-driven preprocessing approach in real-world scenarios.

4.2.2 Related Work

Several image segmentation methods were proposed that utilize polar coordinates. Liu et al. [Liu+19] proposed an approach they call Cartesian-polar dual-domain network (DDNet) to perform optic disc and cup segmentation in retinal fundus images. The neural network contains two encoding branches, one for a Cartesian input image and another for the polar transformation of the same input image. The predictions are fused into a single feature vector which is then decoded into a final segmentation. Salehinejad et al. [Sal+18] used the polar transformation as a way to augment training data by transforming each input image into multiple polar images at various polar origins, thus increasing the number of training data. Kim et al. [Kim+20] proposed a convolutional neural network layer for images in polar coordinates to achieve rotational invariance. Their cylindrical convolution layer uses cylindrically sliding windows to perform a convolution. Kim et al. [Kim+18] proposed a user-guided segmentation method where an expert selects the point used as the polar origin. The transformed image is then segmented using a convolutional neural network (CNN). Esteves et al. [Est+18] proposed a polar transformer network for image classification. Note that "transformer network" here refers to spatial transformer networks [Jad+16] and not attention-based networks commonly called transformers. The network consist of a polar origin predictor and a neural network that predicts a heatmap. The centroid of the heatmap is then used as the origin for a polar transformation of the input image. The polar image is classified using a CNN. This approach is most similar to our proposed method, however, their approach focuses on image classification, not segmentation. In addition, our approach uses two separate neural networks, which allows us to better focus on improving data efficiency.

While there are proposed methods that combine polar transformation with neural networks, most of them solve classification tasks. Some medical image segmentation methods use the polar transformation as a preprocessing step, however, the way they obtain the origin of the polar transformation is usually based on heuristics. To our knowledge, there is currently no work that explores using the polar transformation with a dynamic polar origin as a preprocessing step for semantic segmentation in a variety of medical image datasets.

4.3 Model-Driven Polar Transform Using Centerpoint Prediction

Earlier in this chapter, we discussed how the polar transform can simplify complex decision boundaries, making segmentation tasks more manageable. Drawing from this insight, this section introduces a practical implementation of our model-driven preprocessing approach, utilizing the polar transform. The approach involves two networks:

a transformation parameter predictor and a segmentation network. In this specific implementation, the parameter predictor is designed to determine the polar origin of an image, functioning as a centerpoint predictor network. Its objective is to identify the center of a singular object within the image. We will explore two distinct designs for this centerpoint predictor network:

1. **Segmentation-like Network**: This design focuses on identifying the centroid of a segmentation mask. It operates like traditional segmentation networks but is specifically optimized to pinpoint the center of an object.

2. **Heatmap Prediction Network**: Alternatively, this network predicts the centerpoint directly as a heatmap. This approach is more direct, as it aims to localize the object's center without necessarily delineating its entire boundary.

Once the centerpoint is determined, the image undergoes a transformation using the predicted polar origin. The transformed image is then fed into a segmentation network that has been trained on images processed with the polar transform. This setup aims to demonstrate the effectiveness of preprocessing in simplifying the segmentation task, potentially allowing for more accurate and efficient segmentation with neural networks. The subsequent parts of this section will delve into the specifics of these network designs and their performance in practical scenarios.

Our method is evaluated on the tasks of polyp segmentation, liver segmentation, skin lesion segmentation, and epicardial adipose tissue (EAT) segmentation. The proposed method can be used as a pre-processing step for existing neural network architectures, so we evaluate the methods using common neural network architectures for medical image segmentation including U-Net [RFB15b], U-Net++ [Zho+19] with a ResNet [He+16] encoder, and DeepLabV3+ [Che+18] with a ResNet [He+16] encoder. Evaluation of our approach as a pre-processing step shows that it improves segmentation performance across different datasets and neural network architectures while making the networks more robust to small dataset sample sizes.

4.3.1 Methodology

The polar transform's potential for simplifying the segmentation of images with elliptical borders was discussed earlier in this chapter. Transforming such images into polar coordinates effectively turns a perfect circle from Cartesian coordinates into a straight line, significantly simplifying the decision boundary. This simplification means that a less complex model can accurately predict borders in polar coordinates. As illustrated in Figure 4.6, even for complex shapes, transforming roughly elliptical objects into polar coordinates can reduce the complexity required for accurate segmentation. Moreover, by using the object's center as the polar origin, the approach standardizes the location and border distances in each training example. This standardization allows the model to focus on learning the border distance from the origin at various angles, obviating the need for object localization learning.

Given a polar origin (c_x, c_y) of a Cartesian image $I(x, y)$ of resolution $H \times W$, we obtain each point (r, ϑ) of the polar transformed image $I'(r, \vartheta)$ as:

$$\begin{aligned} r &= \frac{H}{2\pi} \cdot \text{atan2}(x - c_x, y - c_y)[\cdot 180/\pi], \\ \vartheta &= \frac{W}{\sqrt{(W/2)^2 + (H/2)^2}} \cdot \sqrt{(x - c_x)^2 + (y - c_y)^2}, \end{aligned} \tag{4.7}$$

where *arctan* is the 2-argument arctangent function.

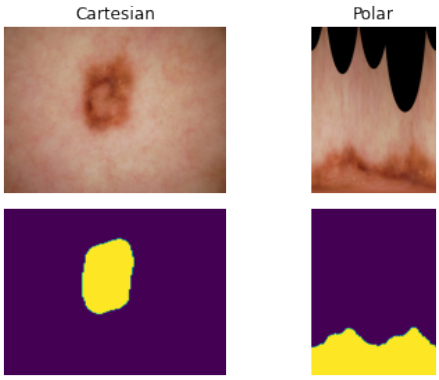


Figure 4.6: An example image and label from the lesion dataset and their corresponding polar transformation. [Ben+21]

All of the proposed methods rely on training a neural network model to segment polar images. To train on polar images, the input images need to be transformed using a polar origin which is near the center of the segmented object. The correct origin is not known ahead of time, so a prerequisite for predictions on polar images is a method to determine the correct polar origin. We propose and evaluate two different methods for automatically obtaining the polar origin: (1) estimation via a segmentation trained on non-polar images and (2) training a center-point predictor that predicts heatmaps from input images. This section describes these methods, as well as methods to train the final segmentation model on the polar images.

In each of our approaches, the final segmentation is done using a neural network trained on polar transformations of the input images. In the rest of this section, we refer to this network as the polar network. In all of the described approaches, the polar transformation is not part of the network architecture itself, but happens as a preprocessing step for the polar network. To transform each input image, the polar origin is determined as the center of mass of the ground truth label for that image. The center of mass of an image $I(x, y)$ is calculated by first calculating the spatial image moments matrix M, where the entry of the matrix at row i and column j is calculated as:

$$M_{ij} = \sum_{x,y} I(x, y) \cdot x^i \cdot y^j. \tag{4.8}$$

The center of mass (c_x, c_y) of the image can then be calculated as:

$$\begin{aligned} c_x &= M_{10} / M_{00}, \\ c_y &= M_{01} / M_{00} \end{aligned}. \tag{4.9}$$

Finally, to increase the model's robustness to suboptimal center point predictions, we augment the calculated center for training images [Est+18]. Each training image has a 30% chance of varying the center's x and y coordinates by a random value in the range $(-0.05S, 0.05S)$, where S is the smallest resolution of the image, i.e. $S = min(W, H)$.

4.3.2 Centerpoint prediction

Once the polar network is trained, inference can be done by transforming an input image to polar coordinates. The polar network requires choosing a center that is close to the center of mass of the segmented object. Because a future input image is unlabeled, the

correct center needs to be inferred from the image. We propose two ways to accomplish this, described in this section.

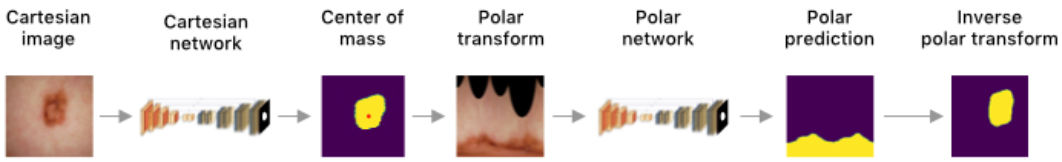


Figure 4.7: A diagram of the approach (A) of predicting polar origins from a Cartesian network. The first network performs an initial segmentation, which is then used to extract a polar origin for the polar transformation. The method does not rely on any specific neural network architecture. The Polar and Cartesian network can be any neural network that takes an input image and produces a binary segmentation mask as output. The red point shows the extracted polar origin. The Polar network is trained on polar image transformations. The polar transformation is not part of the network itself, but happens as a preprocessing step for the Polar network. [Ben+21]

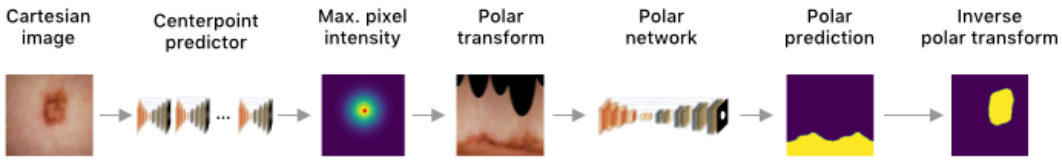


Figure 4.8: A diagram of the approach (B) of using a centerpoint prediction network. The first network can be any neural network that predicts a heatmap from an input image, which is then used to extract the polar origin, shown as a red point. The Polar network can be any semantic segmentation neural network that produces a binary mask output from an input image. The Polar network is trained on polar image transformations. The polar transformation is not part of the network itself, but happens as a preprocessing step for the Polar network. [Ben+21]

(A) Training the same neural network on cartesian and polar images

Our first approach is training the same neural network on cartesian and polar images. A summary of this approach is presented in Figure 4.7. With this approach, the inference is done by first feeding the original Cartesian input images into a neural network used for segmentation. We refer to this network as the Cartesian network. For an input image, the polar origin is calculated as the center of mass of the Cartesian network's prediction for that input image, using (4.9). This polar origin is used to transform the original input image to polar coordinates, and the transformed image is fed to the polar network. The output of the polar network is transformed back to Cartesian coordinates to obtain a final segmentation. We assume identical architectures for both the Cartesian and polar networks. This makes applying this framework to existing architectures very straightforward, as it does not require designing new neural network architectures or specific hyperparameter optimization and allows for the use of transfer learning to initialize the networks.

(B) Training a centerpoint predictor

In the second approach for determining the optimal polar origin, we train a model specifically tasked with predicting the correct polar origin for each input image, which is then

used to transform the input image. The approach is shown in Figure 4.8. We do this by training a neural network based on the stacked hourglass architecture [NYD16] first used for human pose estimation. Instead of training a regressor network to predict key points in an image, the stacked hourglass architecture uses a series of stacked encoder-decoder networks, where the output of each stack is a heatmap centered on the key point to be predicted. The output of each stack is fed as input into the next stack, allowing successive refinement of the heatmap prediction. During training, the loss of each stack's output is averaged to produce the final loss, allowing deep supervision. The final prediction heatmap is the output of the last stack in the network. To predict the center point, we use 8 stacked hourglass blocks, which we empirically determined as the value providing the best results. The network receives images in Cartesian coordinates and predicts a heatmap of the image.

Data Processing

The ground truth heatmaps were generated by calculating the center of mass of each ground truth label image using (4.9). We then create the heatmap as an image with a 2D Gaussian placed on the calculated center of mass and a standard deviation of 8 pixels for all datasets except the liver, and 16 for the liver. The optimal value for the standard deviation was determined empirically on the validation datasets. We found that the optimal value of the standard deviation is proportional to the size of the object. Example heatmaps are shown in Figure 4.9.

Additionally, during training, we use augmentation to increase the number of training inputs. In particular, during training for each input example, the following random augmentations are applied: (1) a 50% chance of a horizontal flip; (2)a 30% chance of a random combination of shifting up to 6.5% of the image dimensions, scaling up to 10% and rotating up to 45°; and (3) a 30% chance for a grid distortion, details of which are described in [Bus+20].

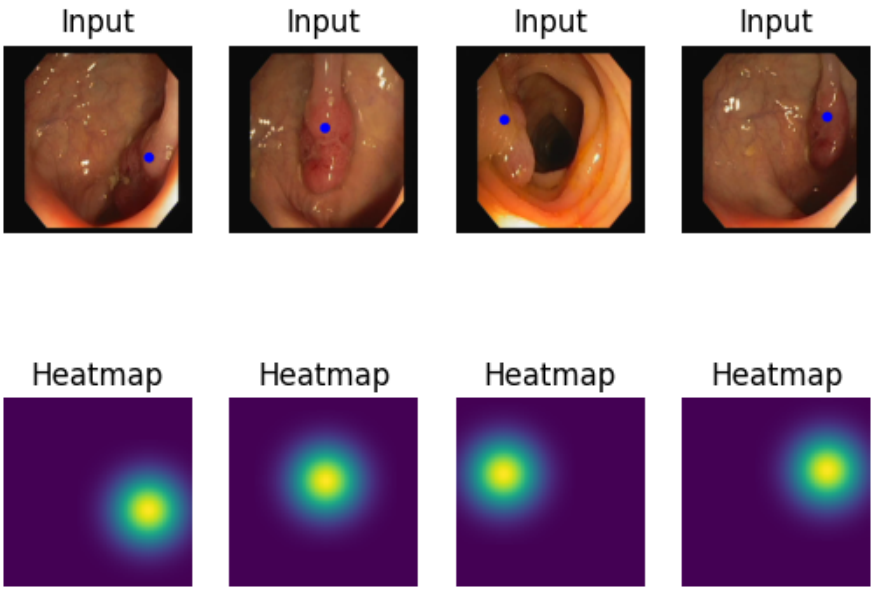


Figure 4.9: Examples of heatmaps generated for the polyp dataset. The heatmap is a Gaussian centered on the center of mass of the ground-truth label of the image. The center is shown as a blue point on the input images. [Ben+21]

The center-point predictor outputs 8 separate heatmaps [NYD16]. We calculate the predicted center as the coordinates of the pixel with the largest intensity in the heatmap predicted by the final layer of the model. This predicted center is then used to transform the input image to polar coordinates, and the transformed image is fed into the polar network to perform the segmentation. Finally, the segmentation label is transformed back to Cartesian coordinates.

4.3.3 Experiments

To validate segmentation accuracy and data efficiency improvements of our method, we trained a variety of neural network architectures on multiple medical imaging datasets. In particular, we trained three different neural network architectures: U-Net [RFB15b], U-Net++ Zhou et al. [Zho+18] with a ResNet encoder and DeepLabV3+ [Che+18] with a ResNet encoder. Notably, each dataset we use presents a problem wherein almost all examples a single roughly elliptical object needs to be segmented. For each dataset and network architecture combination, we train a Cartesian and polar network, and we then perform four different experiments:

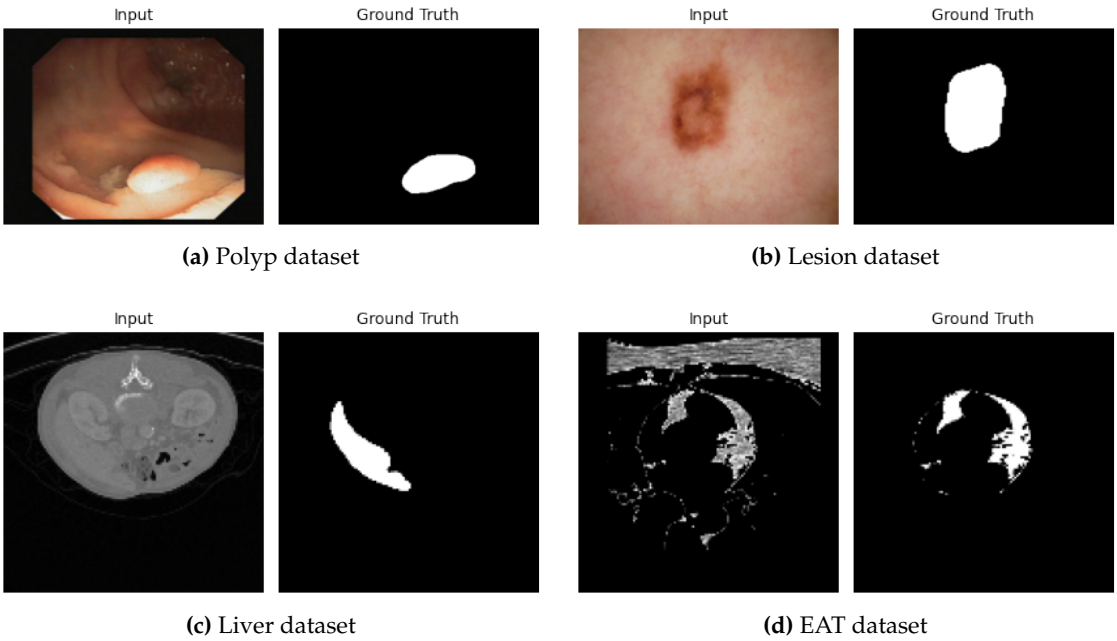
1. testing the Cartesian network using Cartesian images
2. testing the polar network using the ground-truth polar origin
3. testing the polar network using polar origins obtained from predictions of the Cartesian network, as outlined in 4.3.2
4. testing the polar network using polar origins from the centerpoint predictor, as outlined in 4.3.2.

Datasets description

We used four different datasets to train the network. In this section, we give an overview of each used dataset and how it was preprocessed. Note that for training the center-point predictor, the input images were resized to a resolution of 256×256 to improve training times, while the generated heatmaps were resized to 64×64 pixels to improve computational requirements. Otherwise, all preprocessing steps described here are applied to the center-point model datasets as well. Each dataset was globally normalized using predetermined Hounsfield ranges for relevant tissues as minimum and maximum values, and zero-centered based on the mean intensity across the training dataset, to facilitate better network convergence.

Polyp dataset: The CVC-ClinicDB dataset [Ber+15] contains 612 RGB colonoscopy images with the resolution 288×384 with labeled polyps from MICCAI 2015. We normalize each image to a range of $[-0.5, 0.5]$. We use the original image resolution to train all networks except the centerpoint network. As is used in [Jha+20], we use an 80%, 10% and 10% split for training, validation, and testing datasets, respectively. An example of the dataset is shown in Figure 4.10(a).

Liver dataset: The second dataset we use is the LiTS dataset [Bil+19] from the Liver Tumor Segmentation Challenge from MICCAI 2017. The dataset contains 131 CT scans of patients with hepatocellular carcinoma, with the liver as well as tumor lesions labeled by experts. In our experiments, we disregard the lesion segmentation labels and treat the dataset as a binary liver segmentation problem. In addition, we removed all slices that did not contain a ground-truth liver segmentation label, resulting in a dataset of roughly 15,000 slices. Each axial slice is thresholded to a Hounsfield scale range of $[0, 200]$ HU that contains the liver. Next, the slices are normalized to a $[0, 1]$ range and zero-centered



(a) Polyp dataset **(b)** Lesion dataset

(c) Liver dataset **(d)** EAT dataset

Figure 4.10: Example input images and ground-truth labels for each dataset used in our experiments. [Ben+21]

by subtracting the global intensity of all training slices (0.1). We then proceed to train the networks on each axial slice separately. We use 101 scans for training, 15 scans for validation, and the remaining 15 scans for testing. Example liver segmentation images are shown in Figure 4.10(c).

Lesion dataset: The third dataset we use is the ISIC 2018 Lesion Boundary Segmentation dataset [Cod+18; TRK18] which contains 2,694 dermatoscopy images of skin lesions with expert labels of the lesions from various anatomic sites and several different institutions. We resize each image to a resolution of 384×512 and use a training, validation, and test split of 80%, 10%, and 10%, respectively. This is consistent with [Jha+20]. Additionally, we normalize each image to a range of $[-0.5, 0.5]$. An example of a lesion input image and its corresponding label is shown in Figure 4.10(b).

EAT dataset: Finally, we also train on a dataset of labeled EAT regions from 20 patients' cardiac CT scans from the Cardiac Fat Database [Rod+16]. The dataset has three classes labeled: the pericardium, EAT, and pericardial adipose tissue. We disregard all original labels except EAT and treat the dataset as a binary EAT segmentation dataset. The dataset is first split into training (10 patients), validation (5 patients), and test (5 patients) datasets. In the original dataset, each slice is thresholded to the adipose tissue range of $[-200, -30]$ HU and registered so that anatomical structures have the same locations. In addition to these original pre-processing steps, we normalize each slice to a $[0, 1]$ range and zero-center the dataset by subtracting a global mean intensity of the training set (0.1). We then train on each CT slice separately. An input image of the EAT dataset and its corresponding label is shown in Figure 4.10(d).

Implementation details

We use the OpenCV linear polar transformation implementation. Each model is implemented and trained using PyTorch 1.7.1 on an NVIDIA GeForce RTX 3080 GPU. For all networks, we use the Adam optimizer with a learning rate of 10^{-3}. A batch size of 8 was

used for all networks except the center-point model, where a batch size of 6 was used for the lesion and liver datasets and 8 for all remaining datasets. We trained all models up to a maximum of 200 epochs and used checkpoints after each epoch to store the model with the best validation loss. We modify the Dice coefficient to act as a loss function as follows:

$$DSC_{loss} = 1 - \frac{2|X \cap Y| + \lambda}{|X| + |Y| + \lambda}, \tag{4.10}$$

where X and Y are the input and predicted images, respectively, and λ is a smoothing parameter set to 1 in our experiments used to avoid dividing by zero. This loss function is used to train all models except the center-point model.

The centerpoint model outputs eight heatmaps [NYD16]. We use a loss function that is the mean of the mean squared errors between each of the heatmaps and the ground truth heatmap. The code used for all experiments is available at `github.com/marinbenc/medical-polar-training`.

4.3.4 Results and Discussion

We evaluate segmentation performance along with four key metrics: the Dice coefficient (DSC), the median intersection-over-union score (mIoU), precision, and accuracy. Precision and accuracy are both calculated pixel-wise. The results of training the different approaches presented in 4.3.3 are shown in Table 4.4 for polyp, lesion, liver, and EAT segmentation. In all cases, training on polar coordinates improves the segmentation in all metrics when compared to training the same model on Cartesian coordinates. As is to be expected, testing the polar network on images transformed using the ground truth polar origins produces the best results in terms of DSC. A close second predicts the polar origin from the center-point predictor. Predicting polar origins from the Cartesian model leads to less accurate polar origins, and the results are worse in terms of DSC, however, they are still better than using only the Cartesian model.

We also compare our methods to other state-of-the-art methods that use the same datasets, shown in Table 4.5. We achieve state-of-the-art results for the polyp and liver datasets. Additionally, we achieve state-of-the-art liver segmentation when compared to other per-slice methods, and nearly state-of-the-art results when compared to 3D-based methods. For EAT segmentation, our approach outperforms standard medical image segmentation networks but does not achieve state-of-the-art performance due to segmenting EAT directly and not first segmenting the pericardium.

A training graph for a polar and Cartesian U-Net-based network is shown in Figure 4.11. Additionally, we evaluate the accuracy of the different ways of obtaining the polar origin. This accuracy is compared with segmentation performance in Figure 4.12.

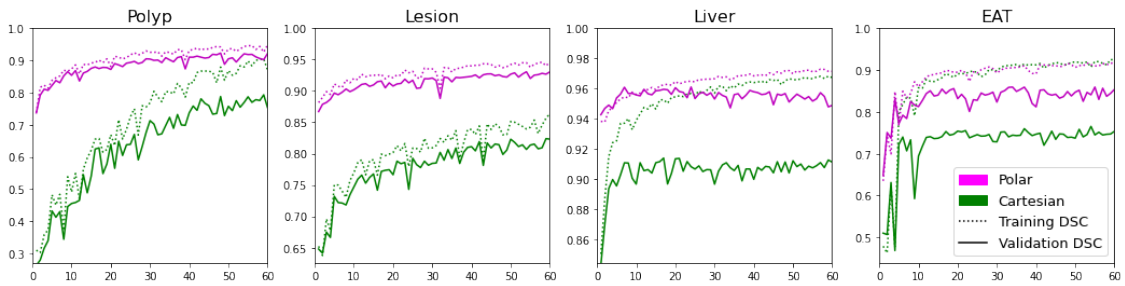


Figure 4.11: The training and validation Dice coefficient (DSC) of the polar and Cartesian U-Net models during training. [Ben+21]

Table 4.1 Results of our proposed approaches for different tasks for three different neural network architectures. The Cartesian network is the network trained on Cartesian images. "GT centers" refers to obtaining a polar origin from the ground-truth labels and segmentation using the polar network."Cartesian centers" refers to predicting the polar origins from the Cartesian network and then performing segmentation using the polar network. "Model centers" refers to using the centerpoint predictor to obtain polar origins. (Continued on the next page.)

Polyp Segmentation

Architecture	Method	DSC	mIoU	Prec.	Rec.
U-Net	Cartesian	0.8315	0.7604	0.8513	0.8334
	GT centers	0.9484	0.9141	0.9563	0.9442
	Cart. centers	0.8973	0.8571	0.8996	0.8998
	Model centers (Ours)	0.9374	0.8977	0.9488	0.9368
Res-U-Net++	Cartesian	0.8356	0.7636	0.9004	0.8256
	GT centers	0.9557	0.9260	0.9583	0.9554
	Cart. centers	0.9063	0.8685	0.9243	0.9027
	Model centers (Ours)	0.9332	0.8924	0.9477	0.9321
DeepLabV3+	Cartesian	0.8706	0.8013	0.8857	0.8876
	GT centers	0.9593	0.9296	0.9576	0.9682
	Cart. centers	0.9212	0.8823	0.9179	0.9397
	Model centers (Ours)	0.9338	0.8967	0.9436	0.9347

Lesion Segmentation

	Method	DSC	mIoU	Prec.	Rec.
U-Net	Cartesian	0.8256	0.7393	0.8407	0.8712
	GT centers	0.9320	0.8824	0.9261	0.9541
	Cart. centers	0.8836	0.8317	0.8746	0.9492
	Model centers (Ours)	0.9224	0.8699	0.9165	0.9494
Res-U-Net++	Cartesian	0.8664	0.7925	0.8728	0.9122
	GT centers	0.9439	0.9014	0.9418	0.9584
	Cart. centers	0.9125	0.8653	0.9075	0.9540
	Model centers (Ours)	0.9253	0.8743	0.9253	0.9464
DeepLabV3+	Cartesian	0.8717	0.7984	0.8807	0.9068
	GT centers	0.9459	0.9059	0.9418	0.9632
	Cart. centers	0.9162	0.8686	0.9097	0.9536
	Model centers (Ours)	0.9235	0.8721	0.9125	0.9570

Table 4.1 Results of our proposed approaches (continued).

Liver Segmentation

Segm. net.	Method	DSC	mIoU	Prec.	Rec.
U-Net	Cartesian	0.8976	0.8505	0.8997	0.9201
	GT centers	0.9553	0.9227	0.9595	0.9569
	Cart. centers	0.9302	0.8985	0.9279	0.9429
	Model centers (Ours)	0.9125	0.8828	0.9108	0.9219
Res-U-Net++	Cartesian	0.8908	0.8463	0.8936	0.9085
	GT centers	0.9548	0.9215	0.9492	0.9661
	Cart. centers	0.9219	0.8898	0.9119	0.9428
	Model centers (Ours)	0.9109	0.8795	0.9009	0.9306
DeepLabV3+	Cartesian	0.8868	0.8341	0.8995	0.8959
	GT centers	0.9518	0.9171	0.9547	0.9550
	Cart. centers	0.9253	0.8932	0.9244	0.9361
	Model centers (Ours)	0.9092	0.8783	0.9075	0.9199

EAT Segmentation

Segm. net.	Method	DSC	mIoU	Prec.	Rec.
U-Net	Cartesian	0.7544	0.5812	0.7190	0.6949
	GT centers	0.8088	0.6607	0.7986	0.7675
	Cart. centers	0.7835	0.6227	0.7455	0.7208
	Model centers (Ours)	0.7840	0.6252	0.7451	0.7302
Res-U-Net++	Cartesian	0.3410	0.1743	0.2700	0.3294
	GT centers	0.8030	0.6827	0.7939	0.8043
	Cart. centers	0.5466	0.3980	0.5286	0.5066
	Model centers (Ours)	0.7740	0.6140	0.7156	0.7453
DeepLabV3+	Cartesian	0.6380	0.4246	0.5665	0.5940
	GT centers	0.6952	0.5123	0.6519	0.6779
	Cart. centers	0.6696	0.4716	0.5988	0.6454
	Model centers (Ours)	0.6720	0.4779	0.6070	0.6488

We also train several models with both polar and Cartesian coordinates on subsets of the training dataset. Namely, we trained models on 25%, 50%, 75%, and 100% of the lesion training dataset for 50 epochs. The results of this training are shown in Figure 4.13. The polar network is much more data efficient and achieves better results than the cartesian network even with only 25% of the data.

Table 4.2 A comparison between our method (model centers approach with best underlying architecture) and the state of the art on the same datasets.

Dataset	Method	DSC	mIoU	Prec.	Rec.
Polyp	PraNet [Fan+20a]	0.8990	0.8490	-	-
	FANet [Tom+21]	-	0.8937	0.9401	0.9339
	Our method	**0.9374**	**0.8977**	**0.9488**	**0.9368**
Lesion	DeepLabV3+ [Che+18]	0.8717	0.7984	0.8807	0.9068
	DoubleU-Net [Jha+20]	0.8962	0.8212	**0.9459**	0.8780
	Our method	**0.9253**	**0.8743**	0.9253	**0.9464**
Liver	U-Net [RFB15b]	0.8976	0.8505	0.8997	0.9201
	KiU-Net 3D [Val+20]	**0.9423**	0.8946	-	-
	Our method	0.9302	**0.8985**	0.9279	0.9429
EAT	U-Net. [RFB15b]	0.7544	0.5812	0.7190	0.6949
	Zhang et al. [Zha+20]	**0.9119**	**0.8425**	-	-
	Our method	0.7840	0.6252	0.7451	0.7302

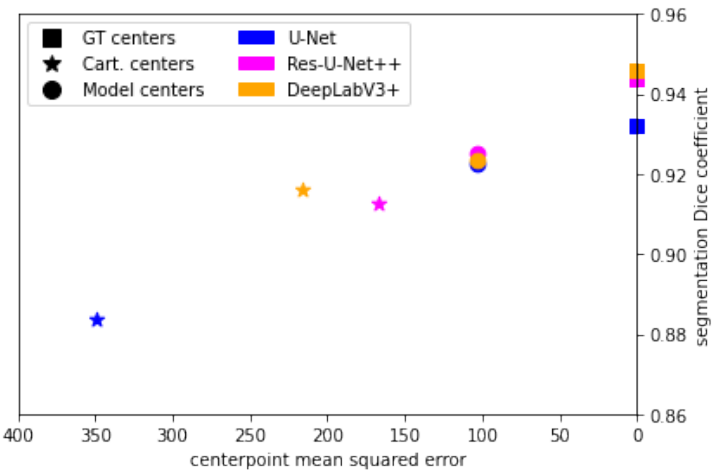


Figure 4.12: The relationship between mean squared errors of the centers used for the polar transformation and segmentation performance of the polar network on the lesion dataset. The mean squared errors are calculated compared to the ground-truth centers. [Ben+21]

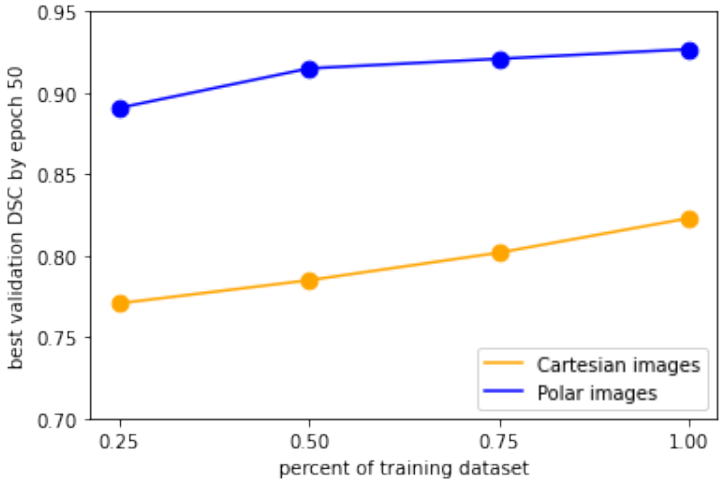


Figure 4.13: The best Dice coefficient by epoch 50 for models trained on subsets of the lesion training dataset. [Ben+21]

Discussion

We obtain state-of-the-art results for polyp and lesion segmentation by training common biomedical image segmentation models. In the liver dataset, we achieve state-of-the-art results when compared to other 2D methods, but 3D methods achieve the same or slightly better results [Val+20]. The liver dataset is by far the largest dataset we evaluated. As such, improvements gained from encoding localization information and reducing dimensionality might not be as large as in smaller datasets, since the network has enough data to learn these complex structures. The EAT dataset is one where the task is not to find a single object, but instead, to segment multiple smaller pockets of tissue around the heart. This task is more challenging for common models like U-Net and requires a more complex approach [Zha+20]. It is possible that combining these existing approaches, namely segmenting the pericardium first, with training on polar coordinates would lead to an improvement in the state of the art.

We also show that training on polar images leads to a significant improvement in segmentation performance when compared to training on Cartesian images for the same network architecture. Additionally, as shown in Figure 4.11, the polar network portions of our approach converge in much fewer epochs than the Cartesian networks. This is in part due to the location information being encoded in the image itself via the polar origin, and in part due to a possible data dimensionality reduction, allowing the network to more easily optimize the loss function. The polar networks are also more robust to low dataset sample size. This is especially important in biomedical image segmentation where the availability of large labeled datasets is often very limited.

Predicting the center point from the Cartesian model, while still an improvement over the plain Cartesian network leads to slightly lower DSC results than those obtained by the center point predictor model. We conclude that segmentation is highly dependent on choosing the correct polar origin. This dependency is somewhat loosened by adding polar origin augmentation when training the polar network.

Figure 4.14 shows a random sampling of predictions from the polar network using the center point predictor for polar origins. Qualitatively, we conclude that the network segmentation achieves a high overlap with the target object. The network successfully segments both small and elliptical as well as large and unevenly shaped polyps. On the lesion images, the network predicts a smooth border when sometimes the actual border

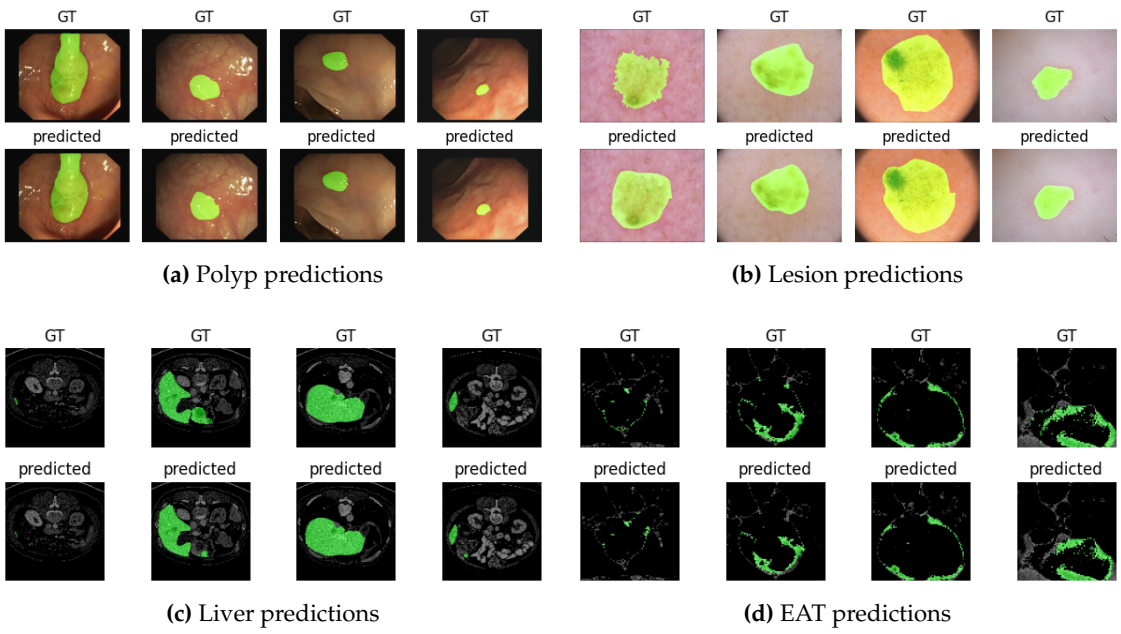
Table 4.3 Ablation study of our approach for the polyp dataset.

Method	DSC	Difference
Cartesian	0.8315	-
Polar (Cartesian origins)	0.8918	+0.0603
Polar (Centerpoint predictor)	0.9094	+0.0176
+ centerpoint augmentation	0.9288	+0.0194
+ polar network training augmentation	0.9374	+0.0086

of the lesion is rough, as shown in the left-most example on Figure 4.14(b), however, the network still does a good job of delineating a lesion border even when the color of the lesion is very similar to the surrounding skin. The network successfully predicts a liver border both when the liver is very large and very small on the image, showing good scale invariance, but sometimes under segments the liver when multiple connected components are needed. On the EAT dataset, the network successfully learns to segment EAT despite its highly discontinuous and sparse distribution. However, the network sometimes under segments EAT.

Finally, we also perform an ablation study shown in Table 4.3. Training on the polar coordinates with the polar origins predicted from the cartesian network yields the largest performance improvement. Predicting the polar origin from the center point predictor as well as adding center point augmentation to the predictor play a roughly equally important role in the performance. Lastly, a small performance improvement is further achieved by using data augmentation when training the polar network.

The center points could also be obtained by a more basic segmentation approach like



(a) Polyp predictions **(b)** Lesion predictions

(c) Liver predictions **(d)** EAT predictions

Figure 4.14: A random sampling of inverse polar transformed predictions from the polar network with the polar origins predicted from the center-point predictor for various datasets. The prediction is shown in green and overlaid on top of the original input image. EAT predictions (d) are cropped and zoomed to better show the predictions. [Ben+21]

thresholding or other traditional image processing methods, leading to a possible reduction in the number of required neural network parameters to achieve good segmentation. Furthermore, in our experiments, we found that the segmentation is dependent on choosing the correct standard deviation of the generated heatmaps for training the center point predictor. An improvement to our method could be made by developing a method to automatically estimate the standard deviation from the training or validation data without needing to first train the center point predictor.

In conclusion, the polar transform demonstrates significant potential in simplifying the segmentation of elliptical objects, thus reducing the complexity of the neural network model required. However, a key limitation of this approach is that only one elliptical object can be segmented. The next section will introduce further advancements to this method, allowing the segmentation of multiple objects on the image.

4.4 Supporting Multiple Transformations of an Image

In this chapter, we've explored an approach where each image undergoes a single transformation before being processed by the segmentation network. We will now extend this model-driven preprocessing approach to accommodate multiple transformations of the same input image. This modification is particularly useful in scenarios where an image contains several objects, each with its distinct decision boundary. By applying a separate polar transformation for each object center, we can generate multiple segmentation maps for a single image. These maps can then be combined to create a unified segmentation result. The process involves the following steps:

1. Our transformation predictor network will be enhanced to predict a set of transformation parameters $\{\phi_1, \phi_2, \cdots, \phi_n\}$, where n varies based on the image and represents the number of transformations. n is dynamically chosen by the network and corresponds to the predicted number of objects in the image.

2. For each transformation parameter in ϕ_i we will produce a corresponding transformed image $I'_i = \mathcal{T}_{\phi_i}(I)$.

3. Each transformed image I'_i will be segmented using the segmentation network, producing a series of segmentation maps $\{M_i\}$, where $M_i = F_{seg}(I'_i)$.

4. Finally, all of the segmentations will be fused using an aggregation function f_{agg}, resulting in a final segmentation map $M = f_{agg}(\{M_i\})$.

To exemplify this, we will present a case of using the polar transformation as we did earlier, only this time we will be segmenting images with multiple connected components (objects) in their segmentation maps. Therefore, we will use multiple polar transformations for each image, one for each connected component. The transformation parameter predictor will be modified to predict the centroid of each connected component, instead of just a single centroid. Each centroid will be used to transform and segment the image. For the aggregation function, we will choose a weighted sum and a hysteresis thresholding function th with thresholds t_1 and t_2:

$$f_{agg}(\{M_i\}) = th(\sum_{i=1}^{n} M_i W_i; t_1, t_2), \tag{4.11}$$

each weight W_i functions as an image matching the dimensions of M_i, where the pixels correspond to the relative importance of that pixel in the corresponding segmentation

map. For instance, in the case of the polar transform, the value assigned to each element in W_i depends on its relation to the connected component associated with the polar origin for the corresponding segmentation mask M_i. Specifically, elements that belong to the same connected component as the polar origin used for the i-th segmentation are assigned a value of 2. Elements outside this connected component are assigned a value of 1. With this weighing strategy, regions belonging to the connected component that is most beneficially transformed are more heavily weighted than other pixels.

The hysteresis thresholding function th, defined by two thresholds t_1 and t_2, operates in two stages. Initially, it eliminates all pixels with intensity levels below t_1. First, it removes all pixels with an intensity lower than t_1. Next, in the remaining pixels of the image, for each connected component, if any of the component's pixel intensities are higher than t_2, the whole connected component is preserved, otherwise the connected component is removed, as illustrated in Figure 4.15.

The approach of model-driven preprocessing using multiple transformations has several advantages. First, each object can benefit from the reduction in segmentation complexity. For example, in the case of the polar transform, one object is always the bottom part of the polar transformed image to be segmented, so the network does not need to learn to first localize the object. Secondly, a consequence of this approach is that images with multiple connected components are over-sampled during training. This is beneficial since these images are usually under-represented (when compared to images with fewer connected components) and harder to segment (since they require segmenting multiple objects).

What follows is an empirical study of this approach for aorta segmentation, which is a modification of the one presented in 4.3 to allow support for multiple transformations of the input image, one for each object in the image.

The aorta is the largest artery of the human body and supplies oxygenated blood from the heart to all parts of the body. It is one of the most clinically significant structures to analyze for cardiovascular disease diagnosis and prevention. Several conditions could occur on the aorta which can be detected using 3D medical imaging, including aneurysms, dissections, stenoses, coarctations, or traumas. All of these conditions can be dangerous and require careful screening, following, and potentially surgical treatment, while a failure or delay in the diagnosis of these conditions could be fatal. Therefore, developing a fully automated method to efficiently and accurately segment the aorta could be beneficial for earlier detection of these conditions. By producing a 3D model of the aorta from CT or MRI scans, a computer algorithm could perform automatic measurements to screen and detect aortic aneurysms, dissections, and other conditions that are

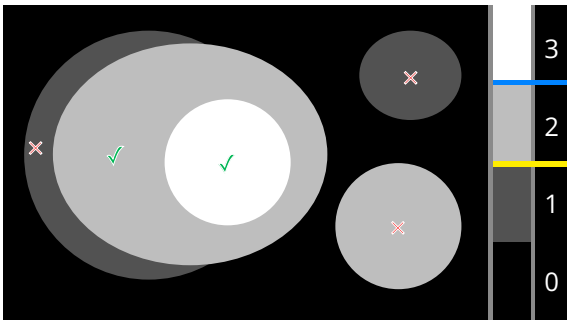


Figure 4.15: A visual explanation of hysteresis thresholding. Kept regions are marked with green checkmarks, while deleted regions are marked with red crosses. The pixel intensity scale is shown on the right, where t_1 is marked in yellow and t_2 is marked in blue.

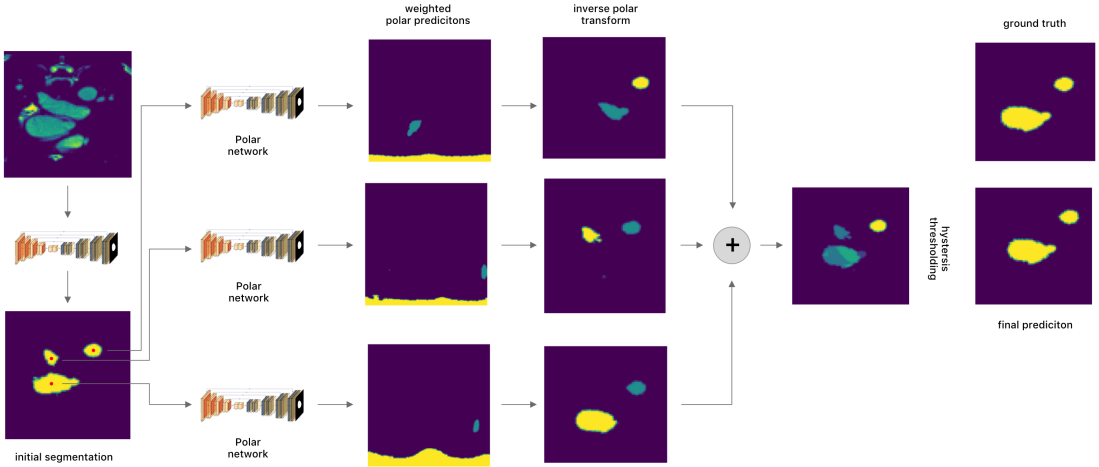


Figure 4.16: A summary of our approach. An input image is first segmented using a U-Net network. For each connected component in the segmentation, the input image is transformed to polar coordinates using the centroid of the connected component as the origin. These images are then fed into a U-Net trained on polar images, and the predictions for each object are fused, hysteresis thresholded, and transformed back to cartesian coordinates. Note how one of the false positive connected components in the initial segmentation was removed during hysteresis thresholding since the component was only predicted in one of the three polar predictions. [Ben+22]

commonly diagnosed by imaging the aorta.

Several methods based on deep learning were proposed for segmenting the aorta from CT images. Fantazzini et al. [Fan+20b] use a cascade of U-Net-based networks. They first perform a rough segmentation on axial slices to extract a region of interest. They then use separate networks to segment axial, sagittal, and coronal slices of the region of interest. Several other papers have used 3D U-Net-based architectures for this task [Yu+21; Che+21a].

4.4.1 Methodology

The approach presented in this section is an extension of the approach presented in 4.3. We obtain the center points of the objects in the image using a rough segmentation from a U-Net-based network, instead of using a center point predictor as is described in 4.3. A summary of the approach is shown in Figure 5.1.

As described earlier in this section, we perform several key modifications to allow the network to segment multiple objects on an image. During the training of the polar network, we construct a dataset that contains one polar transformation per connected component in the ground truth segmentation label. The origins of these polar transformations are the centroid of each corresponding connected component.

We also employ prediction fusion during inference. First, a 2D U-Net-based network is used to obtain an initial rough segmentation. A separate polar transformation for each connected component in the rough segmentation is constructed using the centroid of each component as the origin. The polar network predicts a segmentation map for each transform, resulting in a number of predictions equal to the number of connected components. In the predicted image, a weight of 2 is assigned to the connected component which contains the origin for that prediction, and a weight of 1 to all other connected

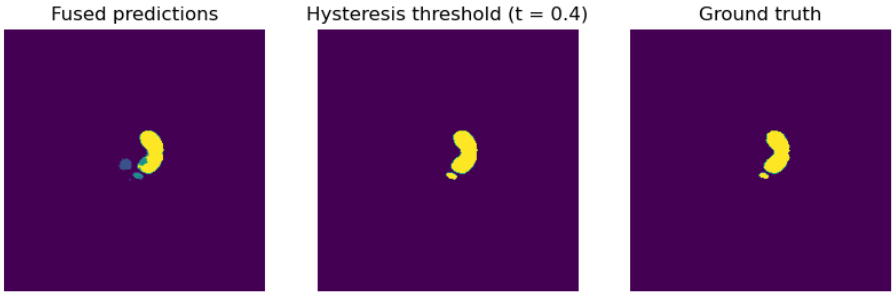


Figure 4.17: Hystersis-thresholded segmentation output. For each polar prediction, the component that contains the origin of the transform gets a weight of 2 assigned, while all other components get a weight of 1. This left-most image is the result of summing the predictions of 3 polar transformations of the original image (one for each connected component), converted to cartesian coordinates. Note how the thresholding removes the false positive object on the left of the image while keeping the true positive objects intact. [Ben+22]

components. We then sum all of the weighted images together. As shown earlier in this chapter, the polar network generally performs best on objects which contain the polar origin, and worse at predicting other objects on the image. Therefore, we assign a larger weight to that component as a proxy for a confidence measure. We then sum all of the weighted predictions together and normalize the prediction to a 0-1 range. This leads to a segmentation map where each non-zero pixel represents the confidence that the pixel belongs to the aorta class. To obtain the final segmentation, we use hysteresis thresholding where the bottom threshold is 0, and the top threshold is 0.4, empirically determined according to the best Dice coefficient on the validation dataset. An example of thresholding a prediction is shown in Figure 4.17.

Data Description and Preprocessing

We used a publicly available dataset of CT scans with corresponding aorta labels [Rad+22] including the ascending aorta, the aortic arch as well as the descending and abdominal aorta. While the original dataset contains scans from three different centers, in our experiments, we only use the data from Dongyang Hospital. In total, we use 18 CT scans, each containing 122-251 slices, with a slice thickness of 2 or 3 mm.

Each CT slice is windowed to a range of 200 to 500 HU to remove information from irrelevant tissues, then normalized to a range of -0.5 to 0.5, and zero centered using the global mean value across all slices in the validation set. The slices were each resized from 512×666 to 256×256 pixels. We use augmentation during training for both the cartesian and the polar network. The augmentations we use include a 50% chance of a random combination of affine transforms including a shift of up to 6.25%, a scale of up to 10% and a rotation of up to 15°; as well as a 30% chance of a horizontal flip.

All of our models were implemented using PyTorch 3.9 using an NVIDIA GeForce RTX 3080 GPU. We use the U-Net [RFB15a] architecture for both the cartesian and the polar network. For training, we use a batch size of 8 and the Adam optimizer with a learning rate of 0.001. All models were trained for 60 epochs with checkpointing where the model with the best validation Dice coefficient was selected. We use the Dice loss function as described in the previous section of this chapter. All of the code, as well as the trained networks, can be found at github.com/marinbenc/medical-polar-training.

4.4.2 Results and Discussion

To perform evaluation, we use 3-fold cross-validation on the 18 scans. For each fold, we train a polar and non-polar model using the slices of 12 CT scans and run inference on the slices of the remaining 6 scans. All results presented in this section are averaged across each CT scan and then across the three folds.

A summary of our segmentation results is presented in Table 4.4. Random examples of segmentation results are shown in Figure 6.5. In the experiments in 4.3 the polar

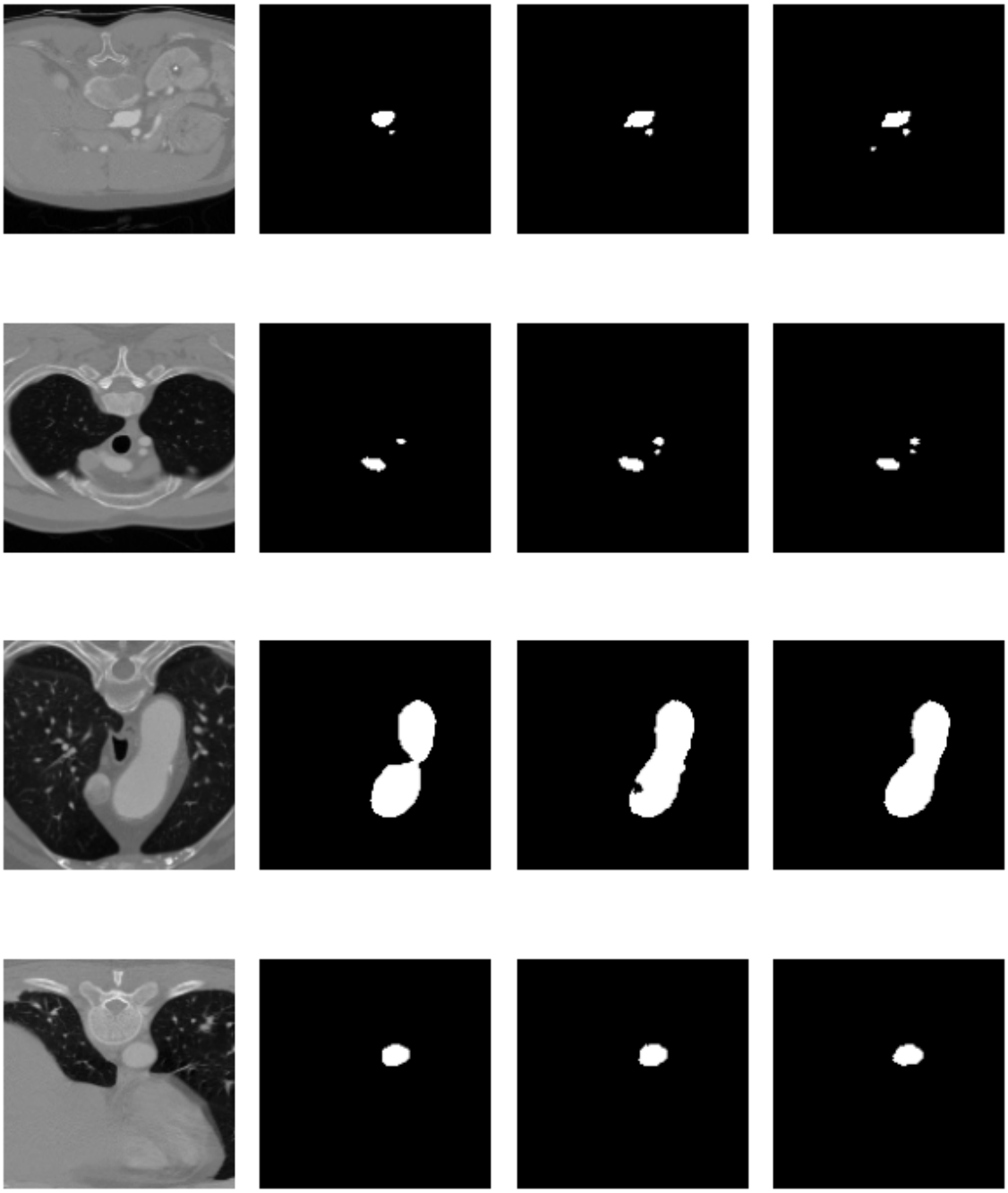


Figure 4.18: Random examples of predictions. Columns from left to right show: the input image, the initial prediction from the non-polar network, the final fused polar prediction, and the ground truth segmentation label. [Ben+22]

networks achieve the best segmentation performance when using accurate center points during inference. As the accuracy of the center points goes down, so does the segmentation performance. The experiments in this section follow the same pattern, the polar networks perform significantly better than the cartesian networks when using ground truth centers. However, even with less accurate centers obtained from initial rough segmentation by the non-polar network, the results yield only slightly lower Dice coefficients than when using ground-truth centers directly. The non-polar network can also be seen as a baseline model, and our approach results in a significant improvement over this baseline in all segmentation metrics.

In some problems in medical imaging, e.g. segmenting cancerous tissues, a higher recall is beneficial since the cost of missing tissues can be very high [TH15]. A key advantage of our approach is that by fusing multiple predictions and using hysteresis thresholding the threshold value can be used to impact the precision–recall tradeoff. Our experiments show that the average per-patient pixel-level recall increased when compared to the baseline model.

We also present the standard deviation across CT scans as a measure of segmentation reliability. Using the polar coordinates decreases the standard deviation between patients of all performance metrics, indicating that the predictions are more reliable and more robust to inter-patient differences. To further emphasize this, we present a box plot of segmentation results for each patient in Figure 4.19. Note that, in contrast with the baseline model, when using our approach there are no outliers in the box plot.

A comparison of our results with other deep learning-based approaches for aorta segmentation in the literature is shown in Table 4.5. Our approach achieves performance comparable to the state of the art, and a large improvement over the baseline methods. Note that the results are evaluated on different datasets and with a different number of cases. Therefore, it is hard to objectively compare these approaches.

Table 4.4 A summary of the mean segmentation results of our experiments. *Non-polar* are the results of the U-Net trained using cartesian images. *Polar + GT centers* are the results of the U-Net trained on polar images, using ground-truth connected component centers during inference, as an example of the best case possible results. *Polar + NP centers* are the results when running inference on the polar model using center points obtained from the non-polar model predictions. *Our proposed method.

Method	**DSC**	**mIoU**	**Prec.**	**Rec.**
U-Net (non-polar)	0.886 ± 0.049	0.825 ± 0.052	0.901 ± 0.074	0.893 ± 0.039
Polar + GT centers	0.937 ± 0.053	0.895 ± 0.055	0.944 ± 0.064	0.937 ± 0.040
Polar + NP centers*	0.932 ± 0.027	0.895 ± 0.033	0.915 ± 0.040	0.973 ± 0.018

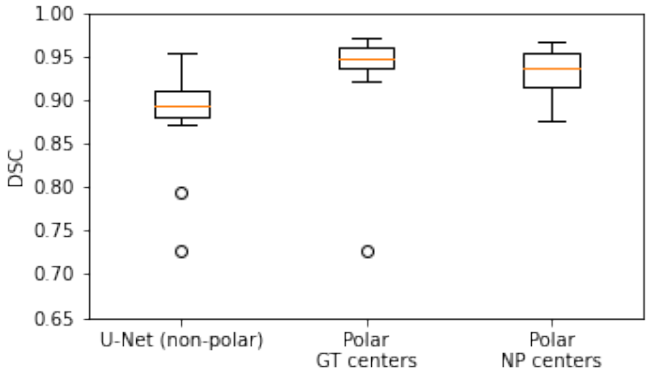


Figure 4.19: A box plot of the per-scan Dice coefficients of our experiments. *Non-polar* are the results of the U-Net trained using cartesian images. *Polar + GT centers* are the results of the U-Net trained on polar images, using ground-truth connected component centers during inference. *Polar + NP centers* are the results when running inference on the polar model using center points obtained from the non-polar model predictions. [Ben+22]

Table 4.5 A comparison of our approach with results reported in papers describing deep learning-based aorta segmentation methods. Note that the datasets used for obtaining the results are not the same. n is the number of cases used to obtain the evaluation.

Method	**DSC**	**mIoU**	**n**
Yu et al. [Yu+21]	0.958	-	25
Fantazzini et al. [Fan+20b]	0.928 ± 0.013	0.866 ± 0.023	10
Cheung et al. [Che+21b]	0.912	-	14
Proposed method	0.932 ± 0.027	0.895 ± 0.033	18

4.5 Conclusion

In this chapter, we've outlined an approach to enhance data efficiency in medical image segmentation by employing two smaller neural networks, rather than a single large one. The first network is designed to predict parameters for a strategically chosen transformation, aimed at reducing the complexity of segmentation. The second network then handles the segmentation of these transformed images. Our empirical studies, particularly with the polar transform, demonstrate the efficacy of this approach across various medical imaging segmentation tasks. We observed notable improvements in training efficiency, especially in cases involving the segmentation of a single, roughly elliptical object. Furthermore, segmentation of polar-transformed images has shown state-of-the-art results in small datasets and near state-of-the-art performance in larger datasets, even when utilizing generic, low-parameter-count models such as U-Net.

We've enhanced this approach by introducing the capability to perform multiple transformations on a single image, followed by the fusion of segmentation results from each transformation using hysteresis thresholding. Applying this method using polar transformations of CT images of the aorta improved segmentation results compared to baseline networks trained on Cartesian images. This improvement was observed across various metrics without a substantial increase in training time or network architecture complexity. Moreover, the fusion of individual object segmentations on an image through hysteresis thresholding has proven beneficial in increasing pixel-level recall, a

valuable trait in medical image segmentation tasks.

We have further improved this approach by allowing multiple transformations of the same image and fusing segmentations of each transformation using hysteresis thresholding. When applied to the polar transform, we showed that this leads to large improvements over baseline networks trained on cartesian images for segmenting the aorta. We see improvements across a variety of metrics without significantly increasing training times or the complexity of the used neural network architectures. In addition, by fusing separate predictions of different objects on the image with hysteresis thresholding we can increase pixel-level recall (at the cost of accuracy) which is often beneficial in medical image segmentation tasks.

This is a flexible approach that allows leveraging domain knowledge as it can be used with any invertible image transformation. Moreover, it is invariant to the downstream segmentation architecture as it is a separate network. Thus, it can be used as a general preprocessing step regardless of the segmentation method. Moreover, if the transformation parameter predictor and segmentation networks share similar architectures, one can use transfer learning between the two networks. The approaches presented in this chapter have been published in [Ben+21] and [Ben+22].

References

[Ben+21] Marin Benčević, Irena Galić, Marija Habijan, and Danilo Babin. "Training on Polar Image Transformations Improves Biomedical Image Segmentation". In: *IEEE Access* 9 (2021), pp. 133365–133375. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3116265.

[Ben+22] Marin Benčević, Marija Habijan, Irena Galić, and Danilo Babin. "Using the Polar Transform for Efficient Deep Learning-Based Aorta Segmentation in CTA Images". In: *2022 International Symposium ELMAR*. Zadar, Croatia: IEEE, Sept. 2022, pp. 191–194. ISBN: 978-1-66547-003-2. DOI: 10.1109/ELMAR55880.2022.9899786.

[Ber+15] Jorge Bernal, F. Javier Sánchez, Gloria Fernández-Esparrach, Debora Gil, Cristina Rodríguez, et al. "WM-DOVA Maps for Accurate Polyp Highlighting in Colonoscopy: Validation vs. Saliency Maps from Physicians". In: *Computerized Medical Imaging and Graphics* 43 (July 2015), pp. 99–111. ISSN: 08956111. DOI: 10.1016/j.compmedimag.2015.02.007. (Visited on 04/13/2021).

[Bil+19] Patrick Bilic, Patrick Ferdinand Christ, Eugene Vorontsov, Grzegorz Chlebus, Hao Chen, et al. "The Liver Tumor Segmentation Benchmark (LiTS)". In: *arXiv:1901.04056 [cs]* (Jan. 2019). arXiv: 1901.04056 [cs]. (Visited on 04/13/2021).

[Bus+20] Alexander Buslaev, Vladimir I. Iglovikov, Eugene Khvedchenya, Alex Parinov, Mikhail Druzhinin, et al. "Albumentations: Fast and Flexible Image Augmentations". In: *Information-an International Interdisciplinary Journal* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125.

[Che+18] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss. Vol. 11211. Cham: Springer International Publishing, 2018, pp. 833–851. ISBN: 978-3-030-01233-5 978-3-030-01234-2. DOI: 10.1007/978-3-030-01234-2_49. (Visited on 08/23/2021).

[Che+21a] Duanduan Chen, Xuyang Zhang, Yuqian Mei, Fangzhou Liao, Huanming Xu, et al. "Multi-Stage Learning for Segmentation of Aortic Dissections Using a Prior Aortic Anatomy Simplification". In: *Medical Image Analysis* 69 (Apr. 2021), p. 101931. ISSN: 13618415. DOI: 10.1016/j.media.2020.101931. (Visited on 05/05/2022).

[Che+21b] Wing Keung Cheung, Robert Bell, Arjun Nair, Leon Menezies, Riyaz Patel, et al. *A Computationally Efficient Approach to Segmentation of the Aorta and Coronary Arteries Using Deep Learning*. Preprint. Cardiovascular Medicine, Feb. 2021. DOI: 10.1101/2021.02.18.21252005. (Visited on 05/09/2022).

[Cod+18] Noel C. F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, et al. "Skin Lesion Analysis toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. Washington, DC: IEEE, Apr. 2018, pp. 168–172. ISBN: 978-1-5386-3636-7. DOI: 10.1109/ISBI.2018.8363547. (Visited on 08/23/2021).

[Est+18] Carlos Esteves, Christine Allen-Blanchette, Xiaowei Zhou, and Kostas Daniilidis. "Polar Transformer Networks". In: *arXiv:1709.01889 [cs]* (Feb. 2018). arXiv: 1709.01889 [cs]. (Visited on 04/01/2021).

[Fan+20a] Deng-Ping Fan, Ge-Peng Ji, Tao Zhou, Geng Chen, Huazhu Fu, et al. "PraNet: Parallel Reverse Attention Network for Polyp Segmentation". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2020*. Ed. by Anne L. Martel, Purang Abolmaesumi, Danail Stoyanov, Diana Mateus, Maria A. Zuluaga, et al. Vol. 12266. Cham: Springer International Publishing, 2020, pp. 263–273. ISBN: 978-3-030-59724-5 978-3-030-59725-2. DOI: 10.1007/978-3-030-59725-2_26. (Visited on 08/23/2021).

[Fan+20b] Alice Fantazzini, Mario Esposito, Alice Finotello, Ferdinando Auricchio, Bianca Pane, et al. "3D Automatic Segmentation of Aortic Computed Tomography Angiography Combining Multi-View 2D Convolutional Neural Networks". In: *Cardiovascular Engineering and Technology* 11.5 (Oct. 2020), pp. 576–586. ISSN: 1869-408X, 1869-4098. DOI: 10.1007/s13239-020-00481-z. (Visited on 05/05/2022).

[He+16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. "Deep Residual Learning for Image Recognition". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA: IEEE, June 2016, pp. 770–778. ISBN: 978-1-4673-8851-1. DOI: 10.1109/CVPR.2016.90. (Visited on 08/23/2021).

[Jad+16] Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. "Spatial Transformer Networks". In: *arXiv:1506.02025 [cs]* (Feb. 2016). arXiv: 1506.02025 [cs]. (Visited on 04/26/2021).

[Jha+20] Debesh Jha, Michael A. Riegler, Dag Johansen, Pal Halvorsen, and Havard D. Johansen. "DoubleU-Net: A Deep Convolutional Neural Network for Medical Image Segmentation". In: *2020 IEEE 33rd International Symposium on Computer-Based Medical Systems (CBMS)*. Rochester, MN, USA: IEEE, July 2020, pp. 558–564. ISBN: 978-1-72819-429-5. DOI: 10.1109/CBMS49503.2020.00111. (Visited on 08/23/2021).

[Kim+18] B.-S. Kim, J.-Y. Sun, S.-W. Kim, M.-C. Kang, and S.-J. Ko. "CNN-based UGS Method Using Cartesian-to-polar Coordinate Transformation". In: *Electronics Letters* 54.23 (Nov. 2018), pp. 1321–1322. ISSN: 0013-5194, 1350-911X. DOI: 10.1049/el.2018.5051. (Visited on 04/01/2021).

[Kim+20] Jinpyo Kim, Wooekun Jung, Hyungmo Kim, and Jaejin Lee. "CyCNN: A Rotation Invariant CNN Using Polar Mapping and Cylindrical Convolution Layers". In: *arXiv:2007.10588 [cs, eess]* (July 2020). arXiv: 2007.10588 [cs, eess]. (Visited on 04/08/2021).

[Liu+19] Qing Liu, Xiaopeng Hong, Wei Ke, Zailiang Chen, and Beiji Zou. "DDNet: Cartesian-polar Dual-domain Network for the Joint Optic Disc and Cup Segmentation". In: *arXiv:1904.08773 [cs]* (Apr. 2019). arXiv: 1904.08773 [cs]. (Visited on 04/01/2021).

[NYD16] Alejandro Newell, Kaiyu Yang, and Jia Deng. "Stacked Hourglass Networks for Human Pose Estimation". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling. Vol. 9912. Cham: Springer International Publishing, 2016, pp. 483–499. ISBN: 978-3-319-46483-1 978-3-319-46484-8. DOI: 10.1007/978-3-319-46484-8_29. (Visited on 08/23/2021).

[Rad+22] Lukas Radl, Yuan Jin, Antonio Pepe, Jianning Li, Christina Gsaxner, et al. "AVT: Multicenter Aortic Vessel Tree CTA Dataset Collection with Ground Truth Segmentation Masks". In: *Data in Brief* 40 (Feb. 2022), p. 107801. ISSN: 23523409. DOI: 10.1016/j.dib.2022.107801. (Visited on 05/04/2022).

[RFB15a] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. May 2015. arXiv: 1505.04597 [cs]. (Visited on 10/06/2023).

[RFB15b] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Ed. by Nassir Navab, Joachim Hornegger, William M. Wells, and Alejandro F. Frangi. Vol. 9351. Cham: Springer International Publishing, 2015, pp. 234–241. ISBN: 978-3-319-24573-7 978-3-319-24574-4. DOI: 10.1007/978-3-319-24574-4_28. (Visited on 08/23/2021).

[Rod+16] É.O. Rodrigues, F.F.C. Morais, N.A.O.S. Morais, L.S. Conci, L.V. Neto, et al. "A Novel Approach for the Automated Segmentation and Volume Quantification of Cardiac Fats on Computed Tomography". In: *Computer Methods and Programs in Biomedicine* 123 (Jan. 2016), pp. 109–128. ISSN: 01692607. DOI: 10.1016/j.cmpb.2015.09.017. (Visited on 04/13/2021).

[Sal+18] Hojjat Salehinejad, Shahrokh Valaee, Tim Dowdell, and Joseph Barfett. "Image Augmentation Using Radial Transform for Training Deep Neural Networks". In: *2018 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. Calgary, AB: IEEE, Apr. 2018, pp. 3016–3020. ISBN: 978-1-5386-4658-8. DOI: 10.1109/ICASSP.2018.8462241. (Visited on 08/23/2021).

[TH15] Abdel Aziz Taha and Allan Hanbury. "Metrics for Evaluating 3D Medical Image Segmentation: Analysis, Selection, and Tool". In: *BMC Medical Imaging* 15.1 (Dec. 2015), p. 29. ISSN: 1471-2342. DOI: 10.1186/s12880-015-0068-x. (Visited on 05/10/2022).

[Tom+21] Nikhil Kumar Tomar, Debesh Jha, Michael A. Riegler, Håvard D. Johansen, Dag Johansen, et al. "FANet: A Feedback Attention Network for Improved Biomedical Image Segmentation". In: *arXiv:2103.17235 [cs, eess]* (Mar. 2021). arXiv: 2103.17235 [cs, eess]. (Visited on 04/19/2021).

[TRK18] Philipp Tschandl, Cliff Rosendahl, and Harald Kittler. "The HAM10000 Dataset, a Large Collection of Multi-Source Dermatoscopic Images of Common Pigmented Skin Lesions". In: *Scientific Data* 5.1 (Dec. 2018), p. 180161. ISSN: 2052-4463. DOI: 10.1038/sdata.2018.161. (Visited on 04/13/2021).

[Val+20] Jeya Maria Jose Valanarasu, Vishwanath A. Sindagi, Ilker Hacihaliloglu, and Vishal M. Patel. "KiU-Net: Overcomplete Convolutional Architectures for Biomedical Image and Volumetric Segmentation". In: *arXiv:2010.01663 [cs, eess]* (Oct. 2020). arXiv: 2010.01663 [cs, eess]. (Visited on 04/22/2021).

[Yu+21] Yitong Yu, Yang Gao, Jianyong Wei, Fangzhou Liao, Qianjiang Xiao, et al. "A Three-Dimensional Deep Convolutional Neural Network for Automatic Segmentation and Diameter Measurement of Type B Aortic Dissection". In: *Korean Journal of Radiology* 22.2 (2021), p. 168. ISSN: 1229-6929, 2005-8330. DOI: 10.3348/kjr.2020.0313. (Visited on 05/05/2022).

[Zha+20] Qi Zhang, Jianhang Zhou, Bob Zhang, Weijia Jia, and Enhua Wu. "Automatic Epicardial Fat Segmentation and Quantification of CT Scans Using Dual U-Nets With a Morphological Processing Layer". In: *IEEE Access* 8 (2020), pp. 128032–128041. ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.3008190. (Visited on 04/22/2021).

[Zho+18] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Ed. by Danail Stoyanov, Zeike Taylor, Gustavo Carneiro, Tanveer Syeda-Mahmood, Anne Martel, et al. Vol. 11045. Cham: Springer International Publishing, 2018, pp. 3–11. ISBN: 978-3-030-00888-8 978-3-030-00889-5. DOI: 10.1007/978-3-030-00889-5_1. (Visited on 08/23/2021).

[Zho+19] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation". In: *IEEE Transactions on Medical Imaging* (2019).

5 Reducing Model Input Sizes with Model-Driven Crops

In this chapter, we will present a simple but powerful way of reducing the sizes of input images to segmentation neural networks using image crops. We will show why reducing the input size can lead to an increase in data efficiency. Finally, we will evaluate the approach empirically using two similar methods on various medical image modalities.

In Chapter 3 of this thesis we discussed how increasing the number of parameters of a model allows it to overfit more easily. Additionally, we discussed how more complex problems require more parameters to be modeled. Thus, for small datasets, it is beneficial to reduce the complexity of the problem to allow for using networks of fewer parameters. A simple way to reduce the complexity of problems in convolutional neural networks is to reduce the size of the images.

To understand how reducing input size reduces model complexity, let us revisit the concept of sample complexity $n(\epsilon, \delta)$ and its established bounds [SB14]:

$$n(\epsilon, \delta) \geq \frac{\log(|\mathcal{F}|/\delta)}{\epsilon}, \tag{5.1}$$

where n represents the minimum number of samples needed for a model from the hypothesis space $\mathcal{F}$ to achieve a performance that is 'probably approximately correct'—that is, within an error margin ϵ with a confidence level of $1 - \delta$. It is clear from this relation that as the size of the hypothesis space expands, so does the sample complexity.

The hypothesis space refers to the set of all functions that the model is capable of approximating, encompassing every conceivable mapping from the domain $\mathcal{D}$ to the codomain $\mathcal{C}$. Therefore, the magnitude of the hypothesis space can be expressed as

$$|\mathcal{F}| = |\mathcal{C}|^{|\mathcal{D}|}. \tag{5.2}$$

For binary CNN-based image segmentation, the model maps c-channeled images $I \in \mathbb{R}^{w \times h \times c}$ to binary masks $M \in \{0, 1\}^{w \times h}$. The hypothesis space thus grows with the image dimensions w, h, and the number of channels c. Reducing image size directly decreases the hypothesis space, lowering the sample size needed for effective training. This relationship is supported by empirical findings — Tan and Le [TL20] show that convolutional neural networks perform best when input size, network depth, and width are scaled proportionally.

Smaller input sizes also offer technical benefits, notably reduced memory demands. Since CNN memory requirements grow exponentially with image size, larger inputs significantly limit batch size during training. This constraint not only slows down the training process but can also affect the accuracy of gradient estimates per batch, potentially destabilizing training.

The method of image scaling is crucial for performance in medical image segmentation tasks, where images are typically downscaled uniformly. This process can lead to the loss of critical fine details, such as edges, which are vital for segmentation accuracy in medical imaging. For example, the pericardium's thin structure around the heart, often

less than 2 mm and represented by only a few pixels in cardiac CT scans, can be obscured or entirely lost through downsampling. This information loss is particularly detrimental for small objects, which are prevalent in medical images.

An effective way to overcome this loss of information while still reducing the image is cropping the image to a region of interest. By focusing on the area containing the target object for segmentation, the network's learning task is simplified, removing the need to filter out irrelevant background areas. Additionally, centering and uniformly scaling the object across images can further reduce the variance of the data, enabling the network to use fewer parameters. Given this, we will now propose a method to effectively use image cropping as a way of reducing neural network input size.

5.1 Model-Driven Image Cropping

We introduce a strategy to minimize the input size for segmentation networks, focusing on cropping rather than uniform downsampling. This method involves isolating each object within the original high-resolution image, cropping these areas, and performing segmentation on the individual cropped sections. By concentrating on smaller crop regions instead of the entire image, we significantly reduce the network's input size while preserving essential pixel information within the objects. This approach is illustrated in Figure 5.1, providing a visual summary of the process.

The inference procedure using our method is as follows. Let I be an input image of size $W \times H$. We obtain a rough segmentation using $M_{rough} = g_{\phi_1}(C(I))$, where M_{rough} is a binary segmentation mask generated by g_{ϕ_1}, a CNN with input and output size $S \times S$, parameterized by ϕ_1; and C is a uniform downsampling operation from $W \times H$ to $S \times S$.

Given N connected components of M_{rough}, a set of bounding boxes $\{b_i\}$, $i \in [1..N]$ is calculated enclosing each connected component, described in more detail later in this chapter. The bounding boxes are used to generate a set of crops $\{I_i : I_i = I(T_i(b_i))\}$,

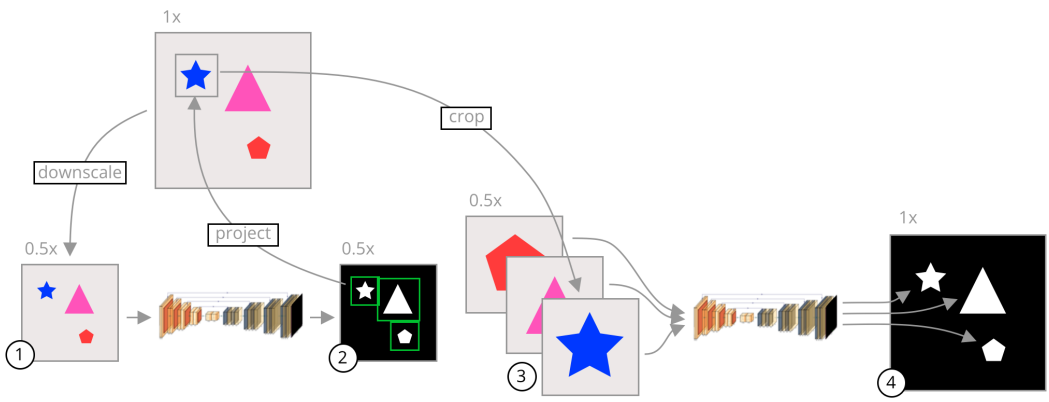


Figure 5.1: A visual summary of our approach. (1) An image is uniformly downsampled from its original resolution. (2) A rough segmentation is predicted by a neural network, and the bounding box of each connected component is calculated. (3) The bounding boxes are scaled to the original image space and crops of the input image are taken in the original resolution and scaled to a common input size. (4) Each crop is segmented separately by a second neural network specifically trained on cropped images. These crops are fused to form a final segmentation in the original high resolution. [Ben+23]

where T_i is a scaling and translation of the bounding box in $S \times S$ space to the corresponding region in the $W \times H$ space. The crops are used to generate a set of fine segmentation masks Y_f using:

$$Y_f = \{Y_{fi} : Y_{fi} = g_{\phi_2}(C_i(I_i)), i \in 1..N\}, \tag{5.3}$$

where g_{ϕ_2} is a CNN of the same architecture as g_{ϕ_1} and size $S \times S$, parameterized by ϕ_2, and C_i is a scaling operation from the width and height of I_i to $S \times S$. A final segmentation y is formed using:

$$y = max(\{(T_i \circ C_i^{-1})(Y_{fi}), i \in 1..N\}), \tag{5.4}$$

where y is the resulting segmentation formed as the maximum value in all of the fine segmentations, transformed to their corresponding regions in $W \times H$ space. This process is described in more detail in Algorithm 1.

The rough segmentation network g_{ϕ_1} is trained on uniformly downsampled images. The network outputs a rough, low-resolution segmentation mask. The rough segmentation mask contains a number of connected components. For each connected component, we calculate a bounding box that encompasses all of its pixels. These bounding boxes are the crop regions used for the fine segmentation network. Since we only use this segmentation to obtain rough regions of interest, the input images to this network can be heavily downsampled without impacting the final fine segmentation.

The fine segmentation network g_{ϕ_2} is trained on cropped images using the ground truth segmentation masks to generate the bounding boxes. This network produces a fine segmentation of that region of the image. Since we know the original bounding box of each crop, we can resize the final segmentation to its original size and translate it to its original position. We perform this for each object in the image, fusing each of the fine segmentation masks into a final segmentation mask in the original image resolution.

In other words, our method performs zooming and panning around the original image and builds a final segmentation piece-wise from each zoom and pan. This allows us to use neural networks with very low input sizes without requiring a large amount of downscaling. What follows is a detailed description of the different parts of the segmentation process.

Cropping

The key to reducing downscaling in our approach is that we take crops from the image in the original resolution. The crop regions themselves are predicted on a downscaled image and then projected to the original image space.

The cropping procedure is as follows. First, a bounding box fully encompassing each connected component in the rough segmentation is calculated. The coordinates of the bounding box are then scaled to the high-resolution image space. An empirically determined padding of $S/8$ (for an $S \times S$ input size) is added along each of the four sides. This way of cropping preserves the context of the object and decreases the number of false negatives inside the bounding box.

The box is then squared, i.e. its height and width are set to the larger of the two dimensions. The box is also shifted (maintaining its width and height) to be fully inside the region of the image. Finally, the bounding box is used as a region to crop the original high-resolution image. The rough segmentation sometimes results in noisy regions, so each crop whose width or height is less than 5 pixels is discarded.

Algorithm 1 Inference algorithm for one input image

Input: High-resolution input image I of size $H \times W$,
input size S, padding k,
neural network NET_1 trained in $S \times S$ downscaled images,
neural network NET_2 trained on ground truth $S \times S$ image crops.
Output: Output image Y of size $H \times W$.

```
I' ← RESIZE(I, (S, S))
y' ← NET_1(I')
y' ← RESIZE(y', (H, W))
ccs ← CONNECTED_COMPONENTS(y')
crops ← [ ]                                   ▷ An array of S × S images
bboxes ← [ ]                                  ▷ An array of bounding boxes for each crop

for cc in ccs do
    bbox ← bounding_box(cc)
    bbox.width ← bbox.height ← MAX(bbox.width, bbox.height)
    bbox ← (bbox.left − k, bbox.top − k, bbox.width + k, bbox.height + k)
    bbox ← SHIFT_TO_IMAGE_REGION(bbox, (H, W))
    crops.ADD(CROP(I, bbox))
    bboxes.ADD(bbox)
end for

for crop, i in crops do
    l, t, w, h ← bboxes[i]
    crop ← RESIZE(crop, (S, S))
    y_crop ← NET_2(crop)
    y_crop ← RESIZE(y_crop, (h, w))
    Y[t : t + h, l : l + w] ← Y[t : t + h, l : l + w] || y_crop
end for
```

Fine Segmentation and Fusion

Each crop of the high-resolution image is scaled to the input size of the fine segmentation network. The fine segmentation network outputs a number of segmentation masks equal to the number of connected components in the rough segmentation. A high-resolution segmentation mask is created by translating and scaling each of the fine segmentation network outputs to their original position. By doing so we construct a full segmentation piece by piece, where each piece is a fine segmentation of a single object in the image, as detected by the rough segmentation. If the cropped regions overlap in the final segmentation, we use a logical `OR` operator to resolve the conflict in the fine segmentation mask. This process is presented in Algorithm 1.

Training the Fine Segmentation Network

The fine segmentation network is trained on ground truth image crops. The crops are obtained by using the connected components of ground truth segmentation masks. From there, the crops are prepared in the same way as described above. Since the images have multiple crop regions, we choose one of the crop regions of the full-resolution image at random during each training iteration. If the original image has no connected components in the ground truth segmentation mask, the whole image is used as training input. All input images to the fine segmentation network are resized to $S \times S$, where S is a predetermined input size that matches the input size used to train the rough segmentation network.

During training, we add an augmentation step by adding random noise to the crop region. When preparing the crops during training, a uniformly distributed random number between ± 16 pixels is added to each dimension of the bounding box (the x- and y-coordinate, width, and height). This ensures that the trained network is robust to imperfect rough segmentation masks during inference.

In our experiments, we used the same architecture for both the rough and fine segmentation networks, as this allows us to use transfer learning. However, there is no requirement that the networks use the same architecture.

5.1.1 Related work

The approach presented in this chapter is motivated partly by the work of Qiu, Qin, and Zhang [QQZ18], where a dataset of images is manually cropped to the object boundary, leading to an increase in segmentation performance. This was applied to skin lesions where it achieved a scale-unifying effect across the dataset. In this chapter, we adapt this approach to use a neural network to predict the optimal object boundary and develop specific ways to train the fine segmentation network on cropped images. In addition, we allow for taking multiple crops on the image and later fusing them in the final segmentation, which makes the method applicable to a wider range of segmentation tasks.

In the previous chapter, a related approach is presented using the polar transform as a pre-processing step. The main novelty in this chapter is the use of cropping as a transformation step. The rest of the approach was then adapted to better suit a cropping transformation, including bounding box augmentation and padding the bounding box. This allows our approach to be used to reduce the input size of the networks.

Detect-then-segment

Several recent end-to-end neural network architectures for segmentation incorporate cropping in one of their layers [Gir+14; He+17]. These generally use object detection

to find a region of interest which is then segmented. This approach can be called *detect-then-segment*. In models such as R-CNN [Gir+14] the objects are first detected and then fed into the segmentation pipeline. Mask R-CNN [He+17] uses object detection to extract a region of interest in the feature masks within the network. These methods effectively concentrate the network on a region of the image. However, information is still lost if the images are uniformly downsampled as a preprocessing step. In contrast, our approach allows one to use low input sizes by cropping the image before it enters the network. Compared to *detect-then-segment* approaches, cropping as a preprocessing step reduces the number of parameters while increasing the pixel-level information in the salient regions of the image. In addition, our approach leads to rescaling each object to the same size before the fine segmentation, which increases the scale-invariance of the models.

Coarse-to-fine Segmentation

Our approach can be described as a coarse-to-fine approach to image segmentation. There have been similar approaches to medical image segmentation. Zhou et al. [Zho+17] describe an approach to pancreas segmentation using a fixed-point model [Li+19]. They train a coarse and a fine segmentation network. The coarse network obtains an initial region of the pancreas which is then re-segmented using the fine network. The fine network then re-segments its output again, and this process is repeated iteratively until a stable solution emerges. They also use bounding box augmentation during training. Our approach differs in two ways. Firstly, we only use one iteration at a stable input size, improving the inference time. Secondly, our approach supports segmenting multiple objects on the image.

Zhu et al. [Zhu+18] describe an approach to pancreas segmentation with two neural networks. The first network is a coarse segmentation network trained on overlapping 3D regions of the whole CT volume. The second network is a fine segmentation network that is trained on only the regions where the ground-truth images contain the pancreas. During inference, the fine network re-segments densely overlapping regions of the rough segmentation. The main difference in our approach is the use of only one region of interest per object where the whole object is visible and uniform in scale. This allows us to use networks of a lower capacity while still maintaining good segmentation results.

Similarly to our approach, Jha et al. [Jha+21] split the segmentation process into detection and segmentation stages. They use a neural network to first detect an object in a downsampled image. They then use the bounding box to crop the object in the high-resolution image. Our approach differs in several ways. Firstly, our approach allows the detection of multiple objects on the image and describes a way to fuse the segmentations of different objects. Secondly, we present new ways to train the fine segmentation network to make the fine segmentation network more robust to imperfect bounding boxes. Finally, we propose a generalized approach evaluated on a variety of different modalities of medical images.

Non-uniform Downsampling

The resolution of an input image for neural networks can be reduced using a more complex sampling strategy. Marin et al. [Mar+19] use non-uniform downsampling for this task. Their approach consists of training a neural network to sample points near object boundaries. The sampling points are then used to downsample the image and perform a final segmentation using a second neural network trained on the downsampled images. Similarly, Jin et al. [Jin+22] use a learnable deformable downsampling module which is

trained together with a segmentation module end-to-end. Our approach differs in the use of cropping instead of non-uniform downsampling, which preserves the topology of the image and provides localization of the object.

Other Approaches to Reducing Input Resolution

Recently, transformer-based architectures such as SegFormer [Xie+21] and the Swin Transformer [Liu+21] have become popular approaches to semantic segmentation. These networks are trained on a large number of small, overlapping patches of the image. The network uses self-attention to determine the saliency of each patch. In a sense, this allows the network to be trained on very small input image dimensions. However, transformers require a very large amount of data to be trained and have large memory requirements [Dos+21], so their use is currently limited in the domain of training on downscaled medical images.

For whole slide images, the input size is often reduced by dividing the image into equally sized patches [NAE18; Hou+16]. A downside of this approach is computational complexity during inference since not all patches are relevant. Additionally, errors can arise when the objects are split by the patch boundary.

5.1.2 Results

We evaluate our approach on three separate datasets, hereafter referred to as the cells, aorta, and polyp datasets. First, for each dataset, we trained a rough segmentation U-Net, Res-U-Net++, and DeepLabv3+ network at various downscaled input resolutions. These models are also used as baseline models to compare against our approach. To evaluate our approach, we train fine segmentation models using the same combinations of datasets, architectures, and input sizes.

Altogether more than 100 neural networks were trained to evaluate our approach, including both the baseline networks and networks trained on cropped images.

Each network is trained from scratch using the downscaled dataset. The outputs from the networks are then upscaled to the datasets' original resolution, and the metrics are calculated using those outputs. We use the held-out test datasets for all of the results reported in this section. The baseline models are used as the rough segmentation networks for the experiments using our approach. The hyperparameters used for each network are reported in Table 5.1.

In the interest of providing objective metrics of model performance, all of the hyperparameters were tuned using the validation dataset on the baseline U-Net. Those same hyperparameters are then used for each of the models in our approach. Each model is trained using the Adam optimizer up to a maximum number of epochs and the best model with the best validation loss is saved during training. The validation loss is calculated as the Dice score coefficient (DSC) over the validation dataset at the same resolution as the input images. We do not upscale the outputs for the validation loss as we do for calculating the final metrics. Each model was trained using PyTorch 1.10 on an Nvidia GeForce RTX 3080 GPU. Where possible, we have fixed the random seed value an arbitrarily chosen value of "2022", but we have also run the experiments on two other random seeds and obtained similar results. Other hyperparameters were chosen based on validation DSC.

Table 5.1 The hyper-parameters used for each of the models in our experiments.

Dataset	Batch size	Learning rate	Max. epochs
Cells	16	$5 \cdot 10^{-4}$	100
Polyp	8	10^{-3}	175
Aorta	8	10^{-3}	100

Datasets

This section briefly describes the datasets used in our experiments as well as the preprocessing steps for the images. For more details, we direct readers to the supplemental code repository available at github.com/marinbenc/segment-then-segment. To evaluate our approach, we chose three datasets across different medical imaging modalities, including CT scans, microscopy imaging, and colonoscopy images. We hope that the variety in the datasets will show the generalizability of our approach. Aside from the variety, the datasets were selected because they include images of large dimensions on which small objects of various sizes need to be segmented. These types of tasks are most likely to suffer from the loss of information due to downscaling and are thus particularly suitable to be segmented using our approach.

Aorta Dataset: For aorta segmentation we use the AVT dataset [Rad+22], a multi-center dataset of labeled CTA scans of the aortic vessel tree. We only use a subset of the dataset from Dongyang Hospital, a total of 18 scans of between 122 and 251 slices. Each slice is windowed to 200 to 500 HU, normalized to $[-0.5, 0.5]$, and zero-centered by subtracting 0.1 from each slice. The original resolution of the slices is 512×666 pixels. We use augmentation during training. Each input has a 50% chance of an affine transform (translation of $\pm 6.25\%$, scaling of $\pm 10\%$, rotation of $\pm 14°$), as well as a 30% chance of a horizontal flip. The dataset is split per patient into a training set (12 scans), validation set (3 scans), and test set (3 scans).

Cells Dataset: For cell nucleus segmentation we use the 2018 Data Science Bowl dataset, otherwise known as image set BBBC038v1 from the Broad Bioimage Benchmark Collection [Cai+19]. We use 670 RGB images and their corresponding labels from the `stage1_train` repository. The original files are of various sizes ranging from 256×256 to 1024×1024 pixels. We did not further split the images into patches, all training is done on the whole images. We use the same augmentation as for the aorta dataset. The dataset is split into a training set (80%, 536 images), validation set (10%, 67 images), and test set (10%, 67 images).

Polyp Dataset: For polyp segmentation, we use the Kvasir-SEG dataset [Jha+20], which contains 1000 annotated gastroscopy images containing polyps. The size of the original images ranges from 332 to 1920 pixels in width. We split the dataset into train (80%, 800 images), validation (10%, 100 images), and test (10%, 100 images) datasets.

Quantitative Assessment

The results of our experiments are shown in Table 5.2. Our approach, using low input sizes, results in segmentation DSC on par or better than baseline models trained on larger input sizes using downscaled images. This is especially apparent at large downscaling factors of 4 and more, where the baseline models quickly deteriorate in their performance but our models were still able to achieve results that are close to those on full-size images. The biggest improvement is seen in terms of recall. For instance, on the 4x downscaled cells dataset, our approach leads to a 4.3 times larger recall. Likewise, for the polyp images, the recall improves from 0.039 to 0.799, a 20 times improvement.

Table 5.2 The results of our approach using U-Net as the underlying architecture. The U-Net results are based on uniformly downscaled images.

	DSC [%]	IoU [%]	Prec. [%]	Rec. [%]
Cells (256 × 256)				
64 × 64 - U-Net	75.70 ± 11.82	62.13 ± 13.29	75.27 ± 12.61	76.79 ± 12.70
64 × 64 - Ours	84.28 ± 12.83	74.50 ± 15.45	84.75 ± 14.85	86.22 ± 13.25
128 × 128 - U-Net	84.52 ± 10.34	74.33 ± 13.05	85.21 ± 11.61	84.66 ± 11.61
128 × 128 - Ours	84.54 ± 9.64	74.26 ± 12.73	83.97 ± 13.00	86.86 ± 9.39
256 × 256 - U-Net	87.62 ± 13.55	79.75 ± 15.46	89.44 ± 14.09	86.58 ± 14.83
256 × 256 - Ours	84.91 ± 12.65	75.24 ± 13.92	83.52 ± 14.50	87.53 ± 13.76
Polyp (256 × 256)				
64 × 64 - U-Net	81.76 ± 18.81	72.40 ± 21.00	83.94 ± 21.36	84.84 ± 17.30
64 × 64 - Ours	83.82 ± 19.91	75.90 ± 22.51	86.32 ± 21.87	86.72 ± 17.84
128 × 128 - U-Net	82.86 ± 20.72	74.75 ± 23.32	83.30 ± 22.73	88.93 ± 17.50
128 × 128 - Ours	81.36 ± 26.57	74.58 ± 27.52	82.49 ± 27.55	85.23 ± 25.33
256 × 256 - U-Net	83.80 ± 16.08	74.88 ± 20.44	87.91 ± 19.29	84.37 ± 16.55
256 × 256 - Ours	84.68 ± 19.40	77.11 ± 22.53	87.10 ± 20.91	87.42 ± 18.41
Aorta (256 × 256)				
128 × 128 - U-Net	81.03 ± 14.45	70.13 ± 17.02	85.09 ± 14.14	78.30 ± 15.04
128 × 128 - Ours	88.50 ± 11.08	80.73 ± 13.93	92.42 ± 10.33	86.02 ± 12.25
256 × 256 - U-Net	72.30 ± 28.69	62.97 ± 28.74	91.21 ± 25.39	63.80 ± 29.11
256 × 256 - Ours	80.10 ± 25.18	72.12 ± 25.74	86.75 ± 24.14	76.39 ± 26.51
512 × 512 - U-Net	89.34 ± 14.59	83.10 ± 18.26	96.03 ± 11.52	85.66 ± 17.44
512 × 512 - Ours	85.51 ± 14.80	76.85 ± 17.17	90.90 ± 14.15	82.39 ± 15.54

Recall is especially important in medical image segmentation since the cost of false negatives can often far outweigh the cost of false positives.

We evaluate how general our approach is by applying it to two other state-of-the-art semantic segmentation architectures, Res-U-Net++ [Jha+19; Zho+18] and DeepLabv3+ [Che+18]. The results of these experiments are shown in Table 5.3.

Furthermore, our approach achieves much better stability of results as the input size decreases. This is shown visually in Figure 5.2. The stability improvements are especially visible in Figure 5.3, where it can be seen that the distribution of the results from our approach remains more stable than in the baseline models.

The goal of our approach is increasing segmentation performance on downscaled images, so we do not expect a performance increase on the full-size images. While our approach offers significant improvements when using downscaled images, the main disadvantage of our approach is that it requires training two separate neural networks. However, this downside is lessened by two key factors. First, the cropped networks converge much faster since the objects are already localized and unified in scale in the images, making the problem easier for the network to learn. Secondly, since the architecture of the two networks is identical, one can use transfer learning from the trained rough segmentation network to the fine segmentation network.

Table 5.3 A comparison of the Dice Score Coefficients of our approach using other underlying architectures at 4x and 2x downscaled images. The Baseline model is trained on uniformly downscaled images.

Model	Size	Baseline [%]	Ours [%]
Cells (256 × 256)			
Res-U-Net++	64 × 64	75.18 ± 11.85	86.63 ± 10.06
Res-U-Net++	128 × 128	84.74 ± 9.64	86.90 ± 11.77
DeepLabv3+	64 × 64	66.93 ± 15.93	80.41 ± 16.80
DeepLabv3+	128 × 128	81.43 ± 13.74	84.34 ± 16.91
Polyp (256 × 256)			
Res-U-Net++	64 × 64	79.92 ± 19.72	81.35 ± 20.82
Res-U-Net++	128 × 128	84.23 ± 17.74	83.34 ± 20.73
DeepLabv3+	64 × 64	76.62 ± 21.78	77.53 ± 24.56
DeepLabv3+	128 × 128	83.65 ± 18.86	85.02 ± 19.37
Aorta (512 × 512)			
Res-U-Net++	128 × 128	81.59 ± 14.43	88.21 ± 13.25
Res-U-Net++	256 × 256	88.21 ± 13.19	86.11 ± 14.02
DeepLabv3+	128 × 128	76.30 ± 22.81	86.73 ± 16.94
DeepLabv3+	256 × 256	87.71 ± 12.13	89.28 ± 11.75

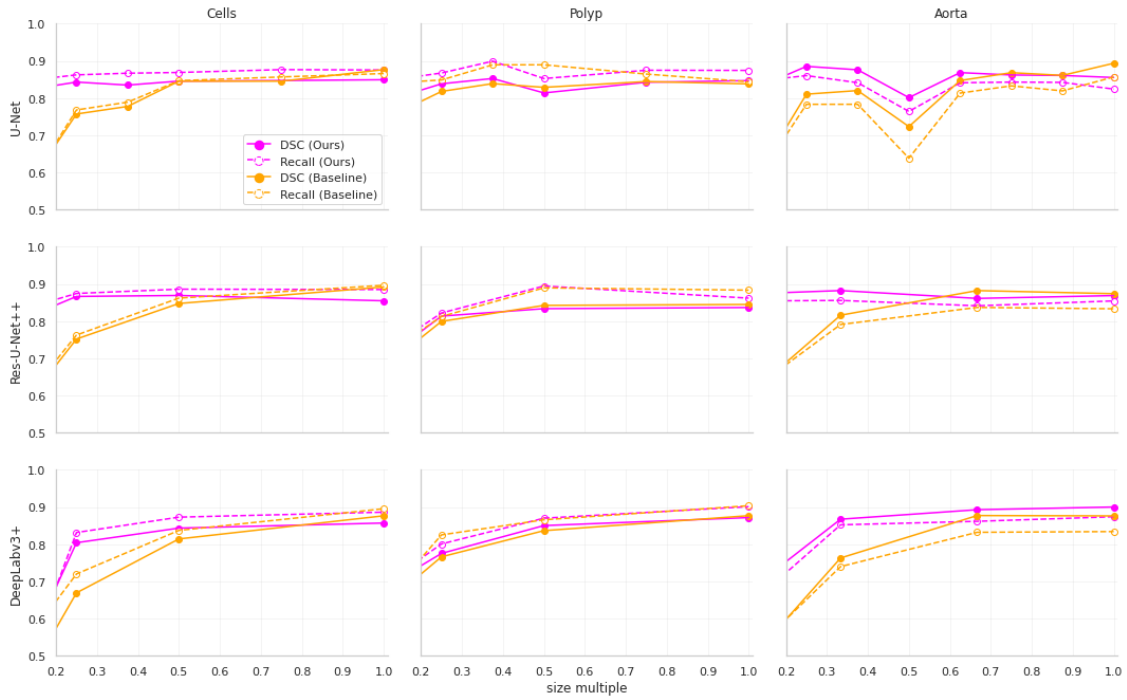


Figure 5.2: The relationship between input dimensions and the mean Dice Score Coefficient (DSC) and recall for different datasets. The points are measured values from our experiments. The Baseline model is trained on uniformly downscaled images.

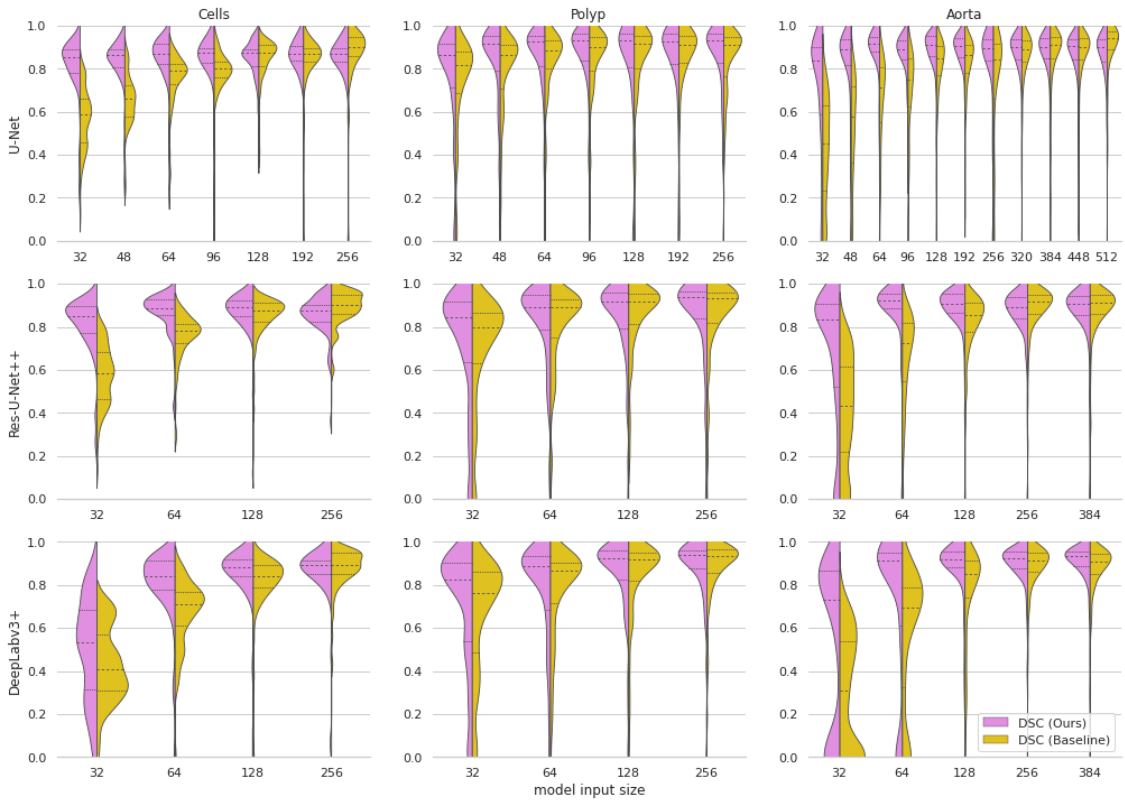


Figure 5.3: Violin plots of Dice Score Coefficients of our approach compared to the baseline models trained on uniformly downscaled images at various input dimensions. The dashed lines represent quartiles of the distributions.

Qualitative Assessment

Qualitatively, there is a large improvement when using our approach over the baseline methods on downscaled inputs. At large downscaling factors, outputs from the baseline models often include artifacts on the border since the pixel grid is not fine enough to represent small variations on the object boundary. These issues disappear with our approach, as the overall downscaling amount is much lower. This effect is especially visible on the cells dataset due to its relatively small object size, as shown in Figure 5.4. We observe a similar effect on the aorta dataset, shown in Figure 5.6.

On the polyp dataset, our approach greatly reduces the number of false negative pixels on the image, as can be seen in Figure 5.5. U-Net fails to predict the whole polyp region on small input sizes, leading to ragged object boundaries and holes in the predicted regions. By comparison, our approach produces smoother, closed contours and manages to capture more of the object boundary than U-Net equivalents at the same input size.

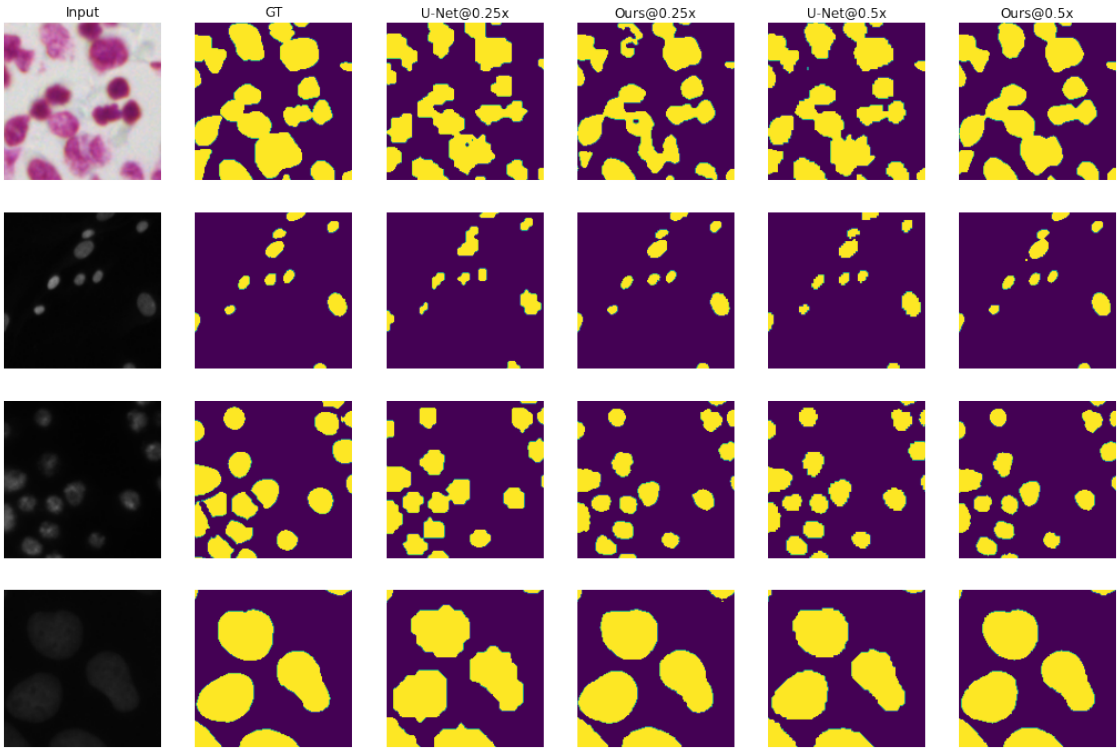


Figure 5.4: Example outputs from the models for the cells dataset at various input sizes. [Ben+23]

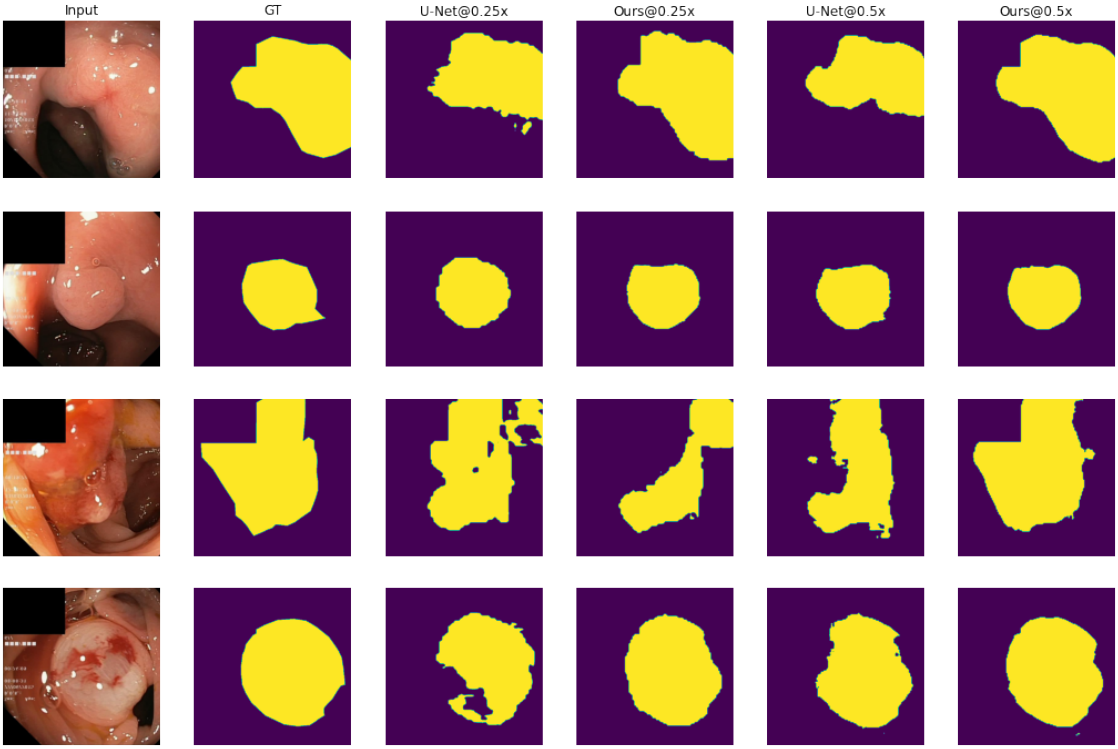


Figure 5.5: Example outputs from the models for the polyp dataset at various input sizes. [Ben+23]

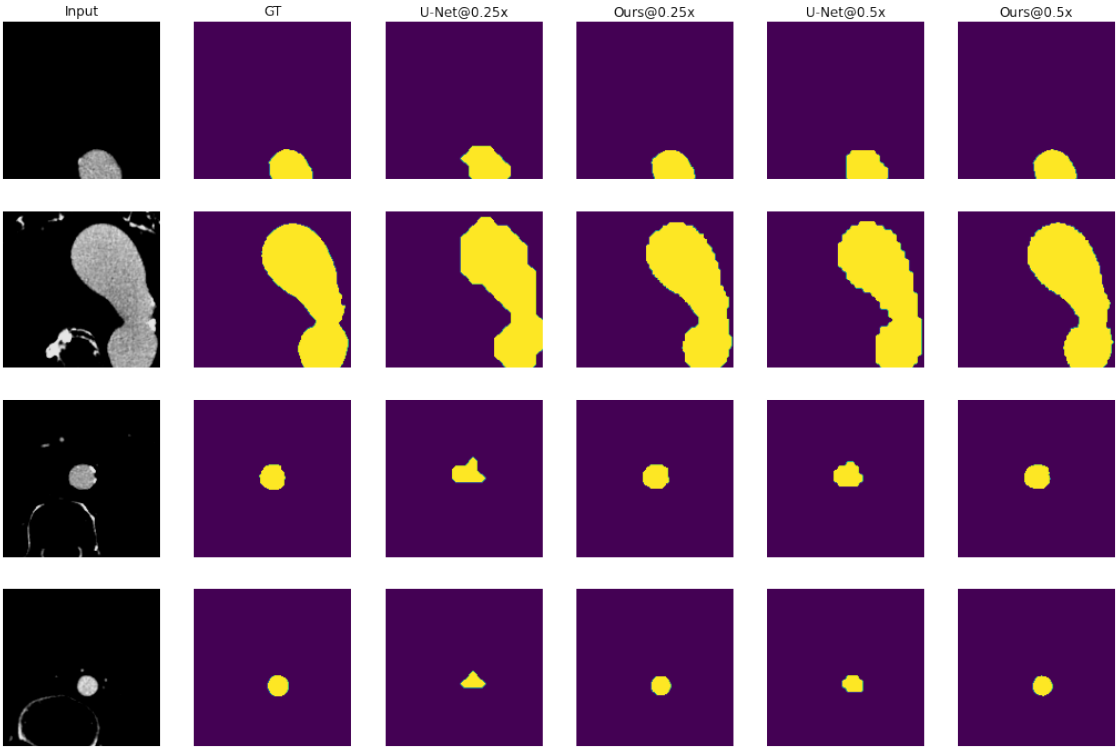


Figure 5.6: Example outputs from the models for the aorta dataset at various input sizes. [Ben+23]

Computational Performance Characteristics

Since our approach consists of using a cascade of two U-Nets, the number of parameters of the network is doubled. However, the peak GPU memory utilization during training

and inference remains exactly the same when using our approach as with a baseline model on the same input size. Our approach allows one to reduce the input size while still maintaining the same segmentation metrics, thus allowing larger batch sizes during training. This is presented in Table 5.4.

In terms of computational performance, the largest downside of our approach is that it increases inference time non-linearly with the size of the images, which can be seen in Figure 5.7.

This increase is explained by two key features of our approach. Firstly, cropping and rescaling operations take up a large percentage of the time. Secondly, each connected component in the initial segmentation is processed separately by the second network, thus inference time increases with the number of objects to be segmented. This is most apparent in the cells dataset, as the input images have the largest number of objects. Note that we use the implementation of `torch.nn.functional.interpolate` with the nearest-neighbor mode in Pytorch 1.10 for all resizing. The inference time depends on

Table 5.4 Performance characteristics of our approach compared to the baseline model with similar mean test Dice Score Coefficients.

	Input size	DSC	Peak VRAM[1,3]	Inf. time[2,3]	Max. batch size[3]
Cells					
U-Net	64^2	75.70	1.9 GB	24 ms	300
Seg-Then-Seg	64^2	84.28	1.9 GB	122 ms	300
U-Net	128^2	84.52	2.3 GB	24 ms	80
Polyp					
U-Net	48^2	78.42	2.9 GB	16 ms	850
Seg-Then-Seg	48^2	81.68	2.9 GB	29 ms	850
U-Net	128^2	82.86	3.8 GB	16 ms	150
Aorta					
U-Net	128^2	81.03	2.9 GB	13 ms	46
Seg-Then-Seg	128^2	88.50	2.9 GB	26 ms	46
U-Net	512^2	89.34	10.1 GB	16 ms	9

[1] Measured using a batch size of 8 for all rows.

[2] Mean inference time across all inputs in the test set, calculated per slice for the aorta dataset.

[3] Measured using PyTorch 1.10 on an Nvidia GeForce RTX 3080 GPU and an AMD Ryzen 7 3700x 8-core CPU with 32 GB of RAM.

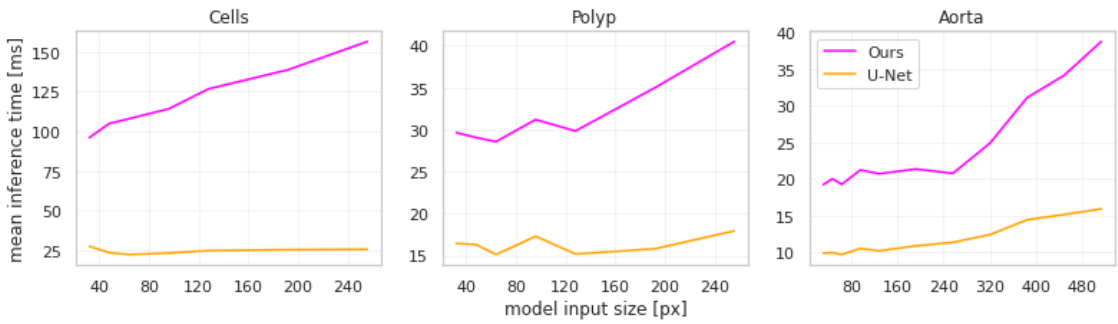


Figure 5.7: Mean per-input inference time across different input sizes for the U-Net-based models. [Ben+23]

the specific implementation of the resizing algorithm as well as the hardware it is being run on.

While these increases seem large in relative terms, it should be noted that in absolute terms the increases are in the order of magnitude of tens of milliseconds. We argue that such an increase in inference time does not limit the applicability of our approach in practical use cases.

Note that the goal of our method is not to produce state-of-the-art segmentation results on high-resolution images. Instead, the goal is to allow training on heavily downscaled images without sacrificing segmentation performance.

This initial version of the approach presents a promising avenue for future research. One way to improve the approach is to use a single end-to-end trainable network. We describe this kind of extension in the next section.

5.2 An End-to-End Extension of Model-Driven Image Cropping

So far, both in this chapter as well as the previous chapter, we employed separate networks for initial rough segmentation and subsequent fine segmentation. Instead of using separate neural networks, connecting the two networks end-to-end and training them as a single architecture presents distinct advantages. Training the rough and fine segmentation modules together could enhance the rough segmentation's ability to identify more accurate regions of interest, while the fine segmentation module could improve its robustness to imperfect initial segmentations.

A straightforward approach might involve chaining the two networks: an input image first undergoes rough segmentation, followed by fine segmentation of the transformed image, with both subnetworks trained simultaneously. This naive approach has several challenges. Firstly, since the network now needs to perform two segmentations, network convergence is much harder as there is no straightforward flow of gradients from the output to the input of the network. Secondly, the transformation procedure is potentially non-differentiable, and thus there could be no gradient calculated between the output layer of one subnetwork and the input layer of the other subnetwork.

To address these issues, we suggest two key modifications for a viable end-to-end model. Firstly, the output from the rough segmentation module is incorporated as an additional input channel to the fine segmentation module, facilitating gradient flow and providing context about the object's location. Secondly, we pre-train both subnetworks so they are already capable of segmenting images before they are fine-tuned together. This drastically reduces the chance that the end-to-end network will not converge. What follows is a more detailed description of the end-to-end network architecture.

5.2.1 End-to-end Network Architecture and Training

As illustrated in Figure 5.8, the network consists of two stages, a coarse and fine segmentation stage, connected by a cropping layer. Both stages use the same U-Net-based segmentation architecture. The input size of each stage is fixed and smaller than the original image resolution. Assuming the input size of the network is $w \times h$, given a high-resolution input image I' of size $W \times H$, we first uniformly downscale the image into I of size $w \times h$ using linear interpolation. This image is then input into the coarse segmentation module producing a segmentation mask $M_{coarse}(I)$.

The segmentation mask M_{coarse} is then upscaled to M'_{coarse} of size $W \times H$ and thus matches the high-resolution image. Then, a bounding box is calculated such that it encompasses all non-zero regions of M'_{coarse} and expanded by 20 pixels in each direction,

clipped to the bounds of the image. The bounding box is used to crop I' and the crop is then scaled to $w \times h$, producing a high-resolution input region of interest I_{crop}. Note that scaling the crop region to $w \times h$ results in a much lower total scaling than it does when scaling down the whole image. In cases where M_{coarse} has no non-zero pixels, the bounding box is set to the whole image, i.e. $I_{crop} = I$. From our experiments, such cases are exceedingly rare.

Then, an input vector is constructed for the fine segmentation network as $x_{fine} = [I_{crop}, M_{coarse}]$ — in other words, the first channel is the high-resolution crop, while the second channel is the coarse segmentation mask. The fine segmentation network then produces a final segmentation mask $M(x_{fine})$ which is used as the output of the whole model.

We use pre-training to allow the two networks to reliably converge. First, we train a regular segmentation network on uniformly downscaled images I. Once trained, the weights of this model are transferred both to the rough and fine segmentation modules of our proposed network architecture.

5.2.2 Experiments in Clinical Dermatological Image Segmentation

We validate this end-to-end architecture on segmenting the skin lesion boundary in clinical dermatological images using datasets of small sample sizes. Namely, we use the University of Waterloo skin cancer database [GWC14] (available at [12b]) of clinical skin lesion images. The dataset consists of manual segmentation labels of two publicly available skin lesion databases, DermIS [12a] (69 images) and DermQuest (137 images, no longer available). We treat these two collections as two different centers to evaluate out-of-sample performance. The images are of various sizes from 300 to 550 pixels in height and width. As a preprocessing step, each image of each dataset is normalized to $[-0.5, 0.5]$.

We use PyTorch 1.10 for all experiments. All models are trained on an Nvidia GeForce RTX 3080. Wherever possible, we fix an arbitrarily chosen random seed value of "2022" to increase reproducibility. We use the Adam optimizer for all training while reducing the learning rate on plateaus by a factor of 0.1. The best model is selected and saved during training based on validation loss. We use a learning rate of 10^{-4} for both pre-training and fine-tuning. We set the batch size to 16 for all input sizes except 512×512, where the batch size was set to 4. The aforementioned hyperparameters were

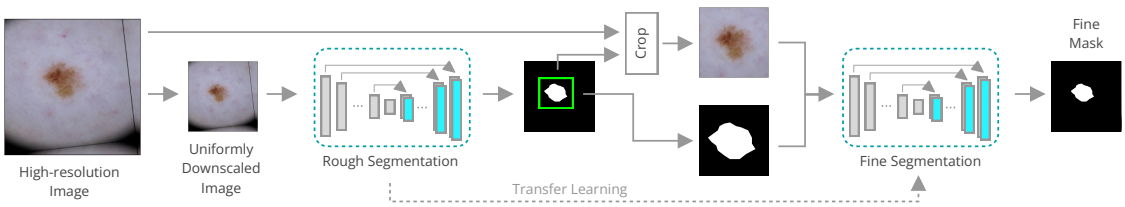


Figure 5.8: An illustration of the end-to-end model-driven cropping architecture, comprising two interconnected modules: a coarse and a fine segmentation module, linked via a cropping layer. Both modules are designed to handle images of small, fixed input sizes. The first module processes a downscaled version of the input image. Its segmentation output determines the region of interest in the high-resolution image, which, along with the cropped initial segmentation, is fed into the fine segmentation module. Before being fine-tuned together, both networks are pre-trained as standard segmentation networks.

chosen based on the validation DSC of the models. For ease of implementation, all high-resolution input images are resized to 1024×1024 pixels, but this is not a required step for our method, which can work with any high-resolution input image size.

Results and Discussion

To evaluate our method, we train a baseline U-Net model and our end-to-end image-cropping approach for input sizes 64×64, 128×128, 256×256, and 512×512. The same architecture was used for the baseline U-Net model as well as for the rough and fine segmentation modules in our method. We employed 5-fold cross-validation to evaluate in-sample performance, resulting in five distinct models for each dataset. For out-of-sample performance evaluation, each image in one dataset was segmented using the five models trained on the other dataset. Segmentation metrics were calculated for each prediction and then averaged, ensuring that all reported out-of-sample metrics represent an average from models trained across five different data splits.

The two segmentation metrics we use for the evaluation are the Dice Similarity Coefficient (DSC) as well as the Thresholded Jaccard Index. DSC takes into account both the accuracy and recall of the method and is thus a good general segmentation metric. The Thresholded Jaccard Index was proposed for lesion segmentation in the ISIC 2018 challenge [Cod+18] to account for specific needs in skin lesion segmentation. Low-accuracy segmentations (in this case defined as those with a Jaccard Index less than 0.65) are not useful for further lesion analysis. Therefore, instead of a simple mean result calculation, all results with a Jaccard Index below 0.65 are set to zero. This has been shown to better represent true segmentation performance in the context of lesion segmentation.

Visually, our approach demonstrates enhanced performance in delineating lesion boundaries, as evident in Figure 5.9. The segmentation results more accurately adhere to the ground truth contours and exhibit fewer false-positive regions compared to those generated by the baseline models.

Quantitatively, in-sample and out-of-sample results of our model compared to a baseline U-Net are shown in Table 6.1. When comparing our method and a baseline U-Net using the same input size, we note an increase in DSC in almost all experiments. We see similar results for the Thresholded Jaccard Index. These increases can give us an estimate of how much we can reduce the input size of an image while still retaining the same segmentation quality. For instance, using our approach, we can go from 256×256 to 64×64 images while achieving the same in-sample as well as out-of-sample Thresholded Jaccard Index for DermIS.

The thresholded Jaccard index assumes that any segmentation with a Jaccard index below 0.65 is a failed segmentation, i.e. it cannot be used for any downstream analysis of the legion due to its insufficient accuracy. Consequently, we examined the frequency of these failed segmentations across various input sizes, as depicted in Figure 5.10. The findings indicate a general improvement in the mean Jaccard Index (exceeding 0.65) when utilizing our model, particularly for smaller input sizes, and this trend persists even in out-of-sample evaluations. Additionally, there is a noticeable decrease in the incidence of failed segmentations (Jaccard Index < 0.65) in all but two experiments. This decline is most pronounced with smaller input sizes, suggesting that our method significantly enhances the robustness of segmentations derived from images with small input sizes.

Figure 5.9: Randomly chosen examples of out-of-sample segmentation results. The columns, from left to right, show the input image with the ground truth segmentation mask, the final output segmentation mask of our approach, the rough segmentation of our approach, and the output of a baseline U-Net model. The images are zoomed in on the lesion region.

Table 5.5 Results of our approach and the baseline model in terms of the Dice Coefficient (DSC) as well as the thresholded Jaccard index (Th. Jacc.) for various experiments. Results are shown in the form of mean ± standard deviation. The top two groups show in-sample performance within the DermIS and DermQuest datasets, while the bottom two groups show out-of-sample performance when trained on DermIS and tested on DermQuest, and vice-versa.

	U-Net		Ours	
Size	**DSC**	**Th. Jacc.**	**DSC**	**Th. Jacc.**
	DermIS			
64	85.51 ± 15.50	70.31 ± 31.38	87.01 ± 15.06	74.57 ± 27.97
128	86.53 ± 15.71	72.21 ± 30.78	88.82 ± 11.93	75.84 ± 28.41
256	88.09 ± 14.58	73.58 ± 33.01	89.88 ± 12.49	77.04 ± 30.77
512	89.85 ± 16.90	81.32 ± 26.32	90.35 ± 17.07	83.40 ± 24.35
	DermQuest			
64	87.75 ± 9.07	71.61 ± 29.56	87.83 ± 8.97	73.96 ± 25.84
128	88.83 ± 10.19	76.59 ± 25.76	89.45 ± 10.09	77.93 ± 25.00
256	89.61 ± 11.20	77.34 ± 27.73	90.10 ± 11.22	78.99 ± 26.18
512	91.14 ± 9.18	81.31 ± 23.69	90.52 ± 12.98	81.16 ± 24.76
	DermIS → DermQuest			
64	77.11 ± 13.58	43.40 ± 34.37	80.94 ± 12.85	54.56 ± 31.45
128	80.20 ± 12.92	50.81 ± 33.87	79.61 ± 14.15	52.46 ± 32.36
256	78.67 ± 15.32	51.10 ± 33.30	80.07 ± 14.55	55.18 ± 31.51
512	80.96 ± 14.38	60.73 ± 26.96	83.34 ± 13.29	66.72 ± 24.61
	DermQuest → DermIS			
64	78.94 ± 18.81	54.98 ± 34.36	81.85 ± 17.24	60.87 ± 32.66
128	78.31 ± 19.36	51.57 ± 37.63	81.51 ± 18.23	60.10 ± 35.86
256	81.10 ± 19.54	60.81 ± 35.41	81.97 ± 19.46	63.12 ± 34.89
512	83.60 ± 18.73	65.59 ± 32.99	82.82 ± 19.70	64.50 ± 33.35

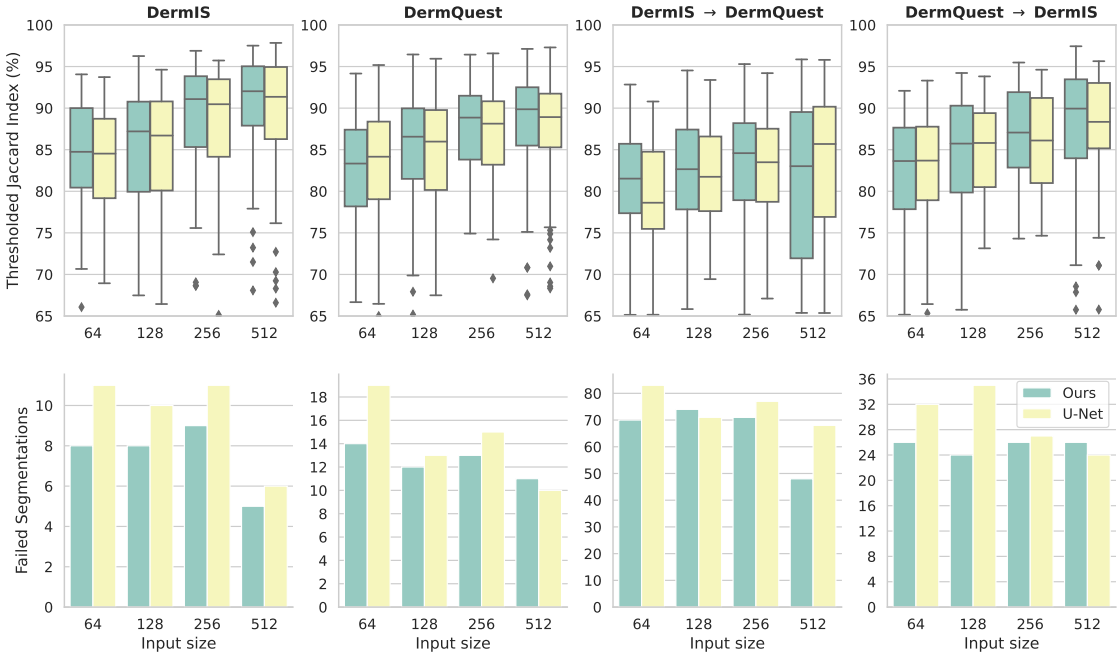


Figure 5.10: A box plot of Jaccard indices above or equal to 0.65 (above) and the number of Jaccard indices below 0.65 (below). The first two columns show in-sample results, while the last two columns show out-of-sample results for models trained on DermIS and tested on DermQuest and vice-versa.

5.3 Conclusion

Downscaling is a common source of segmentation errors in neural networks. In this chapter, we presented an approach to training neural networks that reduces downscaling by utilizing two neural networks and salient crops. We show how training a second neural network on cropped image regions can improve segmentation performance on small input sizes with few downsides. Our approach improves segmentation metrics on downscaled images across different modalities and image sizes, especially in terms of recall. We show that, while this approach increases inference time, it allows for training using much larger batch sizes while maintaining the same segmentation metrics. By utilizing transfer learning, we successfully addressed issues related to network convergence and provided an end-to-end extension of this method. Using our approach, we were able to reduce input sizes by at least half while retaining the same segmentation quality.

A significant advantage of our approach is the reduced requirement for detailed segmentation labels in the rough segmentation stage. This stage only necessitates approximate object localization, achievable through simpler, potentially unsupervised methods like heuristic-based and traditional image processing techniques. Moreover, one can use weakly supervised training using pseudo-labels, instead of segmentation masks. While we did not evaluate our approach on 3D networks in this chapter, there is nothing in our approach that is specific to 2D images. Our approach can be extended to 3D images by using 3D neural network architectures and 3D bounding boxes for crop regions.

However, our approach has certain limitations. Notably, from our experiments in datasets with large sample sizes, our method did not substantially improve segmentation quality. We speculate that our approach acts as a form of regularization, beneficial in scenarios where overfitting is a concern due to small sample sizes and high-capacity networks. This regularization may be redundant in cases with ample sample sizes. Nonetheless, the primary aim of our approach is to improve data efficiency in terms of both input size and sample size.

We believe that this method can be broadly applied to enhance segmentation performance and robustness in a variety of image segmentation challenges, particularly where object scale and location variance are significant and the sample size is small. The methods presented in this chapter were published in [Ben+23] and [BHG24].

References

[12a] *DermIS — Dermis.Net, Dept. of Clinical Social Medicine (Univ. of Heidelberg) and the Dept. of Dermatology (Univ. of Erlangen)*. https://www.dermis.net/. 2012. (Visited on 01/23/2024).

[12b] *Skin Cancer Detection | Vision and Image Processing Lab — Uwaterloo.ca, University of Waterloo*. https://uwaterloo.ca/vision-image-processing-lab/research-demos/skin-cancer-detection. 2012. (Visited on 01/23/2024).

[Ben+23] Marin Benčević, Yuming Qiu, Irena Galić, and Aleksandra Pižurica. "Segment-Then-Segment: Context-Preserving Crop-Based Segmentation for Large Biomedical Images". In: *Sensors* 23.2 (Jan. 2023), p. 633. ISSN: 1424-8220. DOI: 10.3390/s23020633.

[BHG24] Marin Benčević, Marija Habijan, and Irena Galić. "Crop-Guided Neural Network Segmentation of High-Resolution Skin Lesion Images". In: *MOSTART2024: International Conference on Digital Transformation in Education and Artificial Intelligence Applications*. Mostar, Bosnia and Herzegovina, Apr. 2024.

[Cai+19] Juan C. Caicedo, Allen Goodman, Kyle W. Karhohs, Beth A. Cimini, Jeanelle Ackerman, et al. "Nucleus Segmentation across Imaging Experiments: The 2018 Data Science Bowl". In: *Nature Methods* 16.12 (Dec. 2019), pp. 1247–1253. ISSN: 1548-7105. DOI: 10.1038/s41592-019-0612-7. (Visited on 09/15/2022).

[Che+18] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *Computer Vision – ECCV 2018*. Ed. by Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss. Vol. 11211. Cham: Springer International Publishing, 2018, pp. 833–851. ISBN: 978-3-030-01233-5 978-3-030-01234-2. DOI: 10.1007/978-3-030-01234-2_49. (Visited on 10/25/2022).

[Cod+18] Noel C. F. Codella, David Gutman, M. Emre Celebi, Brian Helba, Michael A. Marchetti, et al. "Skin Lesion Analysis toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)". In: *2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)*. Washington, DC: IEEE, Apr. 2018, pp. 168–172. ISBN: 978-1-5386-3636-7. DOI: 10.1109/ISBI.2018.8363547. (Visited on 08/23/2021).

[Dos+21] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, et al. "An Image Is Worth 16x16 Words: Transformers for Image Recognition at Scale". In: *ICLR* (2021).

[Gir+14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *2014 IEEE Conference on Computer Vision and Pattern Recognition*. Columbus, OH, USA: IEEE, June 2014, pp. 580–587. ISBN: 978-1-4799-5118-5. DOI: 10.1109/CVPR.2014.81. (Visited on 10/11/2022).

[GWC14] J Glaister, A Wong, and D A. Clausi. "Automatic Segmentation of Skin Lesions from Dermatological Photographs Using a Joint Probabilistic Texture Distinctiveness Approach". In: *IEEE Transactions on Biomedical Engineering* (2014).

[He+17] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. "Mask R-CNN". In: *2017 IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017, pp. 2980–2988. DOI: 10.1109/ICCV.2017.322.

[Hou+16] Le Hou, Dimitris Samaras, Tahsin M. Kurc, Yi Gao, James E. Davis, et al. "Patch-Based Convolutional Neural Network for Whole Slide Tissue Image Classification". In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2016, pp. 2424–2433. DOI: 10.1109/CVPR.2016.266.

[Jha+19] Debesh Jha, Pia H. Smedsrud, Michael A. Riegler, Dag Johansen, Thomas De Lange, et al. "ResUNet++: An Advanced Architecture for Medical Image Segmentation". In: *2019 IEEE International Symposium on Multimedia (ISM)*. San Diego, CA, USA: IEEE, Dec. 2019, pp. 225–2255. ISBN: 978-1-72815-606-4. DOI: 10.1109/ISM46123.2019.00049. (Visited on 10/25/2022).

[Jha+20] Debesh Jha, Pia H Smedsrud, Michael A Riegler, Pål Halvorsen, Thomas de Lange, et al. "Kvasir-Seg: A Segmented Polyp Dataset". In: *International Conference on Multimedia Modeling*. Springer. 2020, pp. 451–462.

[Jha+21] Aadarsh Jha, Haichun Yang, Ruining Deng, Meghan E. Kapp, Agnes B. Fogo, et al. "Instance Segmentation for Whole Slide Imaging: End-to-End or Detect-Then-Segment". In: *Journal of Medical Imaging* 8.1 (Jan. 2021), p. 014001. ISSN: 2329-4302. DOI: 10.1117/1.JMI.8.1.014001. (Visited on 09/19/2022).

[Jin+22] Chen Jin, Ryutaro Tanno, Thomy Mertzanidou, Eleftheria Panagiotaki, and Daniel C. Alexander. "Learning to Downsample for Segmentation of Ultra-High Resolution Images". In: *International Conference on Learning Representations*. 2022.

[Li+19] Quannan Li, Jingdong Wang, David Wipf, and Zhuowen Tu. "Fixed-Point Model for Structured Labeling". In: *Proceedings of the 30th International Conference on Machine Learning*. Ed. by Sanjoy Dasgupta and David McAllester. Vol. 28. Proceedings of Machine Learning Research. Atlanta, Georgia, USA: PMLR, 2013-06-17/2013-06-19, pp. 214–221.

[Liu+21] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, et al. "Swin Transformer: Hierarchical Vision Transformer Using Shifted Windows". In: *arXiv preprint arXiv:2103.14030* (2021). arXiv: 2103.14030.

[Mar+19] Dmitrii Marin, Zijian He, Peter Vajda, Priyam Chatterjee, Sam Tsai, et al. "Efficient Segmentation: Learning Downsampling Near Semantic Boundaries". In: *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2019, pp. 2131–2141. DOI: 10.1109/ICCV.2019.00222.

[NAE18] Kamyar Nazeri, Azad Aminpour, and Mehran Ebrahimi. "Two-Stage Convolutional Neural Network for Breast Cancer Histology Image Classification". In: *Image Analysis and Recognition*. Ed. by Aurélio Campilho, Fakhri Karray, and Bart ter Haar Romeny. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2018, pp. 717–726. ISBN: 978-3-319-93000-8. DOI: 10.1007/978-3-319-93000-8_81.

[QQZ18] Yuming Qiu, Xiaolin Qin, and Ju Zhang. "Training FCNs Model with Lesion-Size-Unified Dermoscopy Images for Lesion Segmentation". In: *2018 International Conference on Artificial Intelligence and Big Data (ICAIBD)*. IEEE, May 2018. DOI: 10.1109/icaibd.2018.8396187.

[Rad+22] Lukas Radl, Yuan Jin, Antonio Pepe, Jianning Li, Christina Gsaxner, et al. "AVT: Multicenter Aortic Vessel Tree CTA Dataset Collection with Ground Truth Segmentation Masks". In: *Data in Brief* 40 (Feb. 2022), p. 107801. ISSN: 23523409. DOI: 10.1016/j.dib.2022.107801. (Visited on 05/04/2022).

[SB14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. 1st ed. Cambridge University Press, May 2014. ISBN: 978-1-107-05713-5 978-1-107-29801-9. DOI: 10.1017/CBO9781107298019. (Visited on 12/01/2023).

[TL20] Mingxing Tan and Quoc V. Le. *EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks*. Sept. 2020. arXiv: 1905.11946 [cs, stat]. (Visited on 09/16/2022).

[Xie+21] Enze Xie, Wenhai Wang, Zhiding Yu, Anima Anandkumar, Jose M Alvarez, et al. "SegFormer: Simple and Efficient Design for Semantic Segmentation with Transformers". In: *Neural Information Processing Systems (NeurIPS)*. 2021.

[Zho+17] Yuyin Zhou, Lingxi Xie, Wei Shen, Yan Wang, Elliot K. Fishman, et al. "A Fixed-Point Model for Pancreas Segmentation in Abdominal CT Scans". In: *Medical Image Computing and Computer Assisted Intervention - MICCAI 2017*. Ed. by Maxime Descoteaux, Lena Maier-Hein, Alfred Franz, Pierre Jannin, D. Louis Collins, et al. Vol. 10433. Cham: Springer International Publishing, 2017, pp. 693–701. ISBN: 978-3-319-66181-0 978-3-319-66182-7. DOI: 10.1007/978-3-319-66182-7_79. (Visited on 12/19/2022).

[Zho+18] Zongwei Zhou, Md Mahfuzur Rahman Siddiquee, Nima Tajbakhsh, and Jianming Liang. "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Ed. by Danail Stoyanov, Zeike Taylor, Gustavo Carneiro, Tanveer Syeda-Mahmood, Anne Martel, et al. Vol. 11045. Cham: Springer International Publishing, 2018, pp. 3–11. ISBN: 978-3-030-00888-8 978-3-030-00889-5. DOI: 10.1007/978-3-030-00889-5_1. (Visited on 10/25/2022).

[Zhu+18] Zhuotun Zhu, Yingda Xia, Wei Shen, Elliot Fishman, and Alan Yuille. "A 3D Coarse-to-Fine Framework for Volumetric Medical Image Segmentation". In: *2018 International Conference on 3D Vision (3DV)*. Verona: IEEE, Sept. 2018, pp. 682–690. ISBN: 978-1-5386-8425-2. DOI: 10.1109/3DV.2018.00083. (Visited on 12/19/2022).

6 Adding Depth Information to 2D U-Nets for CT Segmentation

In the previous two chapters, we discussed two specific methods of model-driven preprocessing and their extensions. In this chapter, we will introduce a preprocessing technique designed to incorporate depth information into 2D images, aiming to bridge the gap between 2D and 3D image segmentation. CT scans, being 3D images, are often segmented with 3D neural networks. However, directly segmenting 3D scans poses two key challenges:

Firstly, 3D scans are naturally very large, as instead of processing one e.g. 256×256 slice, the network now needs to process a $256 \times 256 \times 256$ volume. This dramatically increases memory requirements and restricts the maximum batch size. In extreme cases where processing an entire scan is infeasible, the data is either partitioned into smaller patches [Zhu+18], downsampled with focused re-segmentation on regions of interest [Ise+21], or segmented using semi-3D networks [Wen+20].

Secondly, the scarcity of large 3D datasets exacerbates the risk of overfitting, as we have established that using large images necessitates the use of high-capacity models. Segmenting individual slices, on the other hand, can act as a form of regularization, creating multiple samples from a single subject and allowing for the use of smaller networks.

Therefore, there is a need for methods that can effectively integrate depth information into 2D segmentation networks, maintaining the technical benefits of a 2D network while enhancing them with depth perception.

One potential solution to incorporate depth information into 2D segmentation networks is to condition the network's output on the slice depth by adding the slice depth coordinate as an additional input feature. This would give the network more information about the expected segmentation shape as a function of slice depth both globally and individually for that slice. However, integrating such scalar information into a fully convolutional network like U-Net is not straightforward.

This problem has been solved in the field of image generation, where the condition is added as a separate input channel. For an image $I \in \mathbb{R}^{W \times H \times C}$ and a condition vector x, the image is augmented to include $|x|$ additional channels. Each new channel $c(i) \in \mathbb{R}^{W \times H}$ is constructed so that every pixel $c(i)_{xy}$ equals $x(i)$ for all x, y coordinates [MO14]. This approach is illustrated in Figure 6.1.

Drawing inspiration from this, we propose a new and simple way of incorporating 3D information into 2D fully-convolutional neural networks for medical image segmentation. Consider a 3D scan comprising D slices:

$$I = \{I_d(x, y) \mid I_d(x, y) \in R^{W \times H}, d \in [1..D]\}. \tag{6.1}$$

For every slice I_d, we create a depth information channel $C_d(x, y) = d/D$ for all x, y, i.e. each pixel is equal to the normalized z-coordinate of that slice. The network's input for each slice becomes $I'_d = [I_d, C_d]$ where the first channel is I_d and the second channel contains the depth information.

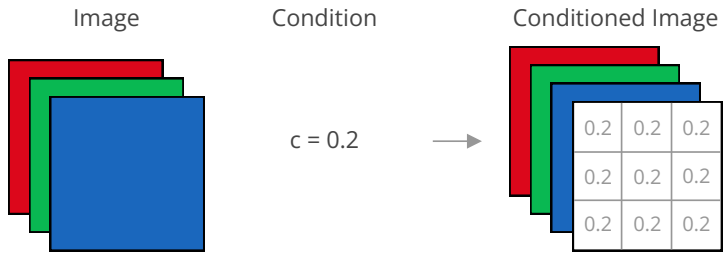


Figure 6.1: An illustration of conditioning an image by adding a channel to the image where each pixel is equal to the condition value.

This method enables 2D networks to utilize depth cues without significant modifications to their architecture or an increase in parameter count, maintaining compatibility with transfer learning from 2D datasets. Throughout the rest of this chapter, we evaluate this technique on the specific task of epicardial adipose tissue (EAT) segmentation.

6.1 Epicardial Adipose Tissue Segmentation

Epicardial adipose tissue (EAT) is a type of adipose tissue located within the pericardium, a protective layer around the heart. EAT lies between the myocardium (the heart muscle) and the fibrous outer layer of the pericardium. Due to EAT's proximity to the myocardium, it is believed to be a metabolically active organ that has a direct impact on various cardiovascular diseases.

For instance, EAT has been shown to play a direct role in coronary atherosclerosis and cardiomyopathy [SF07; MA13]. EAT thickness has been shown to correlate with metabolic syndrome [Che+09] and coronary artery disease independently of obesity [Iac+11]. It relates in general with the progression of coronary artery calcification [Gor+08]. Additionally, the volume and density of EAT have been linked to major adverse cardiac events in asymptomatic subjects [Goe+18]. Additionally, EAT plays a role in insulin resistance, is an accurate therapeutic target, and impacts heart morphology and adiposity [Iac09].

EAT's active role in cardiovascular diseases makes measuring its volume and thickness an important diagnostic tool. Currently, EAT is most often quantified by measuring EAT thickness using echocardiography. This estimation of the volume from the thickness at a single point can introduce inaccuracies and inter-observer variability. Using 3D medical imaging technologies such as CT results in more accurate measurements, however, the availability of CT machines as well as the procedure's cost and duration make it impractical for common clinical use. Another downside of using CT to measure EAT is the time required to manually measure EAT volume, which can take up to an hour per patient [Mil+19] and is prone to inter-observer variability of around 10% [MA13] of the measured volume. Fully or semi-automatic methods of EAT segmentation and quantification from CT images could reduce time requirements and the cost of EAT quantification.

Segmenting EAT is a challenging image-processing task. Its uneven distribution around the heart, peculiar shape, and its similarity to other adipose tissues nearby complicate the process. Segmenting EAT relies on delineating the pericardium, which is less than 2 mm thin and can often be hard to delineate on CT images due to partial volume effects.

6.2 Related Work

Incorporating depth information into 2D segmentation networks often involves training on preselected regions of interest, where multi-channel images are constructed from sequential slices of a 3D volume [Wen+20]. These multi-slice images serve as input for conventional 2D networks. Our approach diverges by directly supplying the z-coordinate as an additional input channel. Similar strategies are employed in U-Net-based architectures in image generation, such as Imagen [Sah+22]. We suggest that a similar method of incorporating supplementary information could improve the data efficiency of image segmentation tasks.

There are several semi-automatic and fully automatic methods for EAT segmentation proposed in the existing literature. One of the first methods of EAT segmentation was proposed by Coppini [Cop10]. They use a semi-automatic approach where an expert places control points on the pericardium which are used to segment the pericardium using thresholding and geodesic active contours. An improved semi-automatic segmentation method is presented by Militello et al. [Mil+19]. Their method requires expert input to define the VoI on a few slices of the scan, and the rest of the VoI is then interpolated, saving time while still offering manual-level segmentation accuracy.

Ding et al. [Din+14] propose a method similar to Coppini [Cop10] but instead of relying on manual initialization, they initialize the pericardium contour using an atlas-based method achieving fully automatic segmentation. Rodrigues et al. [Rod+16] propose a fully automatic machine-learning-based method. Their method consists of extracting hand-selected salient features from the CT slices, which are then segmented by a learned random forest classifier.

Commandeur et al. [Com+18] proposed one of the first EAT segmentation methods to use deep learning. They use a multi-task approach consisting of two convolutional neural networks to classify whether a slice contains the heart and segment the pericardium as well as EAT. More recent works propose deep learning methods based on variations of the U-Net architecture. Li, Zou, and Yang [LZY19] use a U-Net with a pyramid pooling structure, while He et al. [He+20] use a 3D-based U-Net with an added attention mechanism. Zhang et al. [Zha+20] use two stacked U-Net-based networks. The first network segments a pericardium region, which is then refined using morphological operators. This region is used as a mask for the input into the other U-Net, which segments the EAT region. While similar, our approach differs from this work in several ways. We only use one U-Net-based neural network, significantly reducing the number of parameters. Additionally, we modify the input data to allow the network to utilize slice-depth information during training and inference.

6.3 Dataset Description

We use a dataset of CT scans of 20 patients from Rio de Janeiro obtained and released publicly by Rodrigues et al. [Rod+16]. The original ground truth was obtained via manual segmentation of 878 total slices by a physician and a computer scientist. The slices were also registered, scaled, and cropped to a similar anatomical region, as well as thresholded to the adipose tissue range of $[-200, -30]$ HU. Of the 20 patients, 10 are male and 10 are female. The mean age of the patients is 55.4. Each scan has an average of 42.95 slices with a slice thickness of 3 mm. The CT scans were obtained with two different scanners, 9 patients were scanned by a Phillips scanner, while the other 11 were scanned by a Siemens scanner.

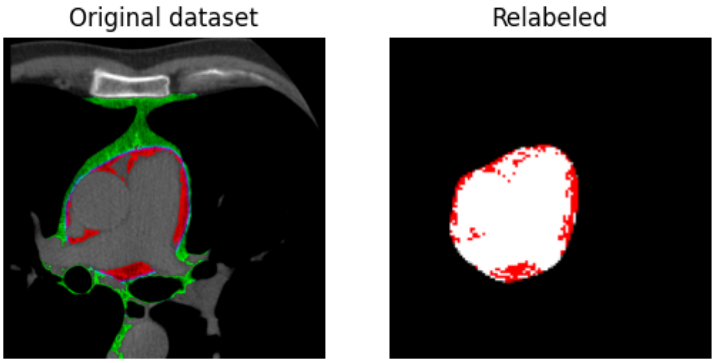


Figure 6.2: An example of the original image label and our relabeled image. The EAT label is shown in red, while the pericardium label is shown in white. [BHG21]

The original labels contain three classes: pericardium, EAT, and paracardial adipose tissues. Using these labels as a reference, we manually labeled a closed pericardium region on each slice. During labeling, we follow the pericardium line where available and mark the pericardium region as the border separating the two adipose tissues when a label is not visible. We then create a separate set of EAT labels by multiplying the input images, thresholded to the range of adipose tissue, with the pericardium label. I.e. we define EAT as all adipose tissue masked by the pericardium label. A comparison of the original label and our re-labeled images is shown in Figure 6.2.

6.4 Methodology

We achieve segmentation of the pericardium using a deep neural network architecture based on U-Net [RFB15]. The input to the model is a 2-channel 128×128 image. The first channel is a single slice of a patient's CT scan. The second channel, as described earlier in the chapter, represents the slice depth of that slice. The pericardium region varies highly by slice depth. Therefore, we utilize depth information by first normalizing the slice depth of each input slice to a value between 0 and 1. We then create a 128×128 image where each pixel has the value of the slice depth. This image is used as the second channel of the neural network input. This allows the network to utilize additional depth information without changing the underlying architecture. Example input images are shown in Figure 6.3.

We use a loss function which is a modified version of the Dice coefficient:

$$DSC_{loss} = 1 - \frac{2|X \cap Y| + \lambda}{|X| + |Y| + \lambda}, \tag{6.2}$$

Figure 6.3: A sample of the inputs to the neural network from a single patient, sorted by slice depth. The first channel (CT adipose tissue) is shown in full white, while the second channel (the slice depth) is shown from black (highest z-axis) to green (lowest z-axis). [BHG21]

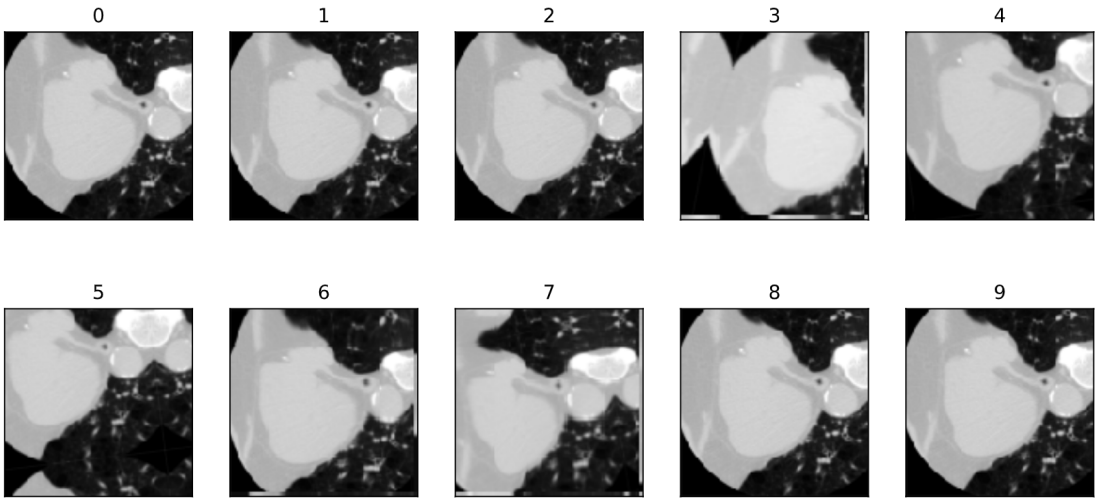


Figure 6.4: Different random augmentation examples of the same input image. [BHG21]

where X and Y are the input and predicted images, respectively, and λ is a smoothing parameter set to 1 in our experiments.

6.4.1 Data Preprocessing

We apply several data preprocessing steps on the original dataset to achieve better segmentation results. First, the images are all normalized and zero-centered by subtracting the global mean intensity of all pixels of the training dataset (0.1). The images are then scaled down from 512×512 to 128×128 pixels. We also removed a total of 112 slices from the original dataset that did not include EAT labels, either because of labeling errors in the original dataset or because the slices were outside the heart region.

Furthermore, we utilize heavy data augmentation to increase model generalizability. In real-world distributions, anatomical structures can differ drastically from one person to the next. To simulate these differences, we add a random chance of augmenting each input image during the training phase. Each input image has a 50% chance for a horizontal flip, and a 30% chance of a random combination of affine image transformations, including (1) a translation of max. 6.25% of the image's width, (2) a scaling of max. 10% of the image's scale, and (3) a rotation of a max. 45 degrees. Additionally, each input image has a 20% chance of a non-linear mesh deform. Examples of augmented images are shown in Figure 6.4.

6.4.2 Model Training

The neural network is implemented and trained using PyTorch 1.7.1 on an NVIDIA GeForce RTX 3080 GPU. The networks were trained for 200 epochs, with a checkpoint mechanism after each epoch. We select the model with the best validation loss during training. In our experiments, the model converged around epoch 100 after 5 minutes. The training was done using the Adam optimizer with a learning rate of 0.001. The used batch size was 8. These hyperparameters were chosen based on the resulting validation Dice Similarity Coefficient of the model. We arbitrarily choose a manual random seed value of 42 for all experiments to increase reproducibility. We use 2-fold per-patient cross-validation for all experiments. Each model was trained on the slices of 10 patients and validated on the remaining 10 patients.

6.5 Experiments and Results

The results in this section are evaluated on the two cross-validation folds and averaged across folds. Segmentation quality is evaluated using the Dice Similarity Coefficient (DSC) and the Jaccard Index. We analyze the model's quantification quality with a Bland-Altman analysis [MA86] as well as with the Pearson correlation coefficient. To calculate the Pearson correlation and to perform the Bland-Altman analysis, we threshold all segmented pericardium regions to obtain the EAT segmentation results and calculate the number of pixels labeled as EAT on each image. We also calculate the number of EAT pixels on the ground truth images. The pixel counts are used as a proxy for volume measurement, as the EAT volume of a patient is proportional to the number of segmented EAT pixels in the patient's CT scan.

Our method's mean DSC and correlation results are presented in Table 6.1. The network provides better results for pericardium segmentation than EAT segmentation, which is expected given the pericardium's smoother shape. The DSC for EAT indicates that our method performs slightly worse inter-observer variability, but still achieves good segmentation in a fraction of the time needed for manual segmentation. The precision and recall have similar values both for EAT and the pericardium, showing a good potential for future use to measure EAT volume. Examples of predictions from our model are presented in Figure 6.5.

The Bland-Altman analysis of our method is presented in Figure 6.6. Bland-Altman analysis is a common way of visualizing the agreement between two clinical measuring methods. It consists of a scatterplot where the x-axis is the average and the y-axis is the

Table 6.1 Mean results of the cross-validation. The Corr. value is the Pearson correlation between the total number of EAT pixels for each slice of the validation dataset ($p < 0.0001$).

Target	DSC	Jaccard	Prec.	Rec.	Corr.
Pericardium	0.9264	0.8819	0.9319	0.9345	-
EAT	0.8646	0.7807	0.8787	0.8690	0.8864

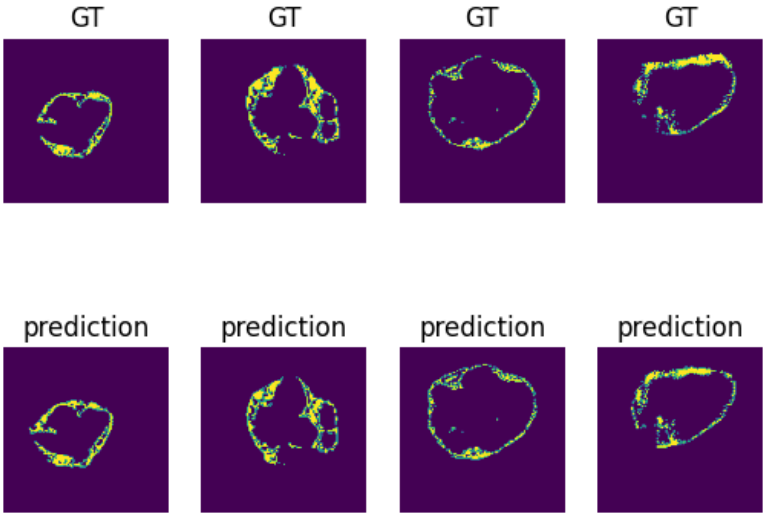


Figure 6.5: Examples of EAT predictions compared to the ground truth images. [BHG21]

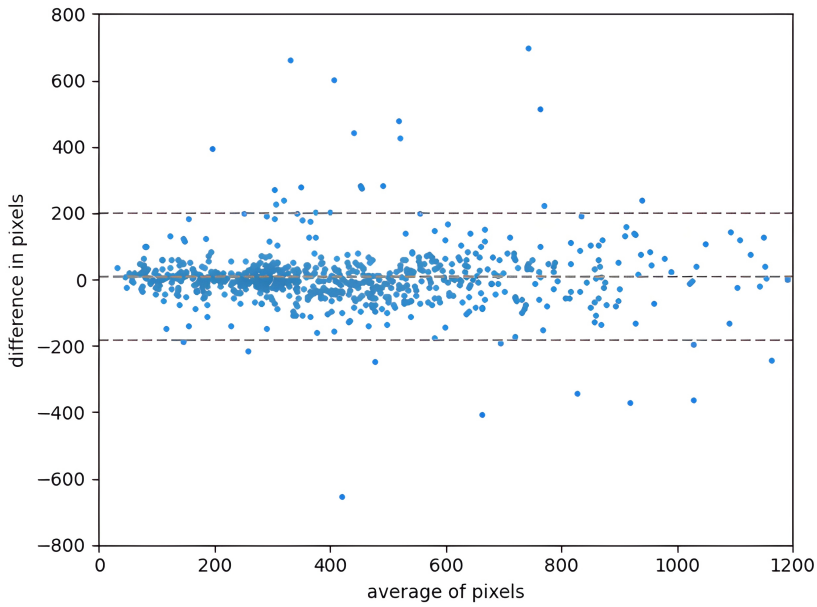


Figure 6.6: The Bland-Altman analysis of the number of pixels predicted as EAT on each slice of the test dataset for each fold. The dashed lines indicate a 95% confidence interval. [BHG21]

Table 6.2 A comparison of our approach with other deep-learning-based approaches for EAT segmentation.

Method	DSC	Jaccard	Prec.	Rec.	Params
U-Net for EAT	0.75	0.58	0.72	0.69	5.8 M
Zhang et al. [Zha+20]	**0.91**	**0.84**	-	-	11.6 M
He et al. [He+20]	0.85	-	0.86	0.89	6.4 M
Our method	0.86	0.78	0.89	0.87	5.8 M

difference between two measurements. In this case, we show the average between the ground truth and the predicted measure of the number of epicardial fat pixels in a slice of a CT scan. The analysis shows a high level of agreement between our method and ground-truth annotations. The plot also shows a small positive bias from our method. Additionally, the plot does not show a strong proportional bias. However, while most measurements fall within the 95% confidence interval, there are outliers. The most likely reason for these errors is the noisy nature of the ground truth labels especially towards the edges of the heart region.

Additionally, we compare our results to using a U-Net to directly segment the pericardium, as well as with other state-of-the-art approaches This comparison is presented in Table 6.2.

6.6 Conclusion

Our approach achieved a mean Dice Similarity Coefficient (DSC) of 0.8574 and a correlation coefficient of 0.8864. Although these figures fall short of the state-of-the-art, it is important to note that our dataset underwent relabeling to minimize noise and errors,

making direct comparisons to results from the original dataset challenging. In addition, our method uses a more direct neural network with fewer preprocessing steps and fewer parameters than existing deep-learning methods ([Com+18; LZY19; He+20]), leading to lower inference times. We demonstrate the effectiveness of segmenting the pericardium region as a proxy for directly segmenting epicardial adipose tissue (EAT), leveraging the simpler task of learning the smooth contour of the pericardium compared to the more complex, irregular distribution of EAT. Furthermore, we explore incorporating depth information into an encoder-decoder neural network without resorting to 3D volume patch training. By adding slice depth as an extra channel we have improved segmentation performance without changing the underlying architecture and while maintaining the ability to use transfer learning from 2D datasets. The method presented in this chapter was published in [BHG21].

References

[BHG21] Marin Benčević, Marija Habijan, and Irena Galić. "Epicardial Adipose Tissue Segmentation from CT Images with A Semi-3D Neural Network". In: *2021 International Symposium ELMAR*. Zadar, Croatia: IEEE, Sept. 2021, pp. 87–90. ISBN: 978-1-66544-437-8. DOI: 10.1109/ELMAR52657.2021.9550936.

[Che+09] Onn Chenn, Ijaz Ahmad, Betty Hua, Joshua A Sockolow, Igor Klem, et al. "Correlation of Pericardial and Mediastinal Fat with Coronary Artery Disease, Metabolic Syndrome, and Cardiac Risk Factors". In: *Journal of Cardiovascular Magnetic Resonance* 11.S1 (Jan. 2009). DOI: 10.1186/1532-429x-11-s1-o16.

[Com+18] Frederic Commandeur, Markus Goeller, Julian Betancur, Sebastien Cadet, Mhairi Doris, et al. "Deep Learning for Quantification of Epicardial and Thoracic Adipose Tissue from Non-Contrast CT". In: *IEEE Transactions on Medical Imaging* 37.8 (Aug. 2018), pp. 1835–1846. DOI: 10.1109/tmi.2018.2804799.

[Cop10] Giuseppe Coppini. "Quantification of Epicardial Fat by Cardiac CT Imaging". In: *The Open Medical Informatics Journal* 4.1 (July 2010), pp. 126–135. DOI: 10.2174/1874431101004010126.

[Din+14] Xiaowei Ding, Demetri Terzopoulos, Mariana Diaz-Zamudio, Daniel S. Berman, Piotr J. Slomka, et al. "Automated Epicardial Fat Volume Quantification from Non-Contrast CT". In: *Medical Imaging 2014: Image Processing*. Ed. by Sebastien Ourselin and Martin A. Styner. SPIE, Mar. 2014. DOI: 10.1117/12.2043326.

[Goe+18] Markus Goeller, Stephan Achenbach, Mohamed Marwan, Mhairi K. Doris, Sebastien Cadet, et al. "Epicardial Adipose Tissue Density and Volume Are Related to Subclinical Atherosclerosis, Inflammation and Major Adverse Cardiac Events in Asymptomatic Subjects". In: *Journal of Cardiovascular Computed Tomography* 12.1 (Jan. 2018), pp. 67–73. DOI: 10.1016/j.jcct.2017.11.007.

[Gor+08] Petra M. Gorter, Alexander M. de Vos, Yolanda van der Graaf, Pieter R. Stella, Pieter A. Doevendans, et al. "Relation of Epicardial and Pericoronary Fat to Coronary Atherosclerosis and Coronary Artery Calcium in Patients Undergoing Coronary Angiography". In: *The American Journal of Cardiology* 102.4 (Aug. 2008), pp. 380–385. DOI: 10.1016/j.amjcard.2008.04.002.

[He+20] Xiuxiu He, Bang Jun Guo, Yang Lei, Tonghe Wang, Tian Liu, et al. "Automatic Epicardial Fat Segmentation in Cardiac CT Imaging Using 3D Deep Attention U-Net". In: *Medical Imaging 2020: Image Processing*. Ed. by Ivana Išgum and Bennett A. Landman. Vol. 11313. SPIE / International Society for Optics and Photonics, 2020, pp. 589–595. DOI: 10.1117/12.2550383.

[Iac+11] Gianluca Iacobellis, Eva Lonn, Andre Lamy, Navneet Singh, and Arya M. Sharma. "Epicardial Fat Thickness and Coronary Artery Disease Correlate Independently of Obesity". In: *International Journal of Cardiology* 146.3 (Feb. 2011), pp. 452–454. DOI: 10.1016/j.ijcard.2010.10.117.

[Iac09] Gianluca Iacobellis. "Epicardial and Pericardial Fat: Close, but Very Different". In: *Obesity* 17.4 (Apr. 2009), pp. 625–625. DOI: 10.1038/oby.2008.575.

[Ise+21] Fabian Isensee, Paul F. Jaeger, Simon A. A. Kohl, Jens Petersen, and Klaus H. Maier-Hein. "nnU-Net: A Self-Configuring Method for Deep Learning-Based Biomedical Image Segmentation". In: *Nature Methods* 18.2 (Feb. 2021), pp. 203–211. ISSN: 1548-7091, 1548-7105. DOI: 10.1038/s41592-020-01008-z.

[LZY19] Zhiquan Li, Lin Zou, and Ran Yang. "A Neural Network-Based Method for Automatic Pericardium Segmentation". In: *Proceedings of the 2nd International Conference on Computer Science and Software Engineering*. CSSE 2019. New York, NY, USA: Association for Computing Machinery, 2019, pp. 45–49. ISBN: 978-1-4503-7172-8. DOI: 10.1145/3339363.3339372.

[MA13] Mohamed Marwan and Stephan Achenbach. "Quantification of Epicardial Fat by Computed Tomography: Why, When and How?" In: *Journal of Cardiovascular Computed Tomography* 7.1 (Jan. 2013), pp. 3–10. DOI: 10.1016/j.jcct.2013.01.002.

[MA86] J. Martin Bland and DouglasG. Altman. "Statistical Methods for Assessing Agreement between Two Methods of Clinical Measurement". In: *The Lancet* 327.8476 (1986), pp. 307–310. ISSN: 0140-6736. DOI: 10.1016/S0140-6736(86)90837-8.

[Mil+19] Carmelo Militello, Leonardo Rundo, Patrizia Toia, Vincenzo Conti, Giorgio Russo, et al. "A Semi-Automatic Approach for Epicardial Adipose Tissue Segmentation and Quantification on Cardiac CT Scans". In: *Computers in Biology and Medicine* 114 (Nov. 2019), p. 103424. DOI: 10.1016/j.compbiomed.2019.103424.

[MO14] Mehdi Mirza and Simon Osindero. *Conditional Generative Adversarial Nets*. Nov. 2014. arXiv: 1411.1784 [cs, stat]. (Visited on 02/14/2024).

[RFB15] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV].

[Rod+16] É.O. Rodrigues, F.F.C. Morais, N.A.O.S. Morais, L.S. Conci, L.V. Neto, et al. "A Novel Approach for the Automated Segmentation and Volume Quantification of Cardiac Fats on Computed Tomography". In: *Computer Methods and Programs in Biomedicine* 123 (Jan. 2016), pp. 109–128. DOI: 10.1016/j.cmpb.2015.09.017.

[Sah+22] Chitwan Saharia, William Chan, Saurabh Saxena, Lala Li, Jay Whang, et al. *Photorealistic Text-to-Image Diffusion Models with Deep Language Understanding*. May 2022. arXiv: 2205.11487 [cs]. (Visited on 01/30/2024).

[SF07] Harold S. Sacks and John N. Fain. "Human Epicardial Adipose Tissue: A Review". In: *American Heart Journal* 153.6 (June 2007), pp. 907–917. DOI: 10.1016/j.ahj.2007.03.019.

[Wen+20] Junhao Wen, Elina Thibeau-Sutre, Mauricio Diaz-Melo, Jorge Samper-González, Alexandre Routier, et al. "Convolutional Neural Networks for Classification of Alzheimer's Disease: Overview and Reproducible Evaluation". In: *Medical Image Analysis* 63 (July 2020), p. 101694. ISSN: 13618415. DOI: 10.1016/j.media.2020.101694. (Visited on 01/30/2024).

[Zha+20] Qi Zhang, Jianhang Zhou, Bob Zhang, Weijia Jia, and Enhua Wu. "Automatic Epicardial Fat Segmentation and Quantification of CT Scans Using Dual U-nets with a Morphological Processing Layer". In: *IEEE Access* 8 (2020), pp. 128032–128041. DOI: 10.1109/access.2020.3008190.

[Zhu+18] Zhuotun Zhu, Yingda Xia, Wei Shen, Elliot Fishman, and Alan Yuille. "A 3D Coarse-to-Fine Framework for Volumetric Medical Image Segmentation". In: *2018 International Conference on 3D Vision (3DV)*. Verona: IEEE, Sept. 2018, pp. 682–690. ISBN: 978-1-5386-8425-2. DOI: 10.1109/3DV.2018.00083. (Visited on 12/19/2022).

7 Conclusion

Recent advances in neural network-based image segmentation have largely been achieved by leveraging very large datasets to train high-capacity neural networks. There has been comparatively less research in improving performance using small neural networks on datasets with few samples. Rather than relying on foundation models that offer good initial segmentation on unseen datasets but lack the precision necessary for medical purposes, our work emphasizes the importance of domain-specific knowledge to develop smaller, more precise networks tailored to specific medical imaging tasks. These networks are not only more suitable for medical applications but also more manageable for fine-tuning and evaluation on standard computing hardware due to their reduced complexity.

Our methodology integrates traditional image processing techniques with neural networks to simplify the segmentation process. We employ small neural networks for initial object localization. Using hand-crafted functions, optimal image transformation parameters are estimated from the initial segmentation. The image transformations themselves are chosen to be beneficial for the specific task based on domain knowledge from both medicine and image processing. As a result, downstream neural networks trained on the transformed images require fewer parameters for effective segmentation, enhancing their data efficiency.

We have investigated this approach in a wide variety of contexts in medical imaging. Initially, we focused on the polar transform to simplify the segmentation boundary of elliptical shapes commonly seen in organs, vessels, and skin lesions. By employing a neural network to determine the optimal origin for the polar transform, we enhanced segmentation accuracy in CT scans, colonoscopies, and dermatoscopic images, achieving state-of-the-art results for lesion and polyp segmentation. We also showed that we can reduce the sample size and still achieve good segmentation performance.

Furthermore, we have evaluated image cropping both for improving segmentation as well as reducing image input sizes in segmentation convolutional neural networks. By cropping a high-resolution image to a region of interest predicted by a neural network, we can segment only the relevant portion of the image, maintaining fine details while reducing the total input size. Since convolutional neural network complexity grows with image input size, this approach can improve data efficiency by reducing model complexity. This concept was further developed into a unified end-to-end trainable model, improving the robustness of both the initial localization and detailed segmentation processes.

Our approach offers a versatile framework for enhancing data efficiency in medical image segmentation, opening up numerous avenues for future research to refine and expand these methods. One promising direction is to reformulate the image transformations we've employed into differentiable functions, allowing them to be integrated into the neural network. This would enable the network to learn image features beneficial for estimating the transformation parameters, potentially improving the results.

Additionally, there is significant potential in leveraging unlabeled data to improve the data efficiency of our approach. Given that the estimation of image transformation parameters depends only on a rough localization of the target object, there is a reduced

need for precise segmentation labels for this stage. This opens up the possibility of training the localization network with weaker forms of annotations, such as bounding boxes or heatmaps derived from neuron activations, which are less granular than segmentation masks but can still provide valuable spatial information about the object's location.

Furthermore, the reliance on hand-crafted functions to estimate transformation parameters based on object localization could be eliminated by employing a neural network to directly predict the optimal parameters for transformation. We have already taken steps in this direction by treating center point prediction as a Gaussian image prediction task. One could go further and use regression neural networks to achieve similar results. However, a regression-based approach might present challenges, as models regressing values from images tend to converge more slowly compared to image-to-image models, such as those used in segmentation tasks. Thus, careful consideration of loss functions would be needed.

Finally, one could fully remove any explicit transformation selection from the approach, and instead use a general deformation field as the transformation. The neural network would generate the values for this deformation field, akin to techniques used in image registration tasks. This would allow the network to automatically discover beneficial transformations or be guided to learn them by training with manually transformed versions of the input image as targets. This approach could potentially improve adaptability and data efficiency in medical image segmentation by allowing the network's capacity to learn and apply optimal transformations for each specific segmentation task and image.

All of these proposed improvements as well as the approaches described in this thesis represent different points on the spectrum of model complexity. As suggested by statistical learning theory, there is an inherent tradeoff between a model's bias and its variance. By explicitly incorporating domain knowledge into the model, we improve its data efficiency but limit its ability to model complex functions. Our approach facilitates a more nuanced management of this balance, offering a level of flexibility and precision that more traditional neural network-based segmentation strategies may not provide. Calibrating the balance between bias and variance is a fundamental decision when creating reliable medical image segmentation models, particularly when working with limited datasets. We hope that this work inspires further future research in ways to embed domain knowledge to simplify segmentation neural networks, as we believe such an approach is necessary in the field of medical imaging to achieve reliable segmentation when faced with small sample sizes.

8 Curriculum Vitae

Marin Benčević was born on November 4, 1996 in Vinkovci, Croatia. He has graduated from Zrinskih elementary school in Nuštar, Croatia, and the Matija Antun Reljković Gymnasium in Vinkovci, Croatia. He has obtained a Bachelor of Computer Engineering degree from the Faculty of Electrical Engineering, Computer Science and Information Technology Osijek at the J. J. Strossmayer University of Osijek, Croatia. He later obtained a Master of Computer Engineering in Information and Data Science. He is enrolled in a joint Ph.D. program to obtain a degree of Doctor of Computer Science at J. J. Strossmayer University of Osijek, and Doctor of Computer Science Engineering at Ghent University.

He is currently employed as a teaching and research assistant at the Faculty of Electrical Engineering, Computer Science, and Information Technology Osijek. During his Ph.D. studies, he has published four research papers in scientific journals (all as first author) and five papers presented at international scientific conferences (of which four as first author).

Marin Benčević
in Osijek, Croatia, July 5, 2024