Automatic Segmentation for Lower Limb Bones & Muscles using Deep Learning

Final Report for ENGSCI700: Part IV Project (Project #28)

ENGINEERING

Report By: Asif Juzar Cheena

Project Partner: Pranav Rao

Project Supervisor: Dr Julie Choisne

Declaration of Originality

This report is my own unaided work and was not copied from nor written in collaboration with any other person.

Name: Asif Cheena

Abstract

The musculoskeletal structure of children remains a significantly underexplored domain in research, plagued by challenges in the analysis of medical imaging data. Within medical imaging workflows, segmentation plays a pivotal role by identifying and localizing anatomical structures, enabling the study of their morphological changes over time. Manual segmentation, especially in paediatric populations, is a laborious and time-intensive task that demands meticulous annotation by expert researchers. In this study, we propose a robust deep learning based segmentation pipeline that performs automatic segmentation on Magnetic Resonance Imaging (MRI) on populations of children. The segmentation model used within the study is the Resnet34 U-Net deep learning architecture because of its strong segmentation accuracy and adaptability to transfer learning. The deep learning segmentation pipeline consists of a medical data processing layer, the Resnet34 U-Net deep learning segmentation model, and a denoising algorithm to post-process segmentation results. This is all integrated in a user-friendly GUI for ease of use. Experimental results were compared to state-of-the-art deep learning based segmentation methods within the biomedical domain, such as the nnU-Net framework, and the H-DenseUNet. The experiment accuracy of the ResNet34 U-Net is 89% (DSC), and outperforms the nnU-Net framework's 2D U-Net and 3D U-Net models on the segmentation of the tibia, femur, fibula and pelvis musculoskeletal structures. As well as the H-DenseUNet on the segmentation of the tibia structure.

Acknowledgements

I would like to express my sincere gratitude to my partner, Pranav Rao, for his unwavering support and dedication throughout this research endeavor. I would also like to extend my heartfelt thanks to Dr. Julie Choisne for her invaluable guidance and advice, which significantly contributed to the success of this study. The knowledge and insights gained from this research will undoubtedly leave a lasting impact on our academic and professional journey.

Table of Contents

List of Figures

List of Tables

1.0 Introduction

Medical imaging plays a pivotal role in modern healthcare, aiding in the diagnosis and treatment of various medical conditions. Segmentation is a vital step in many medical imaging workflows which aims to identify and localize meaningful anatomical structures by extracting their boundaries [1]. The advent of deep learning algorithms has brought about a paradigm shift in the field of medical imaging, enabling the automatic segmentation of complex anatomical structures from scans like Magnetic Resonance Imaging (MRI). The transformative potential of deep learning in medical imaging is underscored by the significant advantages it offers, particularly in the context of paediatric musculoskeletal studies.

For musculoskeletal system analysis, image segmentation is mainly employed to generate three dimensional (3D) models of bones and muscles, which in turn help researchers study how the morphology evolves in time [1]. Manual segmentation of musculoskeletal structures, especially in paediatric populations, can be a laborious and time-intensive process. It often requires expert researchers to spend hours meticulously annotating images. By automating the segmentation process, they reduce the time and effort required for detailed musculoskeletal analysis. This study aligns with the objectives of an ongoing imaging and computer modeling initiative [27] in Tairawhiti, New Zealand, aimed at advancing our understanding of paediatric musculoskeletal structure and function. Deep learning-based automatic segmentation models have the potential to alleviate these challenges they face.

Ongoing research into automatic segmentation methods in the context of medical imaging employs a diverse range of Convolutional Neural Network (CNN) architectures. Researchers predominantly leverage CNN models like U-Net, custom-designed CNNs, and variants of well-established architectures, such as ResNet, VGG, and DenseNet, depending on the specific segmentation tasks and characteristics of the medical images.

This study focuses on developing a robust deep learning based segmentation pipeline for paediatric lower limb musculoskeletal segmentation tasks from MRI scans. Our research journey involved a comprehensive investigation of four deep learning architectures, each tailored to the challenges posed by medical image segmentation. These architectures included the resnet34 U-Net, the 2D U-Net (integrated within the nnUNet framework), the 3D U-Net (also within the nnUNet framework), and the 2D U-Net. Our proposed segmentation pipeline uses a Resnet34 U-Net model because of its strong feature extraction capabilities, and adaptability through transfer learning. Furthermore, our segmentation pipeline is near transferable and has the potential to perform well on diverse paediatric populations, and model generalizability is something we explore in our study. The experimental results of our proposed segmentation pipeline are compared to state-of-the-art biomedical segmentation models.

The research report structure is as follows: Section 2 contains a literature review which examines approaches relating to deep learning based biomedical segmentation. Section 3 outlines the design and methodology of our pipeline and deep learning based model experiments. Section 4 presents testing methodology and results. Finally Sections 5 and 6 respectively contain discussions and conclusions drawn from our research.

1.1 Division of Work

Pranav Rao, and I together worked on the preprocessing pipeline for the medical imaging data. After that stage, I primarily worked on the deep learning model application while *Pranav* worked on data annotation methods and data preparation. Throughout the study at various stages we would concatenate our work into a single segmentation pipeline.

2.0 Literature Review

The literature review consists of background information, existing studies, and literature relating to deep learning based segmentation approaches in the biomedical domain.

2.1 Medical Imaging Segmentation

Medical image segmentation is essential in computer vision for highlighting regions of interest by processing 2D or 3D medical scan images or volumes to form a three-dimensional reconstruction of the anatomical structure [2]. Semantic segmentation is a specific subfield of image segmentation, which is a broader concept in computer vision. Semantic segmentation is primarily applied in the biomedical domain as it provides a class-specific understanding of structures within medical imaging, by classifying each pixel in a scan into a predefined class which represents each anatomical structure being segmented. Our project concentrates on isolating lower limb musculoskeletal groups in paediatric medical images.

In medical imaging, segmentation methods include manual, semi-automated, and automated techniques. Manual segmentation, performed by experts, is labor-intensive and subjective. Semi-automated methods combine human expertise with algorithms to improve efficiency and accuracy. Automated segmentation, relying on algorithms, has the potential to address these limitations, with deep learning-based approaches offering promising solutions for more accurate and efficient segmentation.

2.2 Deep Learning-Based Segmentation in Medical Imaging

Manual and semi-automated segmentation methods in medical imaging are limited by being labor-intensive, time-consuming, and subjective. However, deep learning-based segmentation models offer solutions that enhance efficiency and accuracy. Convolutional Neural Networks (CNNs) are a key component of these models. A CNN is a supervised deep learning framework that can accept images as input, allocate filters to convert image pixels into features, and apply those features to distinguish one data from another [3]. The CNN architecture is widely used in the biomedical segmentation domain due to its effectiveness in automating the segmentation process.

2.3 U-Net

The U-Net [28] is a CNN architecture developed for biomedical semantic segmentation. The topology of the network follows the encoder-decoder network structure of a typical CNN. The encoder path is the first half of the U-Net architecture and it downsamples the spatial resolution and is responsible for feature extraction from the input image. The decoder path is the second half of the architecture and it upsamples and concatenates (skip connections) detailed features maps from corresponding encoder block into the decoder to build the spatial resolution of the image back up, essentially creating the segmentation map. This concatenation of feature

information using skip connections in the U-Net architecture differentiates it from a fully convolutional network (FCN), and makes it highly effective for the task of biomedical semantic segmentation [4].

The encoder path of a U-Net contains the following layers:

1. **Convolutional Layers:** The encoder starts with a series of convolutional layers. These layers perform feature extraction, capturing increasingly abstract features of the input scan as you go deeper into the network. Within this layer a kernel/filter (square/rectangular matrix) containing the learnable parameters (model weights) is slid over the input scan to extract features. Feature extraction is the process of computing the dot product between the kernel model weight values and the localized region of pixel data in the scan it covers. Computing the dot product forms feature maps, patterns in the data, and this process of creating feature maps effectively compresses information within the kernel's respective field, resulting in the dimensional reduction of the image. This dot product value is passed through an activitiation function and this introduces non-linearity to the network to capture more complex patterns. Padding is a technique used to keep the spatial dimensions the same as the feature map by essentially 'padding' it with zeros around its borders. After determining the values of the feature map, this sliding window operation of the kernel/filter shifts from the top left position of the scan into a new position and the stride parameter of the layer determines the degree of shift. The hyper-parameters of this layer include: the number of filters/kernels, kernel size, stride, padding, and activation function. Batch normalization can be applied to feature maps from this layer.
2. **Pooling Layers:** The feature map from the convolution layer is passed into the pooling layer, and this is where the down-sampling occurs. The feature map inputted into the layer is divided into non-overlapping regions/grids and within each region a pooling operation is applied to summarize each region with the single maximum/average value. The hyper-parameter of the layer is the pool size and correlates to the size of each overlapping region.

The decoder path of a U-Net contains the following layers:

1. **Transposed Convolutional Layers (Upsampling):** The decoder starts with this layer, where the down-sampled feature map is passed into. It applies the set of learnable kernels/filters to the input, and performs the convolution operation similar to the encoder but with upsampling. The stride now controls the upscaling factor and determines how much the spatial dimensions increase. Rather than a simple dot product calculation like the convolutional layer in the encoder, the transposed convolutional layer redistributes the weights in a strategic manner to increase the spatial dimensions of the feature map. Each decoder block has multiple of these layers that the feature maps are subsequently passed through. The hyper-parameters of the layer are the same as the convolutional layers in the encoder path, but vary in configuration.
2. **Skip Connections and Concatenation:** This is the defining characteristic of a U-Net, where the output from the corresponding encoder layer is concatenated with the upsampled feature map of the same spatial resolution in the decoder path. This combination of knowledge allows the network to combine low and high level information.

2.3.1 2D U-Net

The 2D U-Net is built with 2D convolutional layers, and handles two-dimension input data, such as 2D scan images, and these models use pixel feature information to form segmentation outputs. The 2D U-Net was

applied in a study [5] looking into musculoskeletal segmentation from MRI scans of a paedritic population, identical to our study, and achieved an experimental accuracy (dice similarity score) of 87%. This performance was on par with a powerful deep hybrid CNN model, H-DenseUNet (90%) [5]. This study used a robust performance evaluation method, testing both models using k-fold (k=5) cross validation on 20 scan volumes, further supporting the learnings derived about the 2D U-Net within a very related medical domain.

A study [6] on using the 2D U-Net to perform prostate segmentation out of MRI scans showcase the architecture specifications used, and demonstrate their success with the network topology achieving a mean experimental accuracy (dice similarity score) of 87.38%. Their 2D U-Net architectural implementation has 5 convolutional encoder blocks and 4 convolutional decoder blocks, and each convolutional block within the respective path have the following characteristics:

1. Input Layer: Scan input shape of 320x320x1, and batch normalization
2. Convolution Blocks: 3x3 convolution layers, batch normalization layers, and ReLU activation layers.
3. Downsampling: 2x2 average pooling
4. Dropout: Dropout layers with a dropout rate of 0.5 are applied after the fourth and fifth convolution blocks.
5. Upsampling: Consist of nearest neighbor interpolation followed by 2x2 convolution. Also include batch normalization and ReLU activation layers.
6. Final Classification Layer: Composed of 1x1 convolution, batch normalization, and sigmoid activation layers.

This architectural design was experimentally optimized within the study, highlighting the considerations made at each iteration in the process. Studies such as [5][7][8], looking into varying biomedical segmentation task domains, lower limb musculoskeletal segmentation from MRI scans, lung segmentation from CT scans, and neuronic structure segmentation from microscopic imaging, respectively demonstrate the powerful performance of the highly regarded 2D U-Net architecture. The studies mentioned outline the adaptability of the network into other biomedical domains, ability to utilize pre-trained models, and lightweightedness (model compactness) relative to other deep learning techniques.

2.3.2 3D U-Net

The 3D U-Net is built with 3D convolutional layers, and handles three-dimension input data, such as 3D scan volumes, and these models are able to concatenate both volumetric information and pixel feature information to form segmentation outputs. The architectural structure of the 3D U-Net and 2D U-Net are fundamentally similar, however they handle scan data in three-dimensions and that means the kernels/filters and strides account for the depth dimension (z), allowing the model's learnable parameters to capture spatial information across the whole volume of the input scan volume.

A study [9] done on the automated detection and segmentation of lymph nodes on pelvic diffusion-weighted imaging (DWI) images using a 3D U-Net architecture achieved an experimented accuracy measure (dice similarity score) of 85%. The model was promising considering data-related challenges encountered such as imbalanced data, specificity, manual segmentation variability, and the overall difficulty of the segmentation task.

Studies such as [5][10] highlight the short-comings of the architecture outlining disadvantages such as high computational demands for the training and inference phases, and model robustness requiring a large amount of labeled data which is scarce and costly in the medical imaging domain. The main disadvantage is the associated memory constraints especially with high-resolution imaging, which requires breaking up the scan volume into 3D patches, and this leads to a loss in global information, and ineffective training. Alternative methods include smaller batch sizing, however this could potentially lead to underfitting issues. The 3D U-Net is a powerful architecture concatenating both 2D and 3D information in segmentation tasks, however poses severe challenges in regard to model performance and computational resource availability tradeoffs.

2.4 nnU-Net

The state-of-the-art nnU-Net (not new U-Net) [11] is a self -configuring framework designed for biomedical semantic segmentation tasks. It represents an extension and refinement of the original U-Net architecture, tailored to the specific requirements of biomedical imaging applications. This self configuring framework automates the preprocessing, post-processing, training and inference pipelines for our given dataset using fixed, rule-based and empirical parameterization.

The nnU-Net framework introduces several key features and innovations to model parameterisation:

1. Fixed Parameterisation: The nnU-Net uses the 2D U-Net and 3D U-Net architectures as deep learning backbone models to the framework. This aspect of the framework remains fixed independent of characteristics of the dataset such as modality of the medical imaging or the biomedical domain of the segmentation task. Other fixed parameters in the framework include the data augmentation methods employed, and hyper-parameters such as: the learning rate, choice of optimizer and loss function.
2. Empirical Parameterisation: The configuration of the preprocessing pipeline and the techniques applied to the dataset is an empirical process as it is heavily reliant on the characteristics of the dataset. Through strategic experimentation the framework configures a preprocessing pipeline for the given dataset employing different parameters and techniques for: image resizing/cropping, scan normalization scheme, and image resampling.
3. Rule-Base Parameterisation: This an algorithmic approach to optimizing the framework to the dataset. It uses a dynamic adaption of network topologies to adapt to characteristics of the dataset mainly input image size, adjusting input patch sizing, number of pooling and convolutional layers per block, and accounting for technical constraints such as memory availability to adjust network hyper-parameters such as batch sizing [11].

The nnU-Net's self-configuring nature adapts to the intricacies of medical data, and its strengths lie in its powerful optimization strategies opposed to revolutionary architectural alterations. A study [5] earlier reference in the same medical domain as our study showcased powerful performances of their 2D U-Net and H-DenseUNet implementations, and in the study it was documented that a nnU-Net was employed, and it achieved dice similarity scores on par with the H-DenseUNet of 89%. Another study [12] applied the nnU-Net to a series of highly distinct medical datasets for a Medical Segmentation Decathlon to analyze its performance in the various domains. The framework outperformed all other models in 12/13 label segmentation tasks involving the segmentation of the following organs: brain tumor (except class 1), heart, liver, hippocampus, prostate, lung, and pancreas. The nnU-Net is a powerful baseline model given any medical domain, and leveraging its established excellence would help evaluate the effectiveness of proposed models.

2.5 Transfer Learning-Based Approaches

Transfer learning-based methods have emerged as powerful tools in biomedical image segmentation tasks, consistently surpassing established deep learning models within the field [13].

Pre-trained models leverage prior knowledge from datasets like ImageNet, CIFAR, and medical datasets like CAMELYON16/17. These models learn transferable features, including edge detection and texture recognition. Fine-tuning on biomedical data tailors these features to the specific task. The adaptation optimizes model parameters and aligns knowledge with the biomedical domain. Pre-training enhances model performance and speeds up convergence, reducing the dependence on manually annotated data. The use of transfer learning fosters model generalizability through general feature knowledge, and this is crucial in the biomedical imaging domain where small perturbations have the potential to throw the model off including slight changes in scan contrast, variations in the patient's position/orientation, or other minor image variations.

A study [14] looking into the segmentation of pelvic structures from Computed Tomography (CT) scans demonstrated the impact transfer learning had on their study when integrated with the U-Net architecture. They experimented with the following 4 CNN models: FCN, PSPNet, U-Net, and Segnet. The U-Net demonstrated the best performance with an experimental accuracy (dice similarity score) of 92.86%. They extended their exploration of the U-Net through transfer learning based modifications, and this improved model performance boosting the experimental accuracy of the pelvis segmentation task to 96.31%. Other studies [15], highlight the positive impact on model performance when extending the U-Net architecture with pre-trained weights, and the importance of weight initialization on U-Net task segmentation.

2.5.1 Pre-trained U-Net

A pre-trained U-Net is an extension of the traditional U-Net that incorporates a pre-trained backbone model as the encoder network to the model. Initializing the encoder path weights of these deep architectures with pre-trained weights from large image datasets such as ImageNet [ref] is a high level methodological overview in implementing a segmentation model using transfer learning. Established pre-trained backbone models have powerful image segmentation architectures and include models such as: VGG16/19 [29], ResNet34/50 [30], Inceptionv3 [31], and MobileNetV2 [32]. As mentioned in **Section 2.3 U-Net**, the encoder to the U-Net architecture is responsible for feature extraction. The use of a pre-trained backbone opposed to a randomly initialized traditional encoder path enhances the feature extraction capabilities of the model and this is evident in studies [15][5] that document faster model convergence times, and improvements in segmentation results when applying transfer learning to the U-Net architecture. The pelvic structure segmentation study [14] mentioned earlier found success using the ResNet backbone model pretrained on the ImageNet [33] dataset. Another study [15] on retina segmentation, on an established Digital Retinal Images for Vessel Extraction (DRIVE) dataset [34], detailed the significant improvement in model performance encountered when comparing their from-scratch implemented U-Net model and ResNet pre-trained U-Net model, reporting a significant increase in DSC from 34% to 82%. In our study, we will empirically/experimentally determine the optimal pre-trained backbone network for our domain.

3.0 Project Scope and Research Objectives

3.1 Project Scope

Deep learning based approaches to the automatic segmentation of paedratic lower limb musculoskeletal structures is a relatively understudied field [ref]. The goal of our study is to develop a deep learning segmentation pipeline with comparable performance to existing state-of-the-art deep learning approaches in the biomedical domain. The results of our study could further aid with diagnosing and monitoring musculoskeletal disorders in children. Furthermore we develop a GUI tool that makes the segmentation pipeline easy to use.

3.2 Research Aims and Objectives

The key objectives of this study were to:

1. Develop a data preprocessing pipeline that puts MRI scan and mask data into a deep learning model input format.
2. Develop multiple deep learning segmentation models for automatic segmentation of lower limb bones from MRI scans in children, and determining the best one from the performance evaluations
3. Prepare annotated data by manual segmentation of lower limb bones from MRI scans under domain expert guidance. In conjunction with *Pranav Rao.*
4. Validate the deep learning segmentation model by comparing model performance with the findings of similar studies, and the nnU-Net segmentation framework.
5. Validate the generalizability of the deep learning segmentation model on a dataset of a distinct paedritic population.

4.0 Methodology

4.1 Medical Imaging Dataset

The project utilized a dataset comprising 30 participants from a paediatric study conducted in Tairawhiti, New Zealand [1]. The children in the study were aged between 7 to 12 and represented Māori and Pakeha demographics. Within the study this will be referenced as the Tairawhiti dataset. Informed consent/assent was obtained from parents before participation, and ethical approval was granted by the Health and Disability Ethics Committee (20/CEN/107). Additionally, the study underwent review by an independent Māori Advisory Board to align with regional values and priorities [27].

From this study, the provided data includes MRI scans of the children and PLY (Polygon File Format) manual segmentations of their right leg lower limb bone groups, including the tibia, femur, and fibula. These manual segmentations were carried out using Stradview 3D Imaging Software [35]. Additionally, for the pelvis structure, manual segmentations were performed on 3D Slicer [26] by my project partner, *Pranav Rao*, for 6 paediatric participants.

The MRI data comprises full-body scans, categorized by patient and imaging type. Water sequence imaging, recognized as the most effective for segmentation-related tasks [3], was employed in our study. Each child's MRI scan volume consisted of 1015 MRI slices, with each slice being a 512x512 pixel grayscale image, having a pixel spacing of 0.9375mm x 0.9375mm. The slices in the MRI scan were taken 3 mm apart along the child's height (slice thickness). This information allows us to define the dimensions of a single voxel, with the voxel size being 0.9375 x 0.9375 x 3mm. The volume of a single voxel is calculated as the product of these spatial dimensions, yielding 2.64mm3. This is a crucial measure for evaluating volume-based performance metrics. The MRI scans are stored in DICOM file format, containing both coordinate and pixel data, facilitating their alignment with the manual segmentations.

4.2 Data Labeling Using Manual Segmentation

To train our deep learning model, we need labeled data, which outlines the region of interest we are training the model to segment. This labeled data (mask data) is the groundtruth data that was obtained through manual segmentation, where domain experts defined the boundaries of bone and muscle groups in the lower limbs in the 2D scan. The resulting masks are then compiled into a 3D representation. The manual segmentations required to acquire labeled pelvis data was performed by my project partner, *Pranav Rao*, and it was done using the 3D Slicer [26] 3D modeling software. The manual segmentation pelvis structures were refined using Gaussian smoothing functions within 3D Slicer [26], and the results were verified by our research supervisor, *Dr Julie Choisne*, a domain expert.

4.3 Preprocessing Overview

As part of the deep learning segmentation pipeline, we created a preprocessing layer that transforms the raw 2D MRI slices and the corresponding 3D ground truth mask into a suitable format and range for effective learning by the neural network. The preprocessing steps included:

4.4 Medical Image Data Preprocessing

Within each MRI slice the file metadata holds the pixel intensity data, and spatial information from the scanned images. This data was extracted using the Pydicom (version 2.4.2) [16] library within Python 3.9 [17].

The pixel data of all scan data is normalized using min-max normalization, and the normalized pixel intensity values fall in the range of [0,1]. Pixel intensity normalization standardizes an intensity scale and range for the deep learning model to effectively extract features. The pixel intensity data was min-max normalized by dividing every pixel value by 255.

The spatial data is less relevant to the model learning task, however, it helps evaluate characteristics of the subjects through the raw data, such as: the height of the child and length of musculoskeletal structures. It also proves helpful for data quality validation to identify at which slice along the child's height a specific bone/muscle group starts and ends.

4.4.1 Biomedical Segmentation Mask Creation

For each MRI slice in our dataset, the creation of a corresponding 2D segmentation mask is a pivotal step in preparing the data for model training. The manual segmentations are stored as 3D geometries within PLY files. Creating the 2D segmentation masks requires spatially aligning the bone/muscle segmentation with the MRI scan, and highlighting the region of interest.

4.4.1.1 Binary Segmentation Masks

Binary segmentation masks are 2D segmentation masks that are binary in nature, where 1 is the pixel value for any pixel within the region of the interest from the manually segmented structure, and 0 is the pixel value for any pixel outside of the region of interest. This allows the model to effectively learn to segment the region of interest out of MRI slices on a 2D level. The effectiveness of this learning process is heavily reliant on the mask quality and the mask quality can be assessed by how well it outlines the region of interest within the raw slice.

3D Slicer [26], an open source 3D Imaging Software, was key in creating the binary masks by utilizing state-of-the-art image registration algorithms. Image registration is a critical step that ensures the spatial alignment of the 2D MRI slices and 2D binary segmentation masks. It corrects for any spatial variations or deformations present in the original 3D manual segmentation. The following were the steps taken in creating the binarized segmentation masks in 3D Slicer [26] for an individual patient:

1. Import both, the raw DICOM MRI scan stack, and the PLY file for the 3D geometry of the manual segmentation.
2. Create binary masks using the Binary Label Map export function within 3D Slicer [ref]
3. Export binary masks as 3D NIfTI (Neuroimaging Informatics Technology Initiative, *.nii) [18] files

4.4.1.2 Multiclass Segmentation Masks

Multi-class masks enable the deep learning algorithm to distinguish various bone/muscle groups and the background, extending the model's capability to perform multi-class classification and segment multiple musculoskeletal structures. This integration occurs within the preprocessing layer, where binary segmentation masks of different musculoskeletal groups are concatenated into a single multi-labeled mask. This process was

accomplished using the Python libraries Nibabel (version 5.1.0) [36] and Numpy (version 1.22.4) [37]. The following steps were taken to create these multi-class segmentation masks:

1. Define class values for the encodings that represent the different bone/muscle groups
    - The following details the encoding values that map to the different segmentation tasks: {0 - Background, 1 - Tibia, 2 - Femur, 3 - Fibula, 4 - Pelvis}
2. Encode the binary masks using the unique class value
    - The NIfTI binary masks were imported into Python and the pixel intensity data was extracted using the Nibabel [ref] library. The mask's non-background encoding value (1) was replaced with the encoding value that mapped to the segmentation task.
3. Combined the encoded binary masks
    - The individual binary masks were concatenated together additively using the NumPy[ref] library
4. Assign the background class value (0) to any overlapping regions between different bone/muscle groups
    - After concatenation, if the multiclass segmentation mask contained any encoding not mapped to a segmentation task, it indicated an overlapping region, and these regions were mapped to the background class. Finding the unique encoding in the multi-class mask was done using the NumPy[ref] library.

Figure 2A displays an MRI scan slice taken around the child's hip region, capturing both the pelvis and femur within the same axial plane. **Figure 2B** represents the corresponding 2D multiclass segmentation mask, where white pixel encodings indicate the pelvis structure, and gray pixel encodings represent the femur structure. **Figure 2C** provides a superimposed visualization of the MRI slice and multiclass mask, illustrating how the mask data correlates with the region of interest in the MRI scan data. **Figures 2A** and **2B** collectively represent a single training instance in the supervised deep learning model.

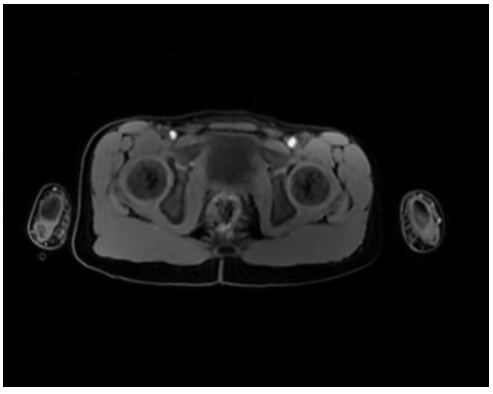

Figure 2A: MRI Slice (Hip Region)

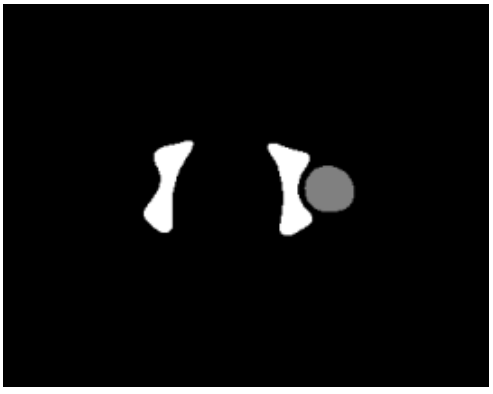

Figure 2B: Multi-class Mask (Pelvis, Femur)

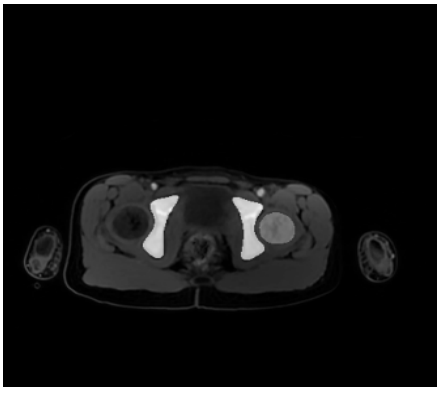

Figure 2C: MRI Slice & Multi-class Mask Superimposed

During the process of combining individual femur and pelvis binary masks, overlapping regions were encountered. To establish distinct boundaries and better model the areas of interest, the overlapping region was encoded with a unique background value (0).

4.4.3 Image Preprocessing Techniques and Data Augmentation

4.4.3.1 Image Data Resizing

Image resizing is a commonly employed image preprocessing technique used in various computer vision tasks. The decision to resize our input scan and mask images was made through experimental exploration, with the primary aim of finding a balance between pixel information preservation and computational efficiency.

In our baseline 2D U-Net model, which was applied to a binary segmentation task on the tibia skeletal structure, we iteratively trained dataset variants with different image size configurations. This was performed on a subset of 10 children (5890 images), with 6 subjects used for training, 2 for validation, and 2 for testing. We employed k-fold cross-validation (k = 5) to assess the mean performance metrics. The model's parameterization and training configurations remained consistent, and we considered three different image sizes: 512 x 512, 400 x 400, and 256 x 256.

Image resizing was integrated into the preprocessing layer, using the Image module from the PIL(Python Imaging Library, 9.3.0) python library [39], using the following steps:

1. Convert the 2D image to a Pillow image
2. Resize the Pillow image to [512 x 512, 400 x 400, 256 x 256] using the Image.BILINEAR parameter

BILINEAR was used as the image interpolation function, and it is a common method for resizing images as it uses information from neighbor pixels to calculate new pixel intensity values when changing the spatial dimensionality of an image, resulting in a smoother transition and preserving image details.

4.4.3.2 Image Data Augmentation

Data augmentation is a technique commonly used in deep learning to artificially increase the diversity of a dataset by applying various transformations to the original data. Data augmentation can be important when working with medical imaging data like MRI scans, where there can be variations due to factors such as child's positioning, and field of view (FOV) shifting. The decision to augment data was dependent on computational resource availability required storing the unaugmented training data and if mask data scarcity was a limiting factor in any bone/muscle group segmentation tasks.

Data augmented was employed on the preprocessed scan and mask data before model training, utilizing the cv2 module (version 4.8.0) from the OpenCV [39] Python library. Augmentations were applied to normalized scans and corresponding masks volumetrically to appropriately model variations in the MRI scanning procedure. The following transformations were applied in the augmentation process:

1. Rotation about the coronal axis (frontal axis) randomly sampled from a uniform distribution within [-15°, 15°]
2. Probabilistically flipping images about the sagittal axis from a uniform distribution within [0, 1].

The augmentation process increased the number of MRI scans in the dataset by a factor subject to the storage resource availability for a given model experiment, however this factor would be 2 for most instances.

4.4.4 Dataset Specific Preprocessing

The 3D scan volumes covered the entire patient's body, but our model focused on musculoskeletal structures below the hip/pelvis. Including irrelevant data in the model can be computationally costly and introduce noise, and negative impacting segmentation accuracy.

To address this, we introduced a vital preprocessing step specific to our dataset. Here's how it works:

1. Using the 3D mask volumes, find the first 2D mask array containing a non-zero encoding, this is the first slice along the participant's height containing information relevant to the segmentation task. The first given slice containing segmentation information is referred to as the "subsetting slice".
2. Subset the 3D scan and mask volume data to only contain information below the subsetting slice

This step followed the creation of multi-class segmentation masks (**Section 4.4.2.2**) and relied on label data to determine the subsetting slice, where the pelvis structure begins. For test scans without manual segmentation, users estimated the subsetting slice. To accommodate potential variations, we added a 50-slice tolerance to mitigate error stemming from user-specific variability.

In our paediatric dataset, the subsetting slice typically fell within the 300-500 range. However, we acknowledge this as a current limitation in our segmentation pipeline, to be discussed further.

4.5 Deep Learning Segmentation Model

4.5.1 Baseline 2D U-Net

The baseline 2D U-Net approach explores whether our musculoskeletal segmentation task can be effectively tackled using a simple 2D U-Net architecture and combining this with the hyper-parameter configurations inspired from an existing implementation within the same medical domain [5]. The implementation of this model will help develop a baseline understanding of the U-Net architecture and the segmentation task complexity. Our implementation takes 2D MRI slices from the participant's scan volume as input with the shape (256x256x1), where 256x256 is the slice resolution, and 1 is the channel size, indicating grayscale data.

The 2D-UNet architecture was built utilizing Keras 2.0 [9] and Tensorflow 2.0 [10]. Our specific implementation was inspired by a 6 stage 2D-UNet built for a brain tissue segmentation task [19].

Figure 3 in the appendix is a visual representation of a 4 encoder-decoder block 2D U-Net architecture.

The 2D U-Net architecture has 4 encoder and decoder stages (additional bottleneck stage), and each of the paths have the following stage properties:

Encoder:

- 2 2D Convolutional Layers per encoder block
  - *Applying a 3x3 kernel/filter size for learning and ReLU activations to capture nonlinear patterns. Padding on the convolution layers is set to 'same' to maintain output size (256x256x1). This is important for preserving spatial information.*
- 1 Max Pooling Layer per encoder block

- *1 max pooling layer per stage, set with a 2x2 kernel/filter size to reduce the dimensions of the slice image (1x1 filter for the input stage)*

Bottleneck:

- 2 2D Convolutional Layers
  - *Applying a 3x3 kernel/filter size for learning and ReLU activations. Padding on the convolution layers is set to 'same' .*
- 1 Dropout Layer
  - *Dropout layers are a regularization technique to reduce overfitting. Applying a 50% dropout rate to the 5th convolutional layer in the bottleneck block*

Decoder:

- 1 Deconvolutional Layers (Upsampling) per decoder block
  - *Set with a 3x3 kernel/filter size to increase the dimensions of the slice image. increase resolution of the slice images, re-creating the original image resolution.*

- Concatenation (Skip Connections) and 2 2D Convolutional Layers per decoder block
  - *The convolution and concatenation layers apply a 3x3 filter size for learning and ReLU activations. detect finer details, such as specific boundaries for the bone groups.*

Output:

- 1 2D Convolutional Layer
  - *1x1 kernel/filter size with a softmax activation to produce a multi-label output segmentation mask.*

The hyper-parameter configuration was derived from the 2D U-Net implementation of another study [5] used and is as follows: Adam optimizer, batch size of 30, learning rate of 1e-3, dice loss function, and the softmax activation function on the output layer.

4.5.2 nnU-Net Model

The nnU-Net, or No New-Net U-Net, is a semantic segmentation method that automatically adapts to a given dataset. It analyzes the provided training cases and automatically configures a matching U-Net based segmentation pipeline [12]. This self configuring framework automates the preprocessing, training and inference phases for our given dataset using fixed, rule-based and empirical parameterization, as explored in the literature section. The implementation of the out-of-the-box method was inspired from the creator's github repository [20].

4.5.2.1 Preprocessing Plan

The scan and mask data was inputted into the nnU-Net as compressed NIFTI (.nii.gz) files, and upon analyzing key properties of the data the nnU-Net extracts a dataset fingerprint. This dataset fingerprint defines a set of dataset-specific properties, including the following features: image sizes, voxel/pixel spacing, and intensity

information. This information is then used to design 2D/3D U-Net configurations held in json files, and each of these pipelines operate on its own preprocessed version of the dataset.

4.5.2.2 nnU-Net Configuration Architectures

The two U-Net configurations explored within our study are the 2D U-Net and 3D U-Net deep learning architectures. Either model has unique hyper-parameter configurations and configured preprocessing pipelines, purely derived from the properties of our paedetric dataset. This includes features such as batch size, patch size, scan data normalization technique, learning rate, and choice of loss function and optimizer (**Section 4.5.2.3.1**). The network architecture is also predefined depending on the configuration chosen, and as stated in the literature the U-Net architecture for each configuration is fixed and is not adaptive to the dataset. The network's topology is dynamic, adjusting the number of convolutional and pooling layers to account for the input shape of our dataset.

4.5.2.2.1 2D U-Net

The 2D U-Net configuration takes an input size of (256, 256, 1) and this is the dimensional representation of a single 2D scan or mask slice from our dataset. The number of classes is a parameter that the nnU-Net framework learns from the number of unique encodings in the mask data, and it represents the number of musculoskeletal structures being trained to segment by the model. The model architecture has 7 encoder and 6 decoder stages (with a bottleneck layer), and each of the paths have the following stage properties:

Encoder:

- 2 2D Convolutional Layers per encoder block
  - *Applying a 3x3 kernel/filter size for learning and ReLU activations to capture nonlinear patterns. Padding on the convolution layers is set to 'same' to maintain output size (256x256x1). This is important for preserving spatial information.*
- 1 Max Pooling Layer per encoder block
  - *1 max pooling layer per stage, set with a 2x2 kernel/filter size to reduce the dimensions of the slice image (1x1 filter for the input stage)*

Decoder:

- 1 Deconvolutional Layers (Upsampling) per decoder block
  - *Set with a 3x3 kernel/filter size to increase the dimensions of the slice image. increase resolution of the slice images, re-creating the original image resolution.*

- Concatenation (Skip Connections) and 2 2D Convolutional Layers per decoder block
  - *The convolution and concatenation layers apply a 3x3 filter size for learning and ReLU activations. detect finer details, such as specific boundaries for the bone groups.*

Output:

- 1 2D Convolutional Layer
  - *1x1 kernel/filter size with a softmax activation to produce a multi-label output segmentation mask.*

4.5.2.2.2 3D U-Net

The 3D U-Net configuration takes an input size of (D, 256, 256, 1) and this is the dimensional representation of the 3D volume scans and masks for a single input participant, where D is the number of 2D slices in the 3D volume. The depth dimension varies on a participant level with varying limb and musculoskeletal structure lengths. The number of classes input parameter is self learnt by the adaptive framework of the nnU-Net, similarly to the 2D U-Net configuration.

The model architecture has 6 encoder and 5 decoder stages (with a bottleneck layer), and each of the paths have the following stage properties:

Encoder:

- 2 3D Convolutional Layers per encoder block
  - *2 convolutional layers per stage, applying a 3x3x3 kernel/filter size that detect volumetric patterns in the 3D scan volumes for learning and ReLU activations to capture nonlinear patterns.*
    Padding on the convolution layers is set to 'same' to maintain output size (Dx256x256x1). This is important for preserving spatial information.

- 1 Max Pooling Layer per encoder block
  - *1 max pooling layer per stage, set with a 2x2x2 kernel/filter size to reduce the dimensions of the slice image (1x1x1 filter for the input stage)*

Decoder:

- 1 3D Deconvolutional Layer (Upsampling) per decoder block
  - *Set with a 2x2x2 kernel/filter size to increase the dimensions of the 3D scan volumes, re-creating the original image resolution.*
- Concatenation (Skip Connections) and 2 3D Convolutional Layers
  - *The convolutional and concatenation layers apply a 3x3x3 filter size for learning and ReLU activations.*

Output:

- 1 3D Convolutional Layer
  - *1x1x1 convolutional layer with a softmax activation to produce a 3D segmentation volume of the musculoskeletal structure.*

4.5.2.3 Training Procedure

4.5.2.3.1 2D U-Net Model

The 2D U-Net model was trained with a mini-batch stochastic gradient descent algorithm using the Adams optimizer. The batch size was set to 12, with the model training for 40 epochs. The learning rate that started at a value of 10^{-3} and with each epoch iteratively decreased by 10^{-5} . The magnitude of decrease is the learning rate

decay schedule and a dynamic learning rate scheme is powerful for model convergence, by progressively reducing the size of model weight updates. The configuration utilized a custom loss function that combines the properties of the Dice loss and Categorical Cross Entropy (CSE) loss functions.

4.5.1.3.2 3D U-Net Model

The 3D U-Net model's training closely resembled that of the 2D U-Net, including the choice of loss function, optimizer, learning rate schedule, and number of epochs. Notably, the batch size was set to 2, which was a logical choice due to the significantly higher memory demands of 3D volumetric data compared to 2D slices. Each 3D MRI volume in our dataset contained around 750-850 slices, so a batch size of 2 with the 3D model roughly corresponds to a batch size of 1500-1700 for the 2D model. To overcome GPU constraints, the nnU-Net employed training on 3D patches, smaller sub-volumes extracted from the larger 3D MRI scan. These patches are subsets of the complete 3D data, and in our dataset, they were defined as (128, 112, 160) voxels in size. This patch size indicates that the model processed a segment of 3D scan data measuring 128 voxels in width, 112 voxels in height, and 160 slices in depth.

4.5.3 Transfer Learning-Based Approach

In our study, the transfer learning-based approach involves leveraging a pre-trained neural network model initially designed for a large image dataset. We adapt this model to perform specific segmentation tasks for our lower limb musculoskeletal domain. As noted in the existing literature, this approach offers advantages such as faster and more efficient parameter convergence, the ability to work effectively with smaller annotated datasets, and improved segmentation results when compared to training a neural network from scratch with randomly initialized encoder and decoder weights. This transfer learning implementation is a fundamental component of our strategy, employed to enhance both the baseline nnUNet models and other biomedical semantic segmentation models featured in various relevant studies.

Our implementation entails constructing a U-Net style architecture. We replace the encoder network with a pretrained Resnet34 backbone model, which is a deep neural network architecture designed for image classification. This Resnet34 [44] model consists of 34 convolutional layers and is described in the paper "Deep Residual Learning for Image Recognition." We use it to process 2D scan slices with dimensions (256x256x3) as inputs and 2D segmentation mask slices with dimensions (256x256x1). Although the scan slice channel size is 3, it's essential to clarify that the data remains grayscale, not RGB, and is min-max normalized. This adaptation is necessary to meet the input shape requirements of the pre-trained Resnet backbone architecture. **Figure 4** in the appendix illustrates the Resnet34 U-Net architecture, highlighting the ResNet34 encoder network, and the U-Net decoder.

The pre-trained backbone weights used in our implementations were trained on the ImageNet [ref] dataset. This is a large-scale image database designed for use in visual object recognition and classification tasks.

The following 3 experimental areas were explored in our methodology using a transfer learning-based approach:

1. Assessing the impact of employing the ImageNet pre-trained backbone for the U-Net's encoder on segmentation results against a model with randomly initialized backbone weights.

2. Identifying the optimal deep learning backbone architecture that yields the best segmentation accuracy within our specific domain.
3. Determining the optimal set of hyper-parameters for our pre-trained U-Net model.

We accessed the backbone architectures and ImageNet [33] pre-trained weights through the segmentation-model (version 0.1.2) [40] Python library and implemented them using Keras (version 2.13.1) [41] and TensorFlow (version 2.13.0) [42].

4.5.3.1 Transfer Learning Experimental Training Strategy

The experimental transfer learning models were constructed with the following hyper-parameters: it employed the Dice loss function, Adam optimizer (SGD), utilized a batch size of 8, underwent 20 training epochs, featured a softmax activation function, and maintained a constant learning rate of 1e-4 without any learning rate decay. The training/fine-tuning process involved 7 participants and encompassed segmentation tasks for the tibia, femur, fibula, and pelvis bone groups. To ensure an unbiased performance evaluation, leave-one-out-cross-validation (LOOCV) was applied at the subject level.

4.5.3.2 Pre-Trained vs Randomly Initialized Model Weights

In our experiment, we aimed to assess the impact of using a pre-trained backbone on model performance when employing a transfer learning-based approach. To achieve this, we utilized a resnet34 [30] image segmentation architecture as the encoder for a U-Net model. We evaluated the U-Net model's segmentation accuracy in two scenarios: one with a resnet34 backbone pre-trained on the ImageNet [33] dataset, and the other with a resnet34 model with randomly initialized weights. The objective was to determine whether fine-tuning pre-trained network weights offered superior segmentation accuracy compared to fine-tuning network weights from scratch (random initialization).

The models were compiled and trained/fine-tuned using the hyper-parameter configuration and training strategy stated in **Section 4.5.3.1**.

The following methodological overview was used in implementing the pre-trained network on our research dataset:

1. Initialize the U-Net with the imagenet pretrained resnet34 architecture as the encoder
2. Initialize the U-Net model’s decoder with the imagenet weights
3. Freeze the encoder network weights
4. Fine-tune the model’s decoder weights using the paedratic dataset

The following methodological overview used in implementing the network with randomly initialized weights on our research dataset:

1. Initialize the U-Net with a randomly initialized resnet34 architecture as the encoder
2. Initialize the U-Net model’s decoder with weights randomized between [-1,1]
3. Fine-tune the U-Net’s encoder and decoder weights from scratch

4.5.3.3 Pre-Trained Deep Learning Backbone Architectures

Our implementation of a transfer learning-based approach uses the U-Net architecture, where the encoder network is replaced with a pre-trained backbone architecture. The encoder path of a U-Net is responsible for feature extraction from image data and using a backbone network as the encoder to the U-Net excels the feature extraction capabilities of the model.

It was important to experimentally determine the backbone model that provided the best segmentation accuracy on specific musculoskeletal tasks. The following backbone models were included in the experiments: Resnet34 [30], Resnet50 [30], VGG19 [29], and MobilenetV2 [32]. The motivation behind the use of these backbone networks stems from success in other biomedical segmentation and wider image classification research [21], and further explored in the literature.

The experiment was carried out by iteratively replacing the U-Net's encoder with an imagenet pretrained backbone network of interest using the hyper-parameter configuration and training strategy stated in **Section 4.5.3.1**

4.5.3.4 Network Hyper-parameter Optimization

The hyper-parameter configurations used for our final transfer learning model were initially chosen with respect to the hyper-parameter configurations and optimization methods used in deep learning based studies for similar biomedical segmentation tasks [5][13]. The following details how each hyper-parameter configuration was derived, or experimentally/empirically optimized for our deep learning implementation:

1. **Optimizer:** We employed the Adam optimizer [22] as the chosen optimization algorithm for our model. This optimizer is widely recognized as highly effective for image classification tasks. Adam represents an extension of the well-known stochastic gradient descent (SGD) algorithm, offering robust adaptive learning and moment estimation capabilities. Its popularity in the realm of stochastic optimization is well-founded [22].
2. **Loss Function:** The choice of loss function was experimentally determined by evaluating segmentation accuracy for independent training tasks using different loss functions. The 3 loss functions of interest included: Dice loss, Categorical Cross Entropy, and a custom loss function that combined the properties of the Dice loss and Categorical Focal loss functions.
3. **Output Activation Function:** Softmax was used as the activation function in the output layer for the multi-class segmentation task.
4. **Learning Rate:** The initialization of the learning rate is a critical factor in facilitating effective model learning. It establishes the scale for learning rate adjustments performed by the adaptive Adam optimizer. In our experiments detailed in **Sections 4.5.3.3** and **4.5.3.3** , we initialized the learning rate at 1e-4. Subsequently, we assessed different learning rates to identify the optimal value. We found that a learning rate smaller than 1e-4 was more effective due to the model's rapid training loss convergence. Therefore, we independently trained the model with the following learning rates: 1e-4, 1e-5, and a dynamic learning rate schedule. This dynamic schedule started at 1e-4 and then decayed by 1e-6 per epoch.
5. **Batch Size:** The batch size plays a crucial role in determining how many scan images are processed in a single training iteration or batch. Our selection of the batch size was based on empirical observations, taking into account various factors, including GPU memory constraints, the risk of model overfitting,

training convergence speed, and the total number of training scans. After careful consideration, we settled on a batch size of 16 as it provided the most balanced trade-off between these considerations.

Hyper-parameters, including the size of convolutional kernels and the number of filters in each convolutional layer, were integrated into the pre-trained backbone's architecture and subsequently adjusted to align with the U-Net's decoder path. This fine-tuning allowed for a tailored configuration to suit our specific segmentation tasks.

Since the Dice Similarity Coefficient (DSC) is our primary performance metric and we aim to maximize it, processes like model selection were fine-tuned to optimize this metric. This included selecting the model with the highest validation DSC.

The hyper-parameter optimization experiments were conducted while adhering to the hyper-parameter configuration and training strategy outlined in **Section 4.5.3.1**.

The empirically optimized set of hyper-parameters comprises a batch size of 16, a dynamic learning rate schedule, a custom loss function combining Dice Loss and Categorical Focal Loss, and the Adam Optimizer.

4.6 Model Training Procedure

To assess the performance of the pretrained resnet34 U-Net, 2D U-Net, 2D/3D U-Net (nnU-Net) deep learning segmentation models in lower limb bone segmentation tasks (tibia, femur, fibula and pelvis), the following training configurations was employed:

- 7 MRI scan volumes, 3020 2D slices, were used to evaluate model performance on the pelvis segmentation task. The dataset was split on a subject-level with 5 3D scan volumes used to train the model, 1 for model validation, and 1 for model testing. Data augmentation was applied to the training set using an augmentation factor of 4. A patient-based leave-one-out cross validation was performed with 50 epochs per fold to determine an unbiased measure of model performance. This was done opposed to 5-fold cross validation as the labeled dataset was relatively small, and it would provide a more robust model performance measure. The dice similarity coefficient (DSC) values and volume error (VError) measures are taken as the mean of the metrics on each of the 7 training instances.

The segmentation learning task was expanded to encompass the pelvic structure in the paediatric population. However, we faced a challenge due to the scarcity of pelvic label data, primarily stemming from the lack of manually segmented data available. This limited our training dataset size to only utilizing 7 of the 30 participants and we delve into this limitation in greater detail in the report's Limitations section (**Section 6.6**).

The hyper-parameter configurations for the pretrained resnet34 U-Net, and 2D U-Net were empirically determined and are stated in **Section 4.5.3.1**. The hyper-parameter configurations for the 2D U-Net and 3D U-Net nnU-Net configurations were optimized through the self configuring framework, and are stated in **Section 6.6**.

To enhance model generalization, we expanded the limited training dataset through data augmentation to prevent overfitting. We employed seven-fold cross-validation (k=7) for unbiased model evaluation. We also applied an early stopping criterion based on validation loss convergence to improve computational efficiency.

This was a critical aspect of our segmentation pipeline, trained on limited labeled data, with potential applications across paediatric populations.

4.7 Post-Processing

We applied post-processing techniques, specifically small artifact removal and hole filling, to the model's outputted segmentations to enhance the quality of the segmentation results. During the examination of segmentation results on a slice-by-slice basis, we observed instances where the model erroneously classified certain areas outside the region of interest as musculoskeletal structures.

To tackle this problem, we developed an adaptive thresholding denoising algorithm. It identifies noise by counting non-background pixels within a region. If this count is below a specified threshold, the pixel is replaced with the background value (0). The threshold adapts to each slice based on the number of pixels in each unique region of interest. This adaptability is crucial for distinguishing small bone structures, especially when the tibia and fibula structures overlap in some axial slices due to their size differences.

Figure 5A illustrates a model output segmentation slice taken near the hip region of a child. In this figure, the white region represents the pelvis, and the gray region represents the femur. Notice the presence of noise generated by misclassified pelvis regions. **Figure 5B** shows the same slice after undergoing post-processing with the denoising algorithm, resulting in the removal of the noisy regions.

Figure 5A: Unprocessed Predicted Segmentation Mask (Pelvis, Femur)

Figure 5B: Post-Processed Predicted Segmentation Mask (Small Artifact Removal Denoising Algorithm)

To address an issue with 3D U-Net nnUNet outputs, we applied a specific post-processing step. The 3D U-Net leverages volumetric data, leading it to partially learn boundaries for the left-side lower limbs despite having only right-side label information. While this demonstrates the power of volumetric information in biomedical segmentation, it exceeded our study's scope since we lacked ground truth labels for the left-side lower limb bone groups. The implementation is as follows:

1. On a subject-level manually identifying dimensionally how far the left limb segmentations span
2. Apply a uniform image crop to remove the noise
3. Apply padding (background labels) to the cropped output to restore slice dimensionality (256x256)
4. Repeatedly apply the steps above to each slice of the segmentation output

The post-processing algorithms developed utilize the ndimage module within the SciPy (1.11.1) library [ref]. The denoising methodology was inspired from a wound segmentation deep learning based study [23].

4.8 Performance Evaluation Metrics

The performance results of experiments were assessed using two evaluation metrics: the DSC, and the volume error (VError) to assess the segmentation accuracy. The DSC and VError assess different properties of the segmentation results. The DSC is a technical metric that assesses the spatial overlap and similarity between the predicted and true segmentation, emphasizing the quality of segmentation shape and position and the VError is a biological/clinical metric that measures the absolute difference in the volume between the predicted and true segmentations, emphasizing the size accuracy of the segmentation.

The DSC is calculated as following:

$$DSC = \frac{2|X \cap Y|}{|X| + |Y|}$$

Where Y is the set of voxels in the predicted segmentation masks, and X is the set of voxels in the ground truth segmentation mask. The value of the DSC is normalized between 0 and 1, where 0 indicates no spatial overlap and 1 indicates a perfect spatial overlap. The equation above models the case of binary class problem, and to evaluate the DSC of a multi-class segmentation mask we simply map the DSC calculation to individual binary class problems for each segmentation unique label encoding.

The VError is calculated as follows:

$$\text{VError} = V * |N_{Predicted} - N_{True}|$$

Where $N_{Predicted}$ is the number of voxels in the predicted segmentation structure, N_{True} is the number of voxels in the ground-truth segmentation structure of the same musculoskeletal group, and V represents the volume of one voxel, which for our Tairawhiti research dataset is 2.64mm^3. The VError ranges from 0 indicating a prefect prediction to a variable upper bound value depending on the volume of the musculoskeletal structure.

4.9 Experimental Environment

All deep learning experiments were conducted using the Google Colaboratory cloud-based platform [ref]. It offers a Python programming environment integrated with Google Drive. The specific computational node used was configured with: 1 Nvidia Tesla T4 GPU (13 GB VRAM), 2 Intel (R) Xeon (R) CPU cores @2.30 GHz, 52 GB RAM, and 108 GB disk space.

The preprocessing and deep learning pipelines were implemented with the following key open-source softwares: Python 3.9.17, Keras 2.13.1, Tensorflow 2.13.0, Segmentation-Models 1.0.1, Nibabel 4.0.2, Sklearn 1.2.2, NumPy 1.23.5, SimpleITK 2.2.1, Pydicom 2.4.2, PIL 9.3.0, and 3D Slicer [ref].

4.10 Segmentation Model Generalizability Across Paediatric Populations

To develop a better understanding of how our model generalizes across paedritic populations our model was applied to an additional dataset of 10 high-quality MRI scans were used from a separate paediatric population for a study based in Auckland, New Zealand. This dataset will be referenced as the Friedlander dataset, and was provided to by our research supervisor, *Dr Julie Choisne*.

The Friedlander dataset contains 10 paediatric hip-down MRI scans, and manual segmentations done using the ITKSnap 3D Imaging Software [43] for the following right lower limb bone groups: tibia, femur, pelvis (right wing), and fibula. Each participant's MRI scan volume contained a variable number of slices varying from 700-900 slices. Each slice in the MRI scan is taken 1 mm apart along the childs's height (slice thickness). The MRI scan data varied in resolution The raw scan data was stored in MRI series sequence files and the mask data as compressed NIfTI files. The MRI series selected was based on which correlated to the mask data, and was converted to a compressed NIfTI file using 3D Slicer [ref]. The following preprocessing steps were applied to the Friedlander MRI scan data: min-max pixel intensity normalization, and uniform image resizing to 256x256. **Figure 6A** is a slice taken near the child's calf and includes the tibia and fibula regions, **figure 6B** is the corresponding multi-class mask, and **figure 6C** is a superimposed representation.

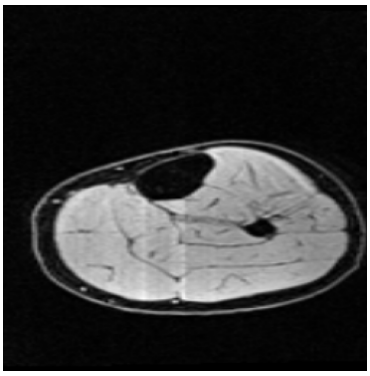

Figure 6A: Friedlander Dataset - MRI Scan Slice (Calf Region)

Figure 6B: Friedlander Dataset - Multiclass Mask (Tibia, Fibula)

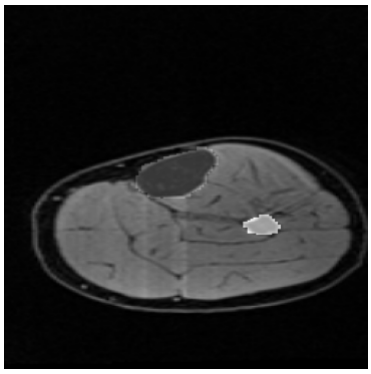

Figure 6C: Friedlander Dataset - Superimposed MRI Scan Slice & Multi-class Mask

The pre-trained resnet34 U-Net model fine-tuned on the Tairawhiti dataset was evaluated on 3 MRI scan volumes from the Friedlander dataset to determine if the model was transferable without the need of additional training/fine-tuning.

It was worth investigating the degree of additional training required to achieve useful segmentation results from the model (DSC > 0.8). The network weight/parameter configuration from the model trained on the Tairawhiti dataset was additionally fine-tuned on the Friedlander data. After each fold in the cross validation process the model weights were reset to ensure no data leakage affected the findings derived. The hyper-parameter configuration for the task is stated in the Transfer Learning Based Approach Methodology section, and used the following training procedure

- 5 raw MRI scans (Friedlander), 3658 2D slices, were used to evaluate model performance on a segmentation task using label data on the tibia, femur, fibula, and pelvis. The dataset was split on a child-level with 3 3D scan volumes used to train the model, 1 for model validation, and 1 for model testing. A child-based LOOCV (k=5) was performed with 20 epochs per fold to determine an unbiased measure of model performance. The dice similarity coefficient (DSC) values and volume error (VError) measures are taken as the mean of the metrics across all 6 folds for each individual bone group for each deep learning model experiment.

4.11 Segmentation Pipeline GUI Tool

We designed a user-friendly GUI for the automatic segmentation tool, tailored to the needs of Tairawhiti paediatric study researchers. This streamlined the deep learning segmentation pipeline, making it accessible without requiring domain or deep learning expertise.. The automatic segmentation tool offers the following features, with visual reference to **Figure 7C** (entire view) in the appendix:

1. **Preprocessing Training Data:** Users can input DICOM scan volumes and binary NIfTI (*nii) masks into folders within a specified base directory. The tool's backend executes our preprocessing pipeline, which generates multi-class segmentation masks from raw binary masks and applies preprocessing steps to the scan pixel intensity data, as detailed in the **Section 4.4**. It then outputs model-ready scan and mask data in a compressed NIfTI (*nii.gz) format. Users can specify the desired image size. **Figure 7A** shows the integration of the preprocessing pipeline into the GUI.
2. **Preprocessing Test Data:** For test data, users can input DICOM scan volumes and specify the approximate slice number where the subject's hip begins, particularly for full-body scans. The backend executes a variation of the preprocessing pipeline, exclusively applying preprocessing steps to the scan data. It then produces the scan data as a compressed NIfTI (*nii.gz) file.
3. **Slice Data Visualization:** Users can input the filename of the preprocessed multi-class mask data and specify the slice number. The backend runs a visualization function, generating slice visualizations of the preprocessed MRI scan slice, preprocessed segmentation mask, and a superimposed representation of the scan and mask slices.
4. **Run Automatic Segmentation Model:** Users input the filename of the preprocessed test data. In the backend, the tool installs the necessary deep learning libraries in the user's environment and processes the test scan using a version of the pretrained Resnet34 U-Net. This produces a segmentation output for musculoskeletal structures, including the tibia, fibula, femur, and pelvis. The segmentation results are provided in the form of a compressed NIfTI (*nii.gz) file for 2D slice information and a PLY file for 3D visualization (refer to Fig X in the appendix for details). **Figure 7B** shows the integration of the Resnet34 U-Net into the GUI.

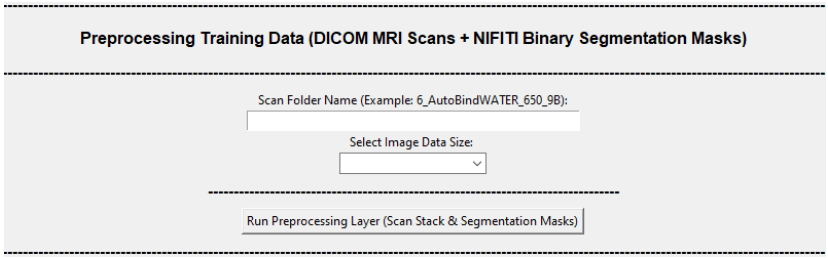

Figure 7A: Preprocessing Medical Data Application in GUI

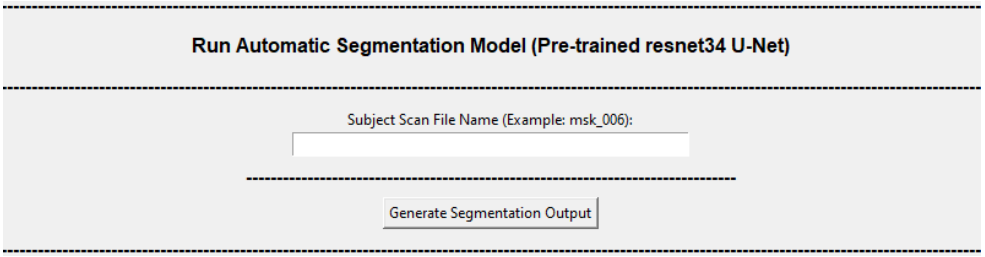

Figure 7B: ResNet34 U-Net Segmentation Model Application in GUI

5.0 Results

5.1 Input Image Dimensionality Experimentation

The image size is a vital parameter for the dimensionality for the input shapes of the scan slices and segmentation masks into the deep learning model. All performance metrics listed were computed using the baseline 2D U-Net, with the defined training strategy and hyper-parameter configurations stated in **Section 4.5.1.** Our results in Table 1 demonstrate that an input size of 256 x 256 yields faster training times and reduced memory usage while maintaining model performance. This improvement enhances the efficiency of our data pipeline, essential for memory-intensive medical data. The reduced memory requirements also enable us to experiment with higher batch sizes, a crucial factor in optimizing hyperparameters to ensure effective learning without underfitting.

Image Size	Average DSC	Average Epoch Duration	Average RAM Utilization
512 x 512	0.74	138 s	60%
400 x 400	0.74	108 s	34%
256 x 256	0.73	79 s	14%

Table 1: Experimental Results with Image Dimensionality

5.2 Transfer Learning Model Experimentation

5.2.1 Pre-Trained vs Randomly Initialized Model Weights

This experiment aimed to empirically evaluate the impact of utilizing pre-trained weights for the encoder network on model performance as opposed to employing randomly initialized encoder network weights. **Section 4.5.3.2** provides a comprehensive description of the methodology used for implementation.

Segmentation Class	Pre Trained Weights	Randomly Initialized Weight
	Mean DSC	Mean DSC
Tibia	0.92	0.89
Femur	0.93	0.90
Fibula	0.72	0.55
Pelvis	0.86	0.83

Table 2A: Experimentation Results using a Pretrained Resnet34 U-Net (ImageNet) vs Randomly Initialized Resnet34 U-Net

The overall mean Dice Similarity Coefficient (DSC) for the pre-trained U-Net model was 0.86, while the randomly initialized U-Net model achieved a mean DSC of 0.79. **Table 2A** outlines the DSC scores for each individual segmentation task in both models.

Comparing the U-Net with a pre-trained encoder to the U-Net with randomly initialized encoder weights, two significant points stand out: segmentation accuracy and the rate of convergence. On average, the U-Net with a pre-trained encoder achieves higher segmentation accuracy across all tasks. The rate of convergence, indicating how quickly the model loss converges, is also noteworthy. **Figure 8A** illustrates the training and validation losses for the pre-trained U-Net, while **figure 8B** shows the losses for the U-Net with randomly initialized weights. It's evident that the pre-trained U-Net exhibits faster loss convergence.

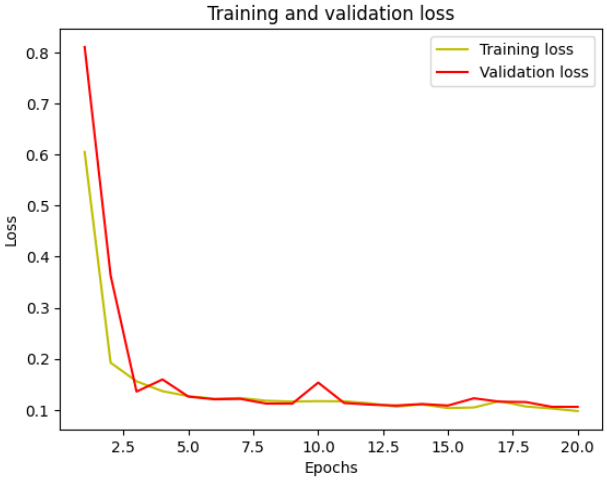

Figure 8A: Plot showing Training/Validation Loss for Pretrained Resnet34 U-Net

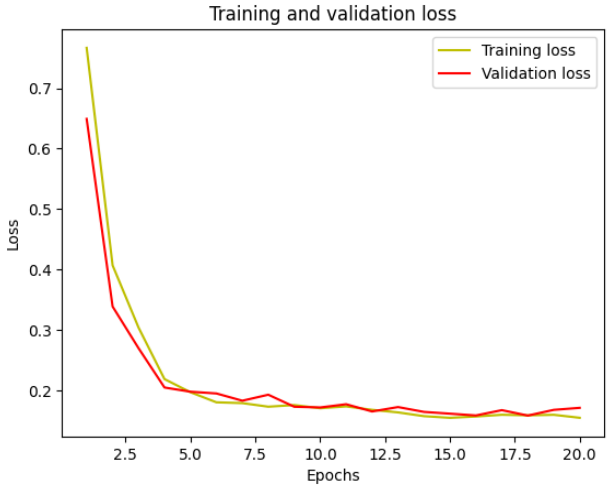

Figure 8B: Plot showing Training/Validation Loss for Randomly Initialized Resnet34 U-Net

5.2.2 Pre-Trained Deep Learning Backbone Architectures

The objective of this experiment was to identify the most suitable pre-trained backbone model for substituting the U-Net's encoder network. The experiment encompassed four pre-trained backbone architectures: Resnet34, Resnet50, VGG19, and MobilenetV2, all initialized with pre-trained weights from the ImageNet [ref] dataset.

Backbone Architecture	Mean DSC
Resnet34	0.86
Resnet50	0.86
VGG19	0.75
MobilenetV2	0.76

Table 2B: Experimentation Results using different Pretrained Backbones Models (ImageNet) with the U-Net

The experimental results indicate that both the Resnet34 and Resnet50 backbone architectures exhibit the highest segmentation accuracy. When their performance is comparable, the preference is for the simpler and more memory-efficient architecture, in line with the model's compactness principle. Consequently, the transfer learning approach employs the Resnet34 backbone architecture as the encoder network for the U-Net, considering that Resnet34 is a 34-layer residual neural network while Resnet50 comprises 50 layers.

5.2.3 Network Hyper-parameter Optimization

The following details how hyper-parameter configurations for the choice of function, and learning rate schedule were empirically optimized for our deep learning implementation:

1. **Table 2C** provides a breakdown of segmentation accuracy for the Resnet34 pre-trained U-Net model trained with three specified loss functions. Notably, the model trained with the custom loss function outperformed the others by a significant margin. Given the prevalent class imbalance in our segmentation mask data, where the background class (0) predominates, the focal loss function demonstrates its efficacy in addressing class imbalance. It achieves this by assigning higher weights to underrepresented classes, which likely accounts for its superior performance when used in combination with the dice loss function.

Loss Function	**Mean DSC**
Dice	0.86
Categorical Cross Entropy	0.80
Dice + Categorical Focal (Custom)	0.88

Table 2C: Experimentation Results of Pretrained Resnet34 U-Net with different Loss Functions

2. In our experiments comparing pre-trained and randomly initialized Resnet34 U-Net models, as depicted in **Figures 8A** and **8B**, we observed swift convergence. This rapid convergence may be attributed to an initial learning rate that is possibly too large for the learning task. Consequently, it suggests that the optimal learning rate initialization is smaller than what was employed in the transfer learning-based experiments.

   The model was trained independently using various learning rates: 1e-4, 1e-5, and a dynamic learning rate schedule that started at 1e-4 and progressively decayed by 1e-6 per epoch. When assessing the impact of different learning rate configurations on segmentation accuracy, we found that both the dynamic learning rate schedule and a learning rate of 1e-4 yielded better performance, as evaluated through Leave-One-Out Cross-Validation (LOOCV).

   The dynamic learning rate schedule, starting at 1e-4 and gradually decreasing by 1e-6 per epoch, contributed to a more stable training process, as it promotes progressive lowering of the learning rate, enhancing performance. **Figure 9A** illustrates the rate of convergence plot for the model trained with the dynamic learning rate schedule, which, compared to **Figure 9B**, exhibits slightly slower convergence. This is beneficial for the final implementation as it allows for more exploration of solutions in the earlier

epochs.

Figure 9B represents the rate of convergence plot for the model trained with a learning rate of 1e-5, demonstrating its computational expense in terms of training time, as it requires over 100 epochs to achieve loss convergence for each fold.

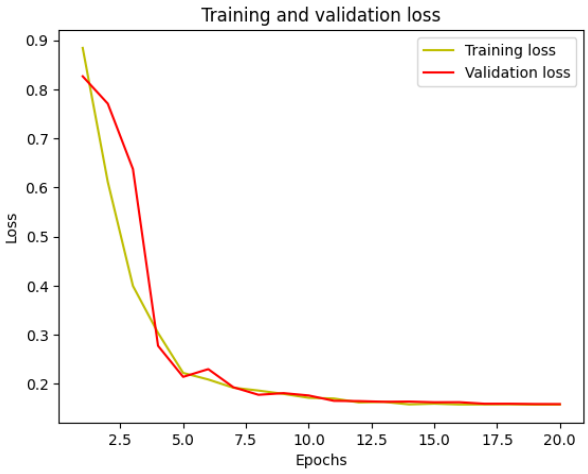

Figure 9A: Plot showing Training/Validation Loss for Pretrained Resnet34 U-Net with Learning Rate = Dynamic Schedule

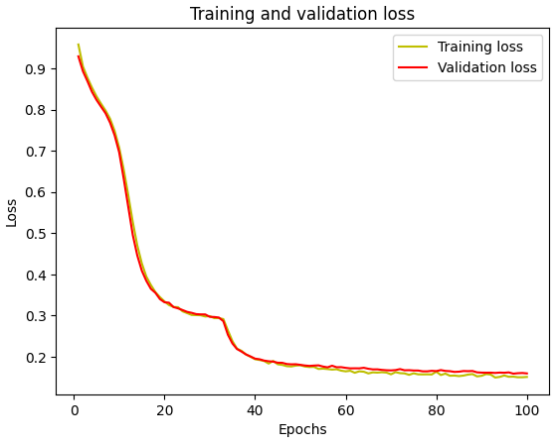

Figure 9B: Plot showing Training/Validation Loss for Pretrained Resnet34 U-Net with Learning Rate = 1e-5

Learning Rate Initialization	**Mean DSC**
1e-4	0.86
1e-5	0.75
Dynamic Learning Rate Schedule	0.86

Table 2D: Experimentation Results of Pretrained Resnet34 U-Net with different Learning Rates

5.3 Deep Learning Segmentation Model Results

	2D U-Net (nnU-Net)		3D U-Net (nnU-Net)		Pretrained Resnet34 U-Net		2D U-Net	
	DSC	**VE(cm^3)**	**DSC**	**VE(cm^3)**	**DSC**	**VE(cm^3)**	**DSC**	**VE(cm^3)**
Tibia	0.89 ± 0.03	18.3 ± 3.4	0.90 ± 0.03	9.6 ± 2.2	0.93 ± 0.02	5.4 ± 0.9	0.78 ± 0.02	25.0 ± 6.2
Femur	0.92 ± 0.03	1.1 ± 0.8	0.91 ± 0.03	0.9 ± 0.2	0.95 ± 0.01	0.9 ± 0.2	0.83 ± 0.03	8.3 ± 2.3
Fibula	0.75 ± 0.03	3.4 ± 1.6	0.75 ± 0.06	5.4 ± 2.6	0.78 ± 0.04	2.9 ± 1.2	0.64 ± 0.03	8.8 ± 3.0
Pelvis	0.84 ± 0.04	31.8 ± 5.1	0.70 ± 0.05	43.3 ± 5.2	0.88 ± 0.03	24.5 ± 3.9	0.68 ± 0.03	53.4 ± 8.1

Table 3: Performance Evaluation of Deep Learning Segmentation Models

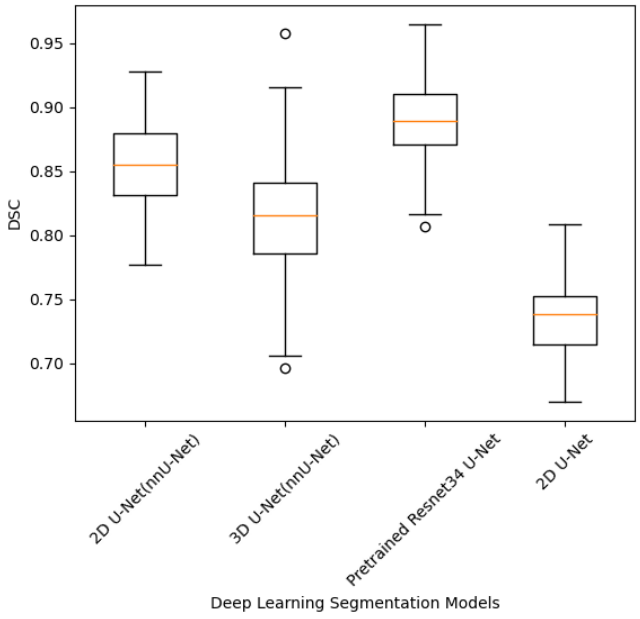

Figure 10: Plot showing Distribution of DSC in Multi-Class Segmentation Task for each Deep Learning Segmentation Model

The deep learning segmentation models were developed and subsequently assessed to identify the model that yields the highest segmentation accuracy for multi-class segmentation, employing the training strategy described in **Section 4.6**. In **Table 3**, we present the performance metrics of Dice Similarity Coefficient (DSC) and Volume Error (VE) for each segmentation model. These models were evaluated using a seven-fold cross-validation (LOOCV), and we provide the mean and standard deviation of segmentation accuracies for each segmentation class, corresponding to each model.

Figure 10 complements the analysis with a visual representation of the segmentation results, consolidated into a multi-class view. Within this figure, each box and whisker section of the plot illustrates the performance of the model across the four segmentation tasks. **Figure 13** in the appendix is a 3D visualisation of the segmentation output from the Resnet34 U-Net model.

Overall, the pretrained resnet34 U-Net model (DSC = 0.89, VError = 8.4 cm^3) demonstrated the best segmentation accuracy on every single individual bone segmentation task, and as a multi-class segmentation model. The 2D U-Net configuration using the nnU-Net framework (DSC = 0.85, VError = 13.7 cm^3) demonstrated relatively similar model performance. The 3D U-Net configuration using the nnU-Net framework (DSC = 0.82, VError = 14.8 cm^3) was the third best performing model, however it showed high variability in model performances across the segmentation tasks, and this is potentially because of insufficient distinct participant training data to capture the relevant volumetric information. Finally, our 2D U-Net (DSC = 0.73, VError = 23.9 cm^3) implementation showcased the worst performance of the studied segmentation models for potential reasons such as poor feature extraction due to simple architectural design or insufficient training time.

All our deep learning segmentation models performed poorly on the fibula segmentation task, and was validated by poor annotation quality for the fibula manual segmentations. This aspect will be further discussed in the limitations sections (**Section 6.6**) of the report.

5.4 Segmentation Model Generalizability Across Paediatric Populations Results

Our first experiment in exploring model generalizability involved using the pretrained resnet34 U-Net model that had been fine-tuned on the Tairawhiti dataset and without exposing the model to the Friedlander dataset determining how it would perform. The mean segmentation accuracy (DSC) across tibia, femur, fibula and pelvis segmentation tasks was 0.23, 0.32, 0.08, and 0.03 respectively. This proved the model required a degree of exposure to the Friedlander data to be effective.

Our next experiment exposed the pretrained resnet34 U-Net model (pretrained on ImageNet and Tairawhiti dataset) to the Friedlander imaging data, and used 5 children MRI scan volumes to fine-tune, validate and test the model on. **Section 4.6** defines the training strategy and hyper-parameter configuration used to produce the results in **Table 4**.

The model exhibits a high degree of transferability with minimal training, as evidenced by performance metrics. Each segmentation task achieved a Dice Similarity Coefficient (DSC) greater than 0.8, and this assessment was conducted rigorously using LOOCV with data leakage protocols to ensure the validity of our findings.

Pretrained ResNet34 U-Net (ImageNet + Tairawhiti Data)				
	Tibia	**Femur**	**Fibula**	**Pelvis**
DSC	0.93 ± 0.01	0.95 ± 0.03	0.87 ± 0.02	0.81 ± 0.02

Table 4: Model Generalizability Experimental Performance Evaluation of Deep Learning Segmentation Models on Friedlander Dataset

5.4 Computational Performance

We conducted an evaluation of the segmentation models from a computational performance perspective, a crucial aspect for the practical use of these models in clinical applications.

Table 5 provides an overview of the count of trainable model weights for each segmentation model. For a deeper understanding of the distribution of trainable and non-trainable parameters in the pre-trained U-Net, please refer to **Figure 11** in the appendix. In this context, trainable parameters (3M) pertain to the decoder weights fine-tuned during training, while non-trainable parameters (24M) refer to the encoder weights derived from the Resnet34 backbone architecture, which are kept fixed during training. For the other deep learning models under investigation, they employ randomly initialized weights, making all network parameters trainable.

Table 5 also details information regarding the time required for predicting or inferring on an MRI scan volume. Notably, the pretrained Resnet34 U-Net model demonstrated rapid inference times, coupled with its relative compatibility. These inference times were assessed on the same participant MRI scan volume and conducted using the same computational resources to ensure fairness.

	Pretrained ResNet34 U-Net	**2D U-Net (nnU-Net)**	**3D U-Net (nnU-Net)**	**2D U-Net**
Trainable Parameters	3M	45M	62M	31M
Inference Time (GPU)	20s	180s	540s	40s

Table 5: Computational Performances of Deep Learning Models

6.0 Discussion

6.1 Interpretation of Results

The pretrained Resnet 34 U-Net model (DSC = 0.89, VError = 8.4 cm^3) demonstrated the best performance for the segmentation of individual lower limb musculoskeletal structure from MRI scans on the Tairawhiti dataset. This aligns with the goal of our study to develop a deep learning based segmentation model that outperforms our established baseline state-of-the-art nnUNet segmentation framework. Our 2D U-Net model did not perform as expected and this can be attested to the relative simplicity in the network architecture. Experimentation with the 2D U-Net model in this study has qualitatively highlighted the importance of the network's feature extraction capabilities, as both the pre-trained U-Net and the nnUNet's 2D U-Net configuration have relatively more complex encoder structures (number of convolutional blocks and layer parameterisation). We are able to conclude from the evaluation of model performance that fully automatic segmentation of individual musculoskeletal structures is feasible with the use of transfer learning based deep learning models.

6.2 Comparison to Existing Literature

In our comparative analysis with a previous study on lower limb musculoskeletal segmentation [5], several key findings and differences emerged. The study provided valuable insights and a benchmark for our research.

Model performance was evaluated with a different training and hyper-parameter configuration, however the most important factor would be the size of the training dataset and performance evaluation method. They use relatively more training data with 20 3D scan volumes used to train their state-of-the-art H-DenseUNet CNN model from scratch, and k-fold (k=5) cross validation to derive their performance metrics.

Our implementation of a pretrained Resnet34 U-Net (DSC = 0.92, VError = 5.4cm^3) outperforms the H-DenseUNet (DSC = 0.89, VError = 24.8cm^3) for the tibia segmentation task. For the fibula segmentation task, the H-DenseUNet (DSC = 0.89, VError = 3.4cm^3) outperforms our model (DSC = 0.78, VError = 2.9cm^3).

The tibia segmentation task performance comparison was promising, and it highlights how our model performs against the H-DenseUNet, a state-of-the-art biomedical segmentation model we analyzed in the literature review of our study. Our model underperformed on the fibula segmentation task, as the fibula manual segmentation data was poorly annotated and this will be further discussed in the limitations section of the report.

In our comparative analysis with a study [14] focusing on pelvis segmentation from CT scans, several key findings and differences became apparent. To understand these differences, we delved into their methodology and training process, which revealed that their model was trained on a dataset of 10 CT scan volumes over 10 training epochs. Performance evaluation was conducted on a single fold, comprising a test set of 5 CT scan volumes.

Notably, on the pelvis segmentation task, their implementation of a Resnet50 U-Net (DSC = 0.96) outperformed our pretrained Resnet34 U-Net (DSC = 0.88). It is important to recognize that our pelvis segmentation task involved the MRI modality and utilized a training dataset of 5 MRI scan volumes per fold. The choice of medical imaging modality significantly influences the model's feature extraction capability. Segmentation of bone structures from MRI scans is inherently more complex due to non-unique pixel intensity differences between bone and tissue [24].

Moreover, the number of training scan volumes plays a crucial role in achieving segmentation accuracy, particularly for complex musculoskeletal structures like the pelvis. A more extensive representation of labeled data for the pelvis segmentation task could potentially enhance our model's performance in comparison to the Resnet50 U-Net from the referenced research.

6.3 Model Generalizability on Paedriatic Populations

Our experimentation assessed the potential of transferring a pretrained Resnet34 U-Net model, initially trained on ImageNet and the Tairawhiti Dataset, into another paediatric population. The results suggest that direct transfer is not straightforward due to variations in MRI scan data quality, population characteristics, and field of view in the Friedlander dataset. Despite these challenges, the model demonstrated strong adaptability when exposed to a limited sample from the Friedlander dataset.

Various components of our pipeline, including preprocessing, post-processing, and hyper-parameter configuration, proved to be easily transferable. However, direct transfer of the pretrained segmentation model to the Friedlander data presented challenges. Yet, fine-tuning the model with exposure to 4 participant MRI scan volumes led to impressive performance on multiple bone segmentation tasks. This fine-tuning process exhibited rapid convergence and produced robust, unbiased performance metrics. It underscored the suitability of the model's initial weights for the task, indicating a strong starting point for learning the dataset, as the model was pretrained on Tairawhiti data of the same modality.

Our exploration of generalizability in paediatric populations highlights the effectiveness of transfer learning methodologies in biomedical segmentation tasks. We ensured the reliability of our findings by employing robust performance evaluation methods and data leakage protocols. **Figure 12** in the appendix is a 2D representation of the segmentation results using the Friedlander fine-tuned Resnet34 U-Net model.

As achieving generalizability was an objective of our deep learning segmentation pipeline, our choices for the data pipeline and architectural design were influenced accordingly. Techniques such as data augmentation, architectural features like dropout layers for regularization and batch normalization, and hyper-parameterization with a modest batch size of 16 were integrated. Additionally, early stopping criteria were applied to prevent overfitting to our specific paediatric dataset.

The predominant factor contributing to achieving generalizability was the utilization of transfer learning. The model's weights encompassed knowledge from both extensive datasets and the specific domain of paediatric lower limb MRI, making it a potent contributor to our results.

6.4 Computational Performance

Our pretrained Resnet34 U-Net outperformed the state-of-the-art nnU-Net, while also being more compact in model design. In the context of clinical applications for deep learning segmentation tasks, model compactness is a pivotal design consideration.

The number of trainable model parameters serves as a measure of model complexity and plays a vital role in assessing model compactness and efficient utilization of computational resources. In this regard, the pretrained Resnet34 U-Net model showcased the most robust computational performance, with the shortest inference times and the smallest number of trainable parameters.

Another noteworthy aspect is the pretrained Resnet34 U-Net's ability to perform inferences without the need for a GPU within a reasonable timeframe. In contrast, the nnU-Net framework necessitates a GPU for inference, which can be a limitation in scenarios where computational resources are constrained.

6.5 Clinical Relevance

The successful implementation of our fully automated segmentation model within a paediatric study in rural New Zealand represents a significant advancement. This study addresses the critical need to better understand paediatric musculoskeletal structures and functions in medical imaging, particularly in rural areas. By automating musculoskeletal segmentation, our deep learning pipeline significantly reduces the labor-intensive aspects of this process, saving valuable time. The integration of a paediatric-specific deep learning model has the potential to greatly enhance research efficiency and accelerate investigations into paediatric musculoskeletal disorders. Additionally, it can alleviate constraints in clinical settings by enabling faster processing of medical imaging data, ultimately facilitating quicker and more accurate patient diagnoses.

Our user-friendly Graphical User Interface (GUI) for automatic segmentation simplifies the complex segmentation pipeline, benefiting clinical and paediatric research applications. Tailored to the Tairawhiti paediatric study, it offers an intuitive design that streamlines data preprocessing, model execution, and visualization, making it accessible even to non-technical users. Notably, it operates efficiently without the need for high GPU compute power, reducing barriers to entry for smaller research groups. This tool enhances the overall workflow, saving time and facilitating musculoskeletal segmentation tasks, ultimately advancing research objectives in the Tairawhiti paediatric study and similar projects.

6.6 Limitations

Within our study we encountered limitations in regard to the modeling and data aspects of the work.

The scarcity of labeled data for pelvis segmentation posed a significant challenge in our project. We had to make a critical decision between narrowing the project's scope by excluding pelvis segmentation and focusing on training a more robust model with a larger dataset, or broadening our segmentation model to include the pelvis while limiting the amount of paediatric MRI data for training. We chose to incorporate the pelvis segmentation task, given its importance in the Tairawhiti paediatric study. To ensure the reliability of our results with limited data, we employed robust model evaluation techniques, specifically a 7-fold cross-validation (LOOCV). The pelvis is a complex and sexually dimorphic musculoskeletal structure, and increasing the labeled data available for this task would significantly benefit the model's performance.

The quality of manual fibula segmentations presented a significant challenge in our study. The suboptimal nature of these annotations made it difficult to accurately evaluate the performance of our deep learning model on this segmentation task. While it's possible to assess the inherent variability in manual segmentations through inter-rater DSC measures, this level of analysis was beyond the scope of our study. Our research supervisor, Dr. Julie Choisne, verified the low quality of these annotations. Fig X in the appendix provides a 3D visualization illustrating inconsistencies in surface details and partially missing components, which further compounded the challenges in fibula segmentation.

6.7 Future Work

The following, outlines key areas where further research and development can contribute to the evolution of deep learning-based segmentation pipeline and address the limitations discussed earlier.

In addressing the limitation of poor fibula annotation quality, future efforts should focus on strategies to enhance the accuracy of the manual segmentation process. Collaboration with domain experts, including researchers in the Tairawhiti paediatric study [ref] and our research supervisor, Dr. Julie Choisne, could prove valuable in achieving more precise segmentations. Through the combined expertise of these professionals, it may be possible to minimize inconsistencies and enhance the overall quality of annotations.

To address the issue of data scarcity, future research endeavors should proactively explore methods to acquire or generate additional labeled data for the pelvis region. A promising approach would involve collaboration with domain experts to expand the dataset through manual segmentation. This collaborative effort can help mitigate the limitation of data scarcity and contribute to the robustness of the model.

The following future research directions detail potential improvements and advancements to the deep learning segmentation pipeline.

Simplifying Binary Mask Generation: Our current process involves manual binary mask creation using the open-source software 3D Slicer. A potential avenue for improvement could involve automating this binary mask creation process and integrating it into the preprocessing pipeline. This enhancement might entail investigating the functionalities of the 3D Slicer API [25] or studying their code repository to replicate their binary label map function [26]. Such automation could enhance workflow efficiency in future studies.

Direct Transferability of the Segmentation Model: The deep learning segmentation pipeline aims to generalize on scans imputed into the model on an annual basis of the same Tairawhiti paediritic population as they develop anatomically. Currently, we are unsure of how the model would handle more developed musculoskeletal features in these children, and this is only determinable with that data available. However, within future improvements made to the model we would aim to make it more robust to small perturbations in the imaging process. Marginal changes in scan contrasts are very common occurrences in medical imaging tasks, and our model currently does not account for this in the augment phase. The future scope of the study would look towards building a more robust model through other augmentation means but more importantly developing a better understanding of the medical imaging domain to address the challenges.

Our model requires user input on inference MRI scan volume that would not have manual segmentation data available specifying for the starting scan slice of the participant’s hip region to perform a task specific preprocessing step subsetting the lower limb regions of the 3D scan volume. This would require someone with domain knowledge to visually inspect the MRI scan volumes and determine an approximate starting slice. We account for the error to a degree by feeding the data some noise around these regions to simulate user variability, however future work would look into a more robust solution to this.

7.0 Conclusion

In conclusion, our study underscores the effectiveness of employing deep learning algorithms for the automatic segmentation of musculoskeletal structures in MRI scans. Through a comprehensive investigation of four deep learning architectures, including resnet34 U-Net, 2D U-Net (nnUNet framework), 3D U-Net (nnUNet framework), and 2D U-Net, we have identified the resnet34 U-Net as the model with the highest segmentation accuracy across all bone groups of interest (DSC = 0.89, VError = 8.4 cm^3). Notably, this performance is comparable to state-of-the-art biomedical segmentation models, such as the H-DenseUNet [ref], and even surpasses the capabilities of the nnUNet self-configuring framework on our specific dataset.

Our study demonstrates the power of transfer learning in biomedical segmentation, particularly when computational resources are limited. Unlike CNN methods trained from scratch, transfer learning remains underexplored in this field [21]. Our research provides compelling evidence of its potential to revolutionize biomedical imaging.

The deep learning segmentation pipeline we have developed contributes to research into deep learning based approaches to biomedical segmentation, and transfer learning studies. The developments benefit the research being done in the Tairawhiti paedatric imaging study, by developing an pipeline for the automatic segmentation of musculoskeletal structures for their participant population, aviating the time-intensive task of manual segmentation task. Fulfilling the objectives in the future work for this study, primarily model generalizability, would extend the scope of contributions into the clinical setting helping improve diagnosis delivery times in healthcare.

In our study we tackled the research objectives set including the following:

- We developed a data preprocessing pipeline that handles the medical data and outputs trainable data for our deep learning segmentation models.
- We developed multiple deep learning segmentation models, evaluating their experimental performances against each other. The Resnet34 U-Net developed in the study showcased the best performance segmenting the following bone groups from paedratic MRI scans: tibia, femur, fibula and femur.
- We prepared groundtruth manual segmentations for the pelvis segmentation task under the expert guidance of *Dr Julie Choisne,* in conjunction with project partner, *Pranav Rao.*
- We validated our Resnet34 U-Net segmentation model by outperforming the state-of-the-art nnU-Net segmentation framework on our dataset, and demonstrating competitive performances with other deep learning models detailed in **Section 6.2**.
- We validated the generalizability of the deep learning segmentation model using the Friedlander dataset, and it demonstrated strong transferability properties, achieving DSC measures above 0.80 on all segmentation tasks with minimal exposure to fine-tuning on the Friedlander data.

As we conclude this research, we encourage further exploration into the realm of deep learning and transfer learning for biomedical segmentation. Our findings emphasize the potential for enhancing the quality and efficiency of medical image analysis. We call on researchers and practitioners to embrace these transformative techniques and work towards their integration into clinical practice. In doing so, we can significantly impact the future of healthcare, offering faster diagnoses, more accurate treatment plans, and ultimately, improved patient outcomes.

8.0 References

1. Boutillon, A., Borotikar, B., Burdin, V., & Conze, P.-H. (2022). Multi-structure bone segmentation in pediatric MR images with combined regularization from shape priors and adversarial network. Artificial Intelligence in Medicine, 132, 102364. https://doi.org/10.1016/j.artmed.2022.102364
2. Liu, X., Song, L., Liu, S., & Zhang, Y. (2021). A review of deep-learning-based medical image segmentation methods. Sustainability, 13(3), 1224. https://doi.org/10.3390/su13031224
3. Sistaninejhad, B., Rasi, H., & Nayeri, P. (2023). A review paper about deep learning for medical image analysis. Computational and Mathematical Methods in Medicine, 2023, 7091301. https://doi.org/10.1155/2023/7091301
4. Kumar, A. (2019, October 29). Understanding U-Net. Towards Data Science. Retrieved from https://towardsdatascience.com/understanding-u-net-61276b10f360
5. Zhu J, Bolsterlee B, Chow BVY, Cai C, Herbert RD, Song Y, Meijering E. Deep learning methods for automatic segmentation of lower leg muscles and bones from MRI scans of children with and without cerebral palsy. NMR Biomed. 2021 Dec;34(12):e4609. doi: 10.1002/nbm.4609. Epub 2021 Sep 21. PMID: 34545647.
6. Astono, I. P., Welsh, J. S., Chalup, S., & Greer, P. (2020). Optimisation of 2D U-Net model components for automatic prostate segmentation on MRI. Applied Sciences, 10(7), 2601. https://doi.org/10.3390/app10072601
7. Takafumi Nemoto, Natsumi Futakami, Masamichi Yagi, Atsuhiro Kumabe, Atsuya Takeda, Etsuo Kunieda, Naoyuki Shigematsu, Efficacy evaluation of 2D, 3D U-Net semantic segmentation and atlas-based segmentation of normal lungs excluding the trachea and main bronchi, Journal of Radiation Research, Volume 61, Issue 2, March 2020, Pages 257–264, https://doi.org/10.1093/jrr/rrz086
8. Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18 (pp. 234-241). Springer International Publishing.
9. Liu, X., Sun, Z., Han, C. et al. Development and validation of the 3D U-Net algorithm for segmentation of pelvic lymph nodes on diffusion-weighted images. BMC Med Imaging 21, 170 (2021). https://doi.org/10.1186/s12880-021-00703-3
10. Yang, Z. (2021). A novel brain image segmentation method using an improved 3D U-net model. Scientific Programming, 2021, 1-10.
11. Isensee, F., Jaeger, P.F., Kohl, S.A.A. et al. nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation. Nat Methods 18, 203–211 (2021). https://doi.org/10.1038/s41592-020-01008-z
12. Isensee, F., Petersen, J., Klein, A., Zimmerer, D., Jaeger, P. F., Kohl, S., ... & Maier-Hein, K. H. (2018). nnu-net: Self-adapting framework for u-net-based medical image segmentation. arXiv preprint arXiv:1809.10486.
13. Karimi, D., Warfield, S. K., & Gholipour, A. (2021). Transfer learning in medical image segmentation: New insights from analysis of the dynamics of model parameters and learned representations. Artificial Intelligence in Medicine, 116, 102078. https://doi.org/10.1016/j.artmed.2021.102078
14. Krawczyk, Z., & Starzynski, J. (2021). Segmentation of bone structures with the use of deep learning techniques. Bulletin of the Polish Academy of Sciences Technical Sciences, 69(1), e136751. https://doi.org/10.24425/bpasts.2021.136751

15. Hassanpour, N., & Ghavami, A. (2023). Deep Learning-based Bio-Medical Image Segmentation using UNet Architecture and Transfer Learning. arXiv preprint arXiv:2305.14841.
16. Pydicom: An open source package for working with DICOM files. (n.d.). Retrieved October 13, 2023, from https://pydicom.github.io/
17. Python 3.9.0. (n.d.). Retrieved October 13, 2023, from https://www.python.org/downloads/release/python-390/
18. Safe Software. (n.d.). NIfTI (Neuroimaging Informatics Technology Initiative) Reader/Writer. Retrieved October 13, 2023, NIfTI (Neuroimaging Informatics Technology Initiative) Reader/Writer (safe.com)
19. Anwai, A. (2021). Brain-Tissue-Seg: A 2D U-Net based segmentation model for brain tissue segmentation. Retrieved October 13, 2023, https://github.com/anwai98/Brain-Tissue-Seg
20. Isensee, F., Jaeger, P. F., Kohl, S. A. A., Petersen, J., Maier-Hein, K. H., & MIC-DKFZ. (2023). nnUNet: Self-adapting framework for U-Net-based medical image segmentation. GitHub. https://github.com/MIC-DKFZ/nnUNet
21. Kim, H.E., Cosa-Linan, A., Santhanam, N. et al. Transfer learning for medical image classification: a literature review. BMC Med Imaging 22, 69 (2022). https://doi.org/10.1186/s12880-022-00793-7
22. AI Plus. (2022, September 27). What is the Adam optimizer and how is it used in machine learning? AI Plus Info. https://www.aiplusinfo.com/blog/what-is-the-adam-optimizer-and-how-is-it-used-in-machine-learning/
23. Wang, Chuanbo, "Medical Image Segmentation with Deep Learning" (2020). Theses and Dissertations. 2434. https://dc.uwm.edu/etd/2434
24. Kuiper, R.J.A., Colaris, J.W., Stockmans, F. et al. Impact of bone and cartilage segmentation from CT and MRI on both bone forearm osteotomy planning. Int J CARS (2023). https://doi.org/10.1007/s11548-023-02929-8
25. 3D Slicer. (2023). 3D Slicer API. https://slicer.readthedocs.io/en/latest/developer_guide/api.html
26. 3D Slicer. (2023). 3D Slicer: A multi-platform, open source software package for visualization and medical image computing. GitHub. https://github.com/Slicer
27. Kumar, H., Green, R., Cornfeld, D. M., Condron, P., Emsden, T., Elsayed, A., ... & Wilson, G. A. (2023). Roadmap for an imaging and modelling paediatric study in rural NZ. Frontiers in Physiology, 14, 172.
28. Ronneberger, O., Fischer, P., & Brox, T. (n.d.). U-Net: Convolutional networks for biomedical image segmentation. University of Freiburg. Retrieved October 13, 2023, from https://lmb.informatik.uni-freiburg.de/people/ronneber/u-net/
29. Viso AI. (2020, October 28). VGG: Very deep convolutional networks. https://viso.ai/deep-learning/vgg-very-deep-convolutional-networks/
30. Intel Corporation. (2023). Resnet-34-pytorch. OpenVINO™ Toolkit. https://docs.openvino.ai/2023.1/omz_models_model_resnet_34_pytorch.html
31. Keras. (n.d.). InceptionV3. https://keras.io/api/applications/inceptionv3/
32. Keras. (n.d.). MobileNet, MobileNetV2, and MobileNetV3. https://keras.io/api/applications/mobilenet/
33. ImageNet. (n.d.). ImageNet. http://www.image-net.org/
34. Andrew. (2020, April 29). DRIVE: Digital retinal images for vessel extraction. Kaggle. https://www.kaggle.com/datasets/andrewmvd/drive-digital-retinal-images-for-vessel-extraction
35. Machine Intelligence Laboratory. (n.d.). Stradview. University of Cambridge. https://mi.eng.cam.ac.uk/Main/StradVie
36. NiBabel. (n.d.). NiBabel: Access a cacophony of neuro-imaging file formats. https://nipy.org/nibabel/

37. NumPy. (n.d.). NumPy. https://numpy.org/
38. Clark, J. A., & Contributors. (2023). Pillow: The friendly PIL fork. https://pypi.org/project/Pillow/
39. OpenCV. (n.d.). OpenCV: Open source computer vision library. https://opencv.org/
40. Yakubovskiy, P. (n.d.). Tutorial. Segmentation models. Retrieved October 13, 2023, from https://segmentation-models.readthedocs.io/en/latest/tutorial.html
41. Chollet, F. (n.d.). Keras: The Python deep learning API. https://keras.io/
42. TensorFlow. (n.d.). An end-to-end open source machine learning platform for everyone. https://www.tensorflow.org/
43. Gerig, G., Yushkevich, P., Hao, J., Pouch, A., & Ravikumar, S. (n.d.). ITK-SNAP: A software application for segmentation of biomedical images. http://www.itksnap.org/pmwiki/pmwiki.php
44. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition (pp. 770-778).

9.0 Appendices

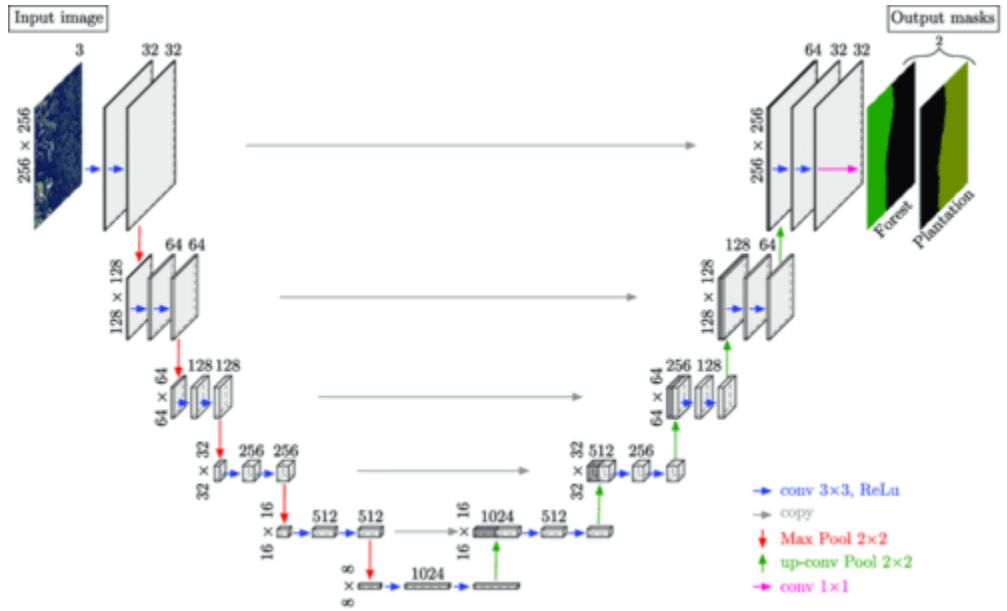

Figure 3: 2D U-Net Architecture

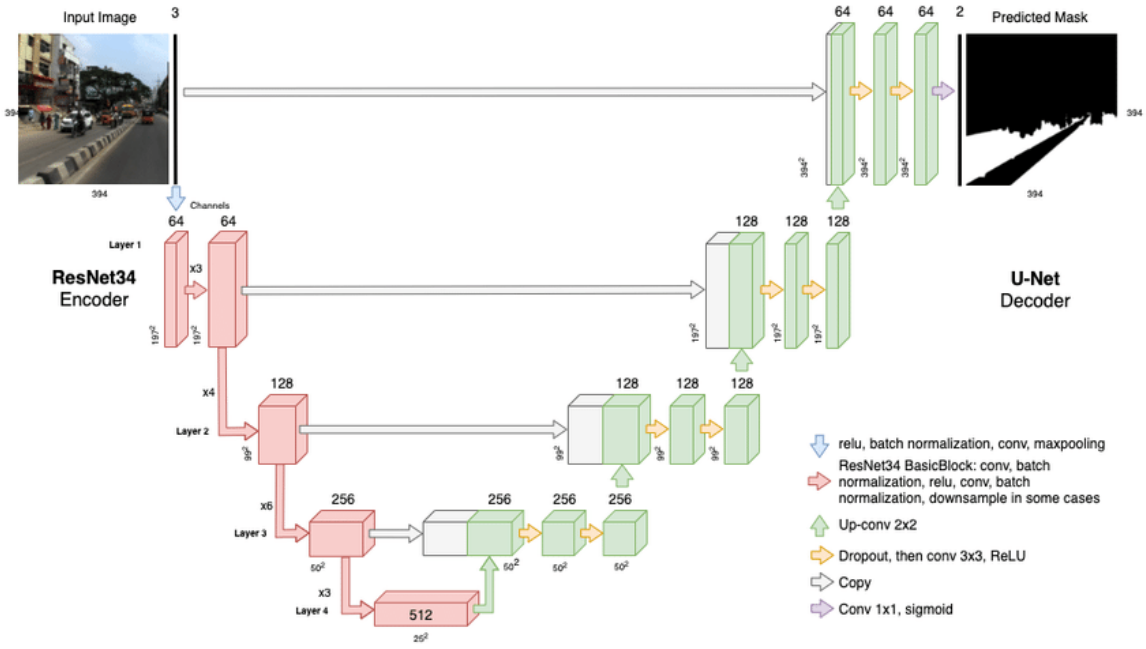

Figure 4: Resnet34 U-Net Architecture

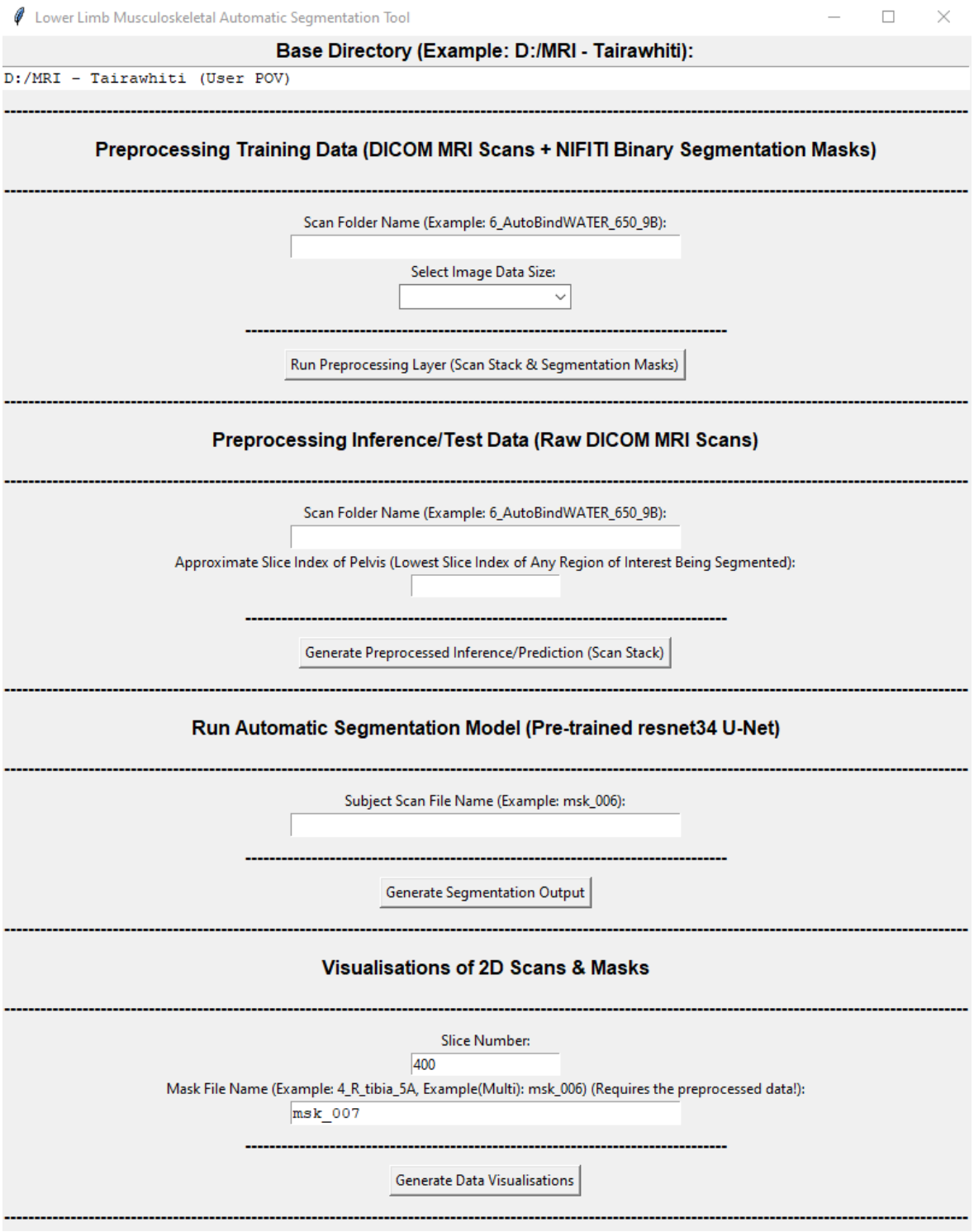

Figure 7C: Lower Limb Musculoskeletal Automatic Segmentation Tool

```
Total params: 24456734 (93.30 MB)
Trainable params: 3167640 (12.08 MB)
Non-trainable params: 21289094 (81.21 MB)
```

Figure 11: Model Parameter Architecture Summary for Resnet34 U-Net

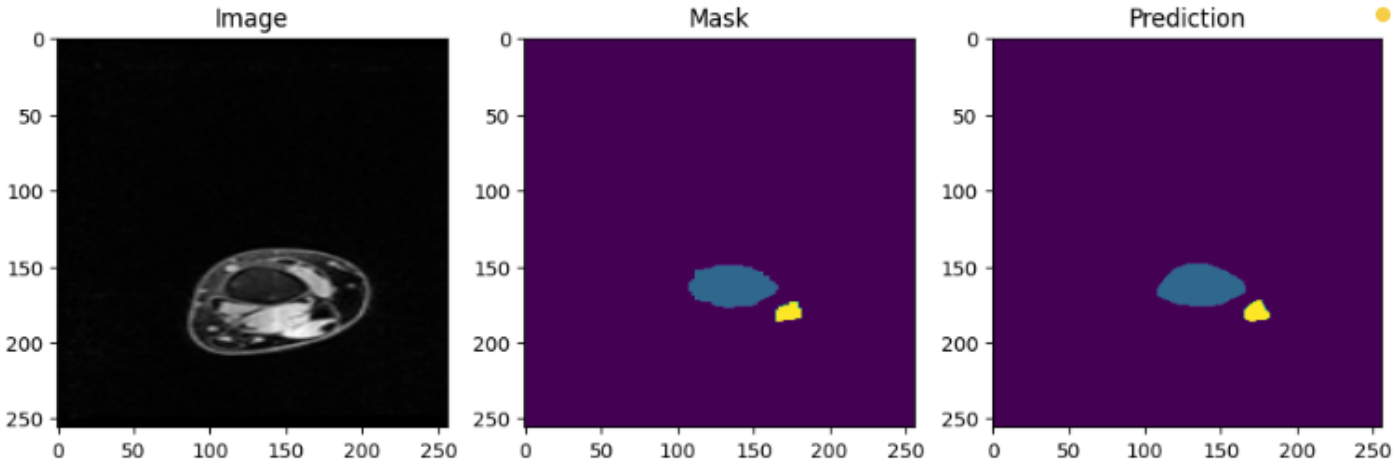

Figure 12: Friedlander Data Inference Process on Tibia/Fibula using Resnet34 U-Net

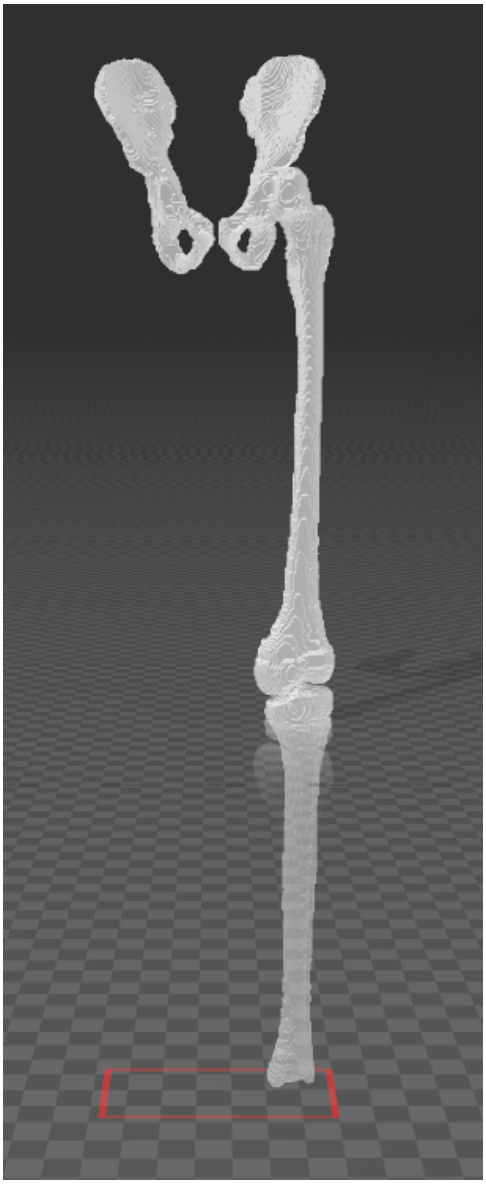

Figure 13: 3D Visualisation of Segmentation Output using Resnet34 U-Net