Methods for 3D Femur Segmentation with Ospedale Rizzoli
Pattern Recognition course project

Francesco Pivi, Matteo Fusconi
Methods for 3D and 2D Femur Segmentation
{ francesco.pivi, matteo.fusconi4 }@studio.unibo.it
Master degree in Artificial intelligence

Abstract

2D and 3D Binary Segmentation in the medical field is crucial as it provides doctors with valuable information for patient diagnosis and treatment. Femur segmentation, in particular, allows for precise isolation of bones from other tissues, aiding in the assessment of fracture risk in older individuals.

This study aims to enhance 3D femur segmentation from CT images. Starting from TotalSegmentator as baseline [37], which is a 3D full-resolution nnUNet [17], and a small dataset of 40 CT femur images provided by Ospedale Rizzoli, our contribution is to try different Deep Learning methods to surpass its performance.

Our research has focused on both 3D and 2D methods for medical image segmentation. Specifically, we fine-tuned Total Segmentator for 3D methods, and trained UNet [30] and UNet++ [40] with ResNet50 [11], ResNet34 [11], and DeepLabv3 [6] backbones for 2D methods. We evaluated these models using metrics such as Dice score, mean Intersection over Union (mIOU), and Volume Similarity.

Our findings indicate that fine-tuning a large 3D UNet on a small dataset is suboptimal as evidenced by the lower scores obtained in our experiments. Additionally, UNet++ consistently outperformed UNet, and the DeepLabV3 backbone demonstrated better performances in segmenting entire femurs, achieving a Dice score of 0.952. Meanwhile, the ResNet50 backbone proved to be more effective for segmenting femur heads, with a Dice score of 0.970.

The source code is available on github[24].

1 Introduction

Segmentation is a crucial process in the field of computer vision, defined as the division of an image into non-overlapping regions that together represent the entire image [38]. This set of techniques allows for the extraction and conveyance of information from images through morphological properties, assigning each pixel a unique label associated with a specific class. The main segmentation methods [28] include pixel-based approaches, which focus on the luminance of individual pixels; edge-based approaches, which utilize discontinuity criteria; region-based approaches, which use similarity criteria among neighboring regions; and model-based approaches, which rely on geometric models of the object being searched for. Additionally, supervised methods, which require neural networks and transfer learning [41], automate the recognition of relevant features [31].

Modern segmentation techniques enable faster and more accurate image analysis, removing background information and allowing deep learning algorithms to process a significantly larger number of images than humans. This leads to quicker and more effective patient care [9].
Our goal is to explore over this dataset the efficiency and accuracy of femur segmentation using the currently available state-of-the-art deep learning models, thus making a significant contribution to clinical practice and medical research.

In the medical field, 2D and 3D semantic segmentation is of paramount importance as it provides physicians with crucial information for patient diagnosis and treatment [32]. Specifically, femur segmentation allows for the precise isolation of bones from surrounding tissues, facilitating the assessment of the risk of spontaneous fractures in the elderly [4]. This study aims to enhance the 3D segmentation of femoral fractures from CT images by comparing various deep learning methods such as TotalSegmentator [37], UNet [30], and UNet++[40] with backbones like ResNet50, ResNet34[11], and DeepLabv3[6]. The results show that using 2D finetuned models over our dataset can be beneficial in terms of Dice score, mean Intersection over Union (mIOU) and hard-

ware costs.

2 Background

2.1 2D Image Segmentation [25]

Given images of the size $H \times W$, the 2D image segmentation task consist in determining a class label for each pixel:

$$p(u, v) \rightarrow c_{uv} \in \mathbb{C}$$

where $p(u, v)$ is the function learned by our model (u and v are the pixel coordinates), $\mathbb{C} = \{1, .., C\}$ is a set of predefined categories, with each category corresponding to a non overlapped specific anatomical region of interest. In medical imaging, this often involves segmenting organs, tumors, or other pathological regions from the surrounding tissues. Convolutional Neural Networks (CNNs) have become the standard approach for 2D image segmentation due to their ability to learn hierarchical features directly from the image data. Architectures such as U-Net have been particularly successful, employing an encoder-decoder structure that captures context and high-level features while preserving spatial resolution for precise boundary delineation [30].

2.2 3D Image Segmentation [12]

3D image segmentation extends the principles of 2D segmentation to three-dimensional volumetric data, giving a label to each voxel (the 3D equivalent of a pixel) in the volume:

$$I(x, y, z) \rightarrow c_{uv} \in \mathbb{C}$$

This is particularly relevant in medical imaging modalities like CT and MRI, which produce volumetric scans composed of multiple slices.

The primary advantage of 3D segmentation over 2D segmentation is its ability to leverage spatial information across multiple slices, leading to more coherent and accurate segmentations [33]. This is crucial for structures that span several slices and exhibit complex 3D shapes. However, 3D segmentation also presents significant computational challenges due to the increased data volume and the need for more complex network architectures capable of handling three-dimensional convolutions.

The nnUNet architecture [18] is particularly noteworthy in this context. It automates the configuration and training of 3D CNNs for segmentation tasks, adapting to the specific characteristics of the dataset without manual tuning. nnUNet's pipeline includes pre-processing steps such as normalization and data augmentation, optimal network architecture selection, and post-processing to refine the segmentation results. This comprehensive approach ensures high performance and robustness across diverse 3D medical imaging datasets.

In our project, we applied both 2D and 3D segmentation approaches to analyze medical images.

2.3 Unet [30]

U-Net is a prominent deep learning architecture, initially introduced in the paper "U-Net: Convolutional Networks for Biomedical Image Segmentation" [30]. This architecture was specifically designed to address the challenge of limited annotated data in the medical field, aiming to achieve high accuracy and speed even with smaller datasets.
It has an unique architecture consisting of a contracting path and an expansive path, connected through skip connections that enable the network to effectively leverage available data.

U-Net builds upon the principles of Fully Convolutional Networks (FCNs) [20], which were developed to overcome the limitations of classic convolutional neural networks (CNNs) [27] in segmentation tasks. FCNs are designed to make pixel-by-pixel predictions, assigning a label or class to each pixel in the input image. By using transposed convolutions (also known as deconvolutions [26] or upsampling layers) to replace fully connected layers, FCNs upsample the feature maps to match the spatial resolution of the input image. Skip connections in FCNs help preserve fine-grained details and contextual information, enhancing the accuracy of segmented regions.

In U-Net, the contracting path, or encoder, is responsible for extracting relevant features from the input image. It consists of a series of convolutional layers that reduce the spatial resolution while increasing the depth of the feature maps, capturing increasingly abstract representations of the input. Each step in the contracting path involves two convolutional layers with ReLU activation functions, followed by a max-pooling operation that downsamples the feature maps, reducing their spatial dimensions.

The expansive path, or decoder, works on decoding the encoded data and reconstructing the spatial resolution of the input. The decoder layers upsample the feature maps and perform convolu-

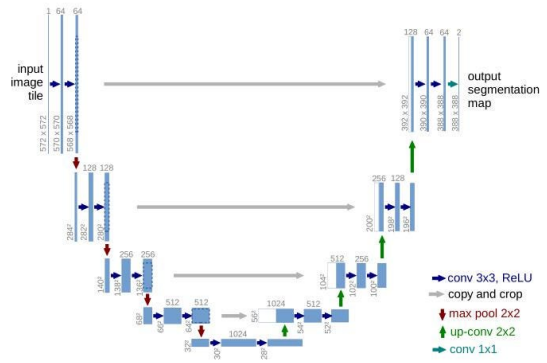


Figure 1: Unet architecture [30] We can notice on the left size the encoder block while on the right the decoder one. In the center are visible the skip connections, useful to combine contextual and pixel information.

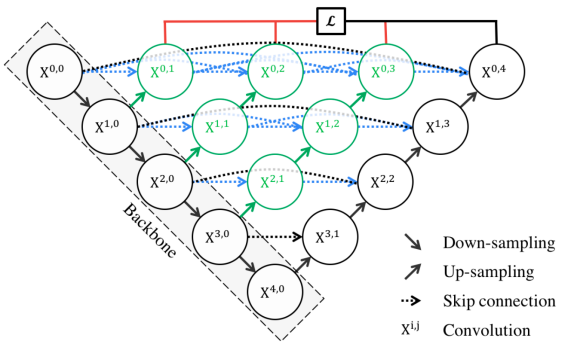


Figure 2: Unet++ architecture [40] As in the Unet we observe the downsampling and upsampling blocks on the left and right respectively. In the center now we have different levels of upsampling and dense connections.

tional operations to refine the features. Crucially, the expansive path includes skip connections from the contracting path, which help preserve spatial information lost during downsampling. These skip connections enable the decoder to accurately locate features, resulting in precise segmentation maps.

The skipping connections also enable the combination of activations of high-level and low-level features, enabling it to capture fine-grained details while maintaining contextual information. This makes U-Net particularly effective for image segmentation tasks, where precise delineation of object boundaries is critical. The architecture's ability to integrate multi-scale feature maps from various layers enhances its capacity to absorb contextual information and capture details at different levels of abstraction, resulting in accurate and detailed segmentation results.

Unet [40]

UNet++ [40], an evolution of the original UNet architecture, introduces several enhancements aimed at improving medical image segmentation performance. It retains the core encoder-decoder structure of UNet but integrates deeply-supervised learning and nested, dense skip connections between encoder and decoder pathways. These innovations effectively reduce the semantic gap between feature maps, enabling more efficient learning processes for optimizers. UNet++ has been extensively evaluated across diverse medical segmentation tasks, demonstrating superior performance compared to both the traditional UNet and wide U-Net models. Specifically, in tasks such as nodule segmentation in low-dose CT scans, nuclei segmentation in microscopy images, liver segmentation in abdominal CT scans, and polyp segmentation in colonoscopy videos, UNet++ achieves significant improvements with an average IoU gain of 3.9 and 3.4 points over U-Net and wide U-Net, respectively. This architecture's ability to preserve detailed information and contextual understanding across multiple scales makes it particularly effective in complex medical imaging scenarios where precise segmentation is critical.

2.4 ResNets [11]

Residual Networks [11], or ResNets, are a type of neural network architecture that introduced the concept of residual learning. Traditional deep neural networks suffer from degradation problems as they get deeper, meaning they struggle to maintain or improve performance with increased depth.

The core idea of ResNet is the use of residual blocks. A residual block is defined as:

$$\mathbf{y} = \mathcal{F}(\mathbf{x}, \{W_i\}) + \mathbf{x}$$

where $\mathbf{x}$ is the input to the block, $\mathbf{y}$ is the output, $\mathcal{F}$ is a residual function to be learned, and $\{W_i\}$ denotes the weights of the block. The key insight is that instead of trying to directly learn a mapping from input to output, ResNet learns residual mappings. This means the network learns the difference between the desired output and the input, rather than the output itself.

This formulation helps in training deeper networks effectively by mitigating the vanishing gradient problem and allowing easier flow of gradients during back-propagation.

In practice, ResNet architectures typically stack multiple residual blocks together. Each block can

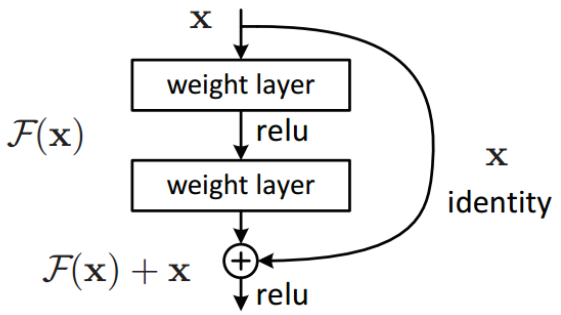


Figure 3: A single ResnetBlock [11] It's clear that then network giving a small weight to the processing path can still learn the identity function.

have multiple layers, and the overall network can have hundreds or even thousands of layers while remaining trainable and effective.

ResNet-50 and ResNet-34 are both variants of the ResNet architecture, but they differ primarily in their depth and the number of layers.

ResNet-34:

- Consists of 34 layers (excluding the final fully connected layer).
- Contains basic residual blocks (also known as ResNet-B blocks) with two convolutional layers and a skip connection.
- Generally lighter and faster to train compared to deeper variants like ResNet-50.
- Suitable for tasks where a moderate depth is sufficient, such as image classification on smaller datasets or scenarios where computational resources are limited.

ResNet-50:

- Consists of 50 layers (excluding the final fully connected layer).
- Utilizes deeper residual blocks (ResNet-B blocks) with three convolutional layers in each block.
- Can capture more complex features due to its increased depth.
- Tends to perform better on larger and more complex datasets or tasks requiring more intricate feature extraction.
- Requires more computational resources and training time compared to ResNet-34.

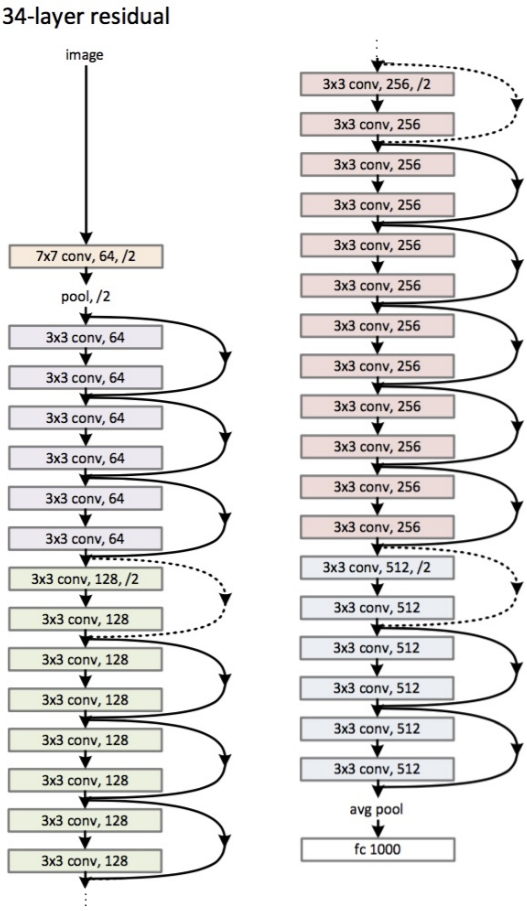


Figure 4: Resnet34 [34]. We can notice the huge depth of the network respect to the Unet architecture.

In summary, the main difference between ResNet-50 and ResNet-34 lies in their depth and complexity, which impacts their performance and suitability for different types of tasks and datasets.

2.5 DeepLabv3 [6]

DeepLabv3 [6] is a state-of-the-art deep learning model designed for semantic image segmentation, where each pixel in an image is assigned a semantic label. It utilizes a deep convolutional neural network (CNN) architecture, typically based on networks like ResNet or MobileNet [13], to extract hierarchical features that capture both local details and global context necessary for accurate object segmentation.

A key innovation in DeepLabv3 is its use of atrous (or dilated) convolutions, which expand the network's receptive field without increasing parameters.

This enables the model to effectively capture fine details and handle objects at multiple scales

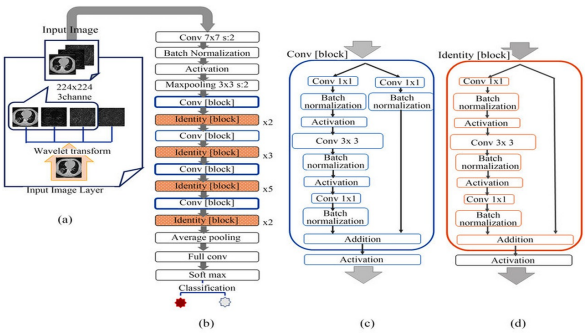


Figure 5: Resnet50 [23]. We can notice the huge depth of the network respect to the all the other infrastructures described.

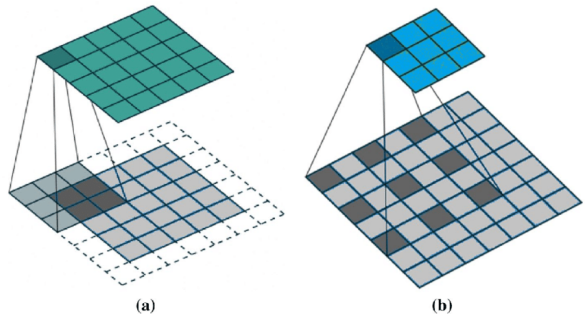


Figure 6: Comparison between classic (a) and atrous (b) convolutiuons. We can observe the enlargement of the receptive field in the atrous ones [5].

within an image.

Additionally, DeepLabv3 incorporates atrous spatial pyramid pooling (ASPP), integrating multiple parallel atrous convolutional layers with different rates to capture multi-scale context and enhance segmentation accuracy across objects of varying sizes. DeepLabv3 has demonstrated superior per-

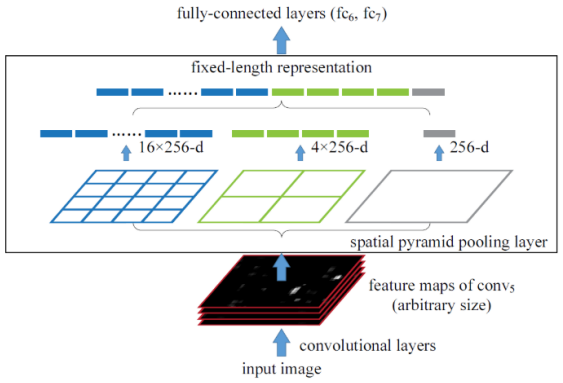


Figure 7: Example of implementation of spatial pyramid pooling [29]

formance on benchmark datasets such as PASCAL VOC[8] and COCO [19], consistently achieving state-of-the-art results in semantic segmentation tasks. Its ability to combine deep learning innovations with efficient feature extraction and multi-scale context modeling makes it a popular choice for applications in diverse fields including medical imaging, autonomous vehicles, and more. In sum-

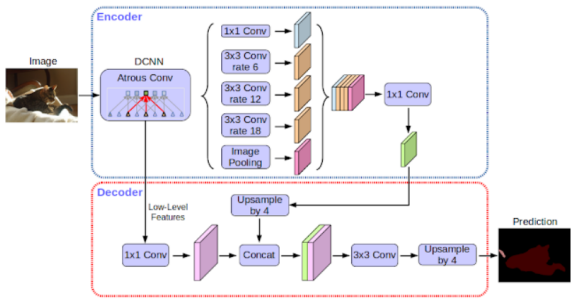


Figure 8: Deeplabv3 network. We can notice the encoder with the atrous convolution and the ASPP block, and the decoder with the dilated convolution.[2]

mary, DeepLabv3 represents a significant advancement in semantic image segmentation, leveraging advanced CNN architectures and innovative convolutional techniques to achieve precise pixel-level labeling of objects in images.

2.6 nnUNet v2 - Total Segmentator [37]

The nnUNet, short for "no-new-Net" [18], represents an innovative approach in the field of automated medical image segmentation. This framework does not introduce a new network but rather provides an automated methodology for configuring, training, and evaluating convolutional neural networks dedicated to segmentation. The nnUNet stands out for its ability to automatically adapt to the specific problem without requiring significant manual intervention. It includes a complete pipeline that covers data pre-processing, model parameter optimization, and result post-processing.

The framework automatically determines the optimal network configuration using a set of heuristic rules that optimize parameters such as batch size, learning rate and network structure. This automation enables highly competitive results with minimal manual effort, making nnUNet an ideal choice for complex medical image segmentation tasks.

The chosen network configuration is 3D full resolution, and it is a UNet, with the architecture in Fig.9

In our study, we utilized both a pre-trained model called TotalSegmentator and fine-tuned the model to further improve performance. TotalSegmentator

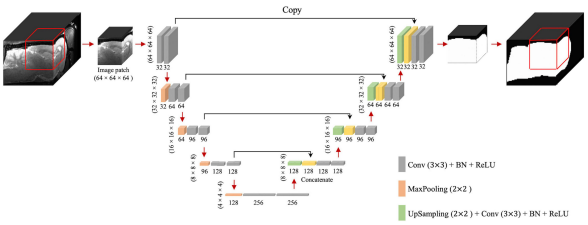


Figure 9: nnUNet 3D architecture [14].

is a pre-trained model of 3D segmentation, which was pretrained on a very large dataset consisting of 117 classes, and it can provide results at resolution 3mm and 1.5mm of voxel size. Using this model as a starting point, we performed fine-tuning with our specific data to optimize the model's performance on our images and class, namely the femur.

3 Materials and Methods

Our work is based on real data, kindly given us by Ospedale Rizzoli. The data consisted of 40 CT images of femurs along with the corresponding gold standard for segmentation, which has been manually annotated in precedence. Each 3D image of a femur was stored in the NRRD format (Nearly Raw Raster Data), which is a library and file format for the representation and processing of n-dimensional raster data. The reading of these files was facilitated by the Python library SimpleITK [3][39][21], which provides information about both voxel values and physical spacing.

A 3D image has three planes of view:

- **Sagittal**: This plane divides the body into left and right parts.
- **Axial**: This plane divides the body into upper (superior) and lower (inferior) parts.
- **Coronal**: This plane divides the body into anterior (front) and posterior (back) parts.

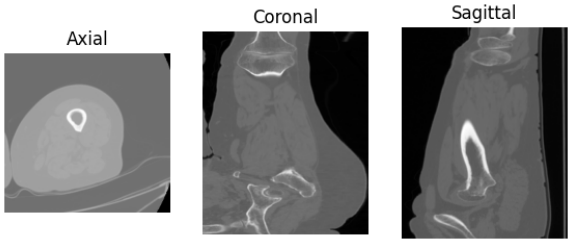


Figure 10: Example of axial, coronal and sagittal planes relative to femur CT image.

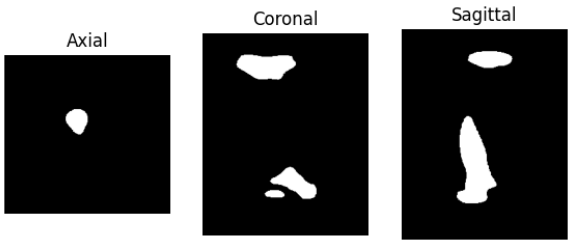


Figure 11: Example of axial, coronal and sagittal planes relative to femur Gold Standard for segmentation of the image above.

We divided the dataset into 24 images for the training set, 8 for the validation set, and the remaining 8 for the test set.

2D segmentation

To achieve improved 2D binary segmentation, the approach involved integrating various networks discussed in the background section. The goal was to leverage the strengths of each network while mitigating their weaknesses. Specifically, U-Net and U-Net++ were used as backbones, with encoder and decoder blocks composed of ResNet34, ResNet50, and DeepLabv3, respectively.

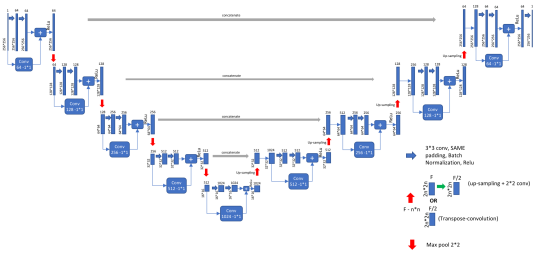


Figure 12: Composition of backbone Unet with Resnet encoder decoder blocks. [35]

For the 2D segmentation approach, we converted the 3D CT images into 2D slices using SimpleITK, and stored them in a PyTorch [1] dataloader with a batch size of 16, the maximum our model could handle. We sliced the 3D images in sagittal, axial and coroanl planes. The images were resampled to a resolution of 256x256. It was not possible for us to rotate 2D images due to problems in out of Ram memory of Google Colab free version [10] that is where we trained our models. Regarding instead the segmentation of the heads, observing the data we understood that we wanted to observe, for each femour, the first and the last 2D non-fully

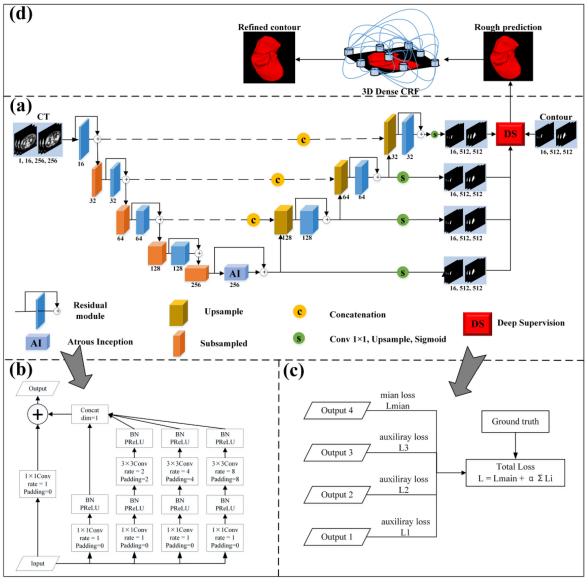


Figure 13: Composition of backbone Unet with Deeplabv3 encoder decoder blocks. The image represent the usage of that architecture for liver segmentation. [22]

black 20 images from an axial persective. We utilized pre-trained neural networks from the PyTorch segmentation models [15] library, initialized with weights from ImageNet [7] segmentation tasks.

We used CrossEntropy as the loss function:

$$\mathcal{L}(x, \theta) = -\frac{1}{N} \sum_{i=1}^{N} \log \left(\frac{e^{x_{i,y_i}}}{\sum_j e^{x_{i,j}}} \right)$$

where

- $\mathcal{L}(x, \theta)$: The cross-entropy loss.
- N: The number of samples in the batch.
- x: The input tensor (logits) of size (N, C), where C is the number of classes.
- $x_{i,j}$: The logit of the i-th sample for the j-th class.
- y_i: The ground truth class index for the i-th sample.
- θ: The parameters of the model.

Then we tested both AdamW and Adam optimizers with a learning rate of 3e-4, finding that Adam provided the best results.

3D segmentation

Regarding 3D segmentation, we followed the nnUNetv2 pipeline [16], which is quite complex and standardized, and it can be summarized by the following steps:

- **Preprocessing**:
  - *Cropping* to the region of nonzero values in order to reduce the size;
  - *Resampling* CT images to the median voxel spacing of their respective dataset, and third order spline interpolation is used for image data and nearest neighbor interpolation for the corresponding segmentation mask. This is made make CNNs better understand the extent of spacing;
  - *Clipping* CT images to the [0.5, 99.5] percentiles of the intensity values, and subsequent *z-score Normalization*, since the original intensity scale of CT scans is absolute.
- **Fine Tuning**: in order to perform transfer learning, the full Total Segmentator architecture and weights were taken, except for the last segmentation layer, which is replaced by a new layer that predicts the new class "Femur".
- **Data Augmentation**: in general to prevent over-fitting, and especially having a very small dataset, we performed data augmentation automatically with total segmentator. As it's svirtten in the paper the architecture posses a data augmentation pipeline inside itself that performs automatically during training the following transformations: random rotations, random scaling, random elastic deformations, gamma correction augmentation and mirroring [18].
- **Losses**: nnUNet v2 uses a combination of the Binary CrossEntropy loss with a novel one based on the dice score:

$$\mathcal{L} = \mathcal{L}_{BCE} + \mathcal{L}_{dice}$$

$$\mathcal{L}_{BCE} = -\sum_{i \in I} [v_i \ln(u_i) + (1 - v_i) \ln(1 - u_i)]$$

$$\mathcal{L}_{dice} = -\frac{2 \sum_{i \in I} u_i v_i}{\sum_{i \in I} u_i + \sum_{i \in I} v_i}$$

where u is the output of the network, v is the ground truth segmentation map, and $i \in I$ are the batch images. Regarding the optimizer, we use Adam with an initial learning rate of 3e-4.

- **5-fold Cross Validation**: instead of splitting train and validation sets in a fixed way, we opt to do it in 5 different ways, namely performing 5-fold stratified cross validation. After training for 5 times with 5 different training sets, we are able to find the model that generalizes better on the validation set, preventing the risk of overfitting.

4 Results

We evaluated our results using the Intersection over Union score, the Dice coefficient scores and volume similarity.
The first is the area of the intersection between the ground truths (Gt) and predicted masks (Pred) over the area of their union

$$\text{mIoU} = \frac{1}{N} \sum_{i=1}^{N} \frac{|\text{Pred}_i \cap \text{Gt}_i|}{|\text{Pred}_i \cup \text{Gt}_i|}$$

while the second is twice the area of the intersection over the total area,

$$\text{Dice} = 2 \times \frac{|\text{Pred} \cap \text{Gt}|}{|\text{Pred}| + |\text{Gt}|}$$

namely the sum of the area of the ground truth mask and the area (GT) of the predicted one (Pred).
We conducted an evaluation of these models in both two-dimensional and three-dimensional formats: in 2D scores, we consider as "area" the area of the 2D slices, whereas in 3D scores we consider as "area" the volume of the whole 3D image. This is made because 2D scores for 2D models serve as indicators for the goodness of the model itself, since it considers specifically its inputs and outputs, while 3D scores are in general indicators for the goodness of the result itself.
In addition to mIoU and Dice, we also used the mean Volume Similarity score as a 3D score. It is defined as

$$VS = 1 - VD$$

, where VD is the ratio of the difference of volumes and their sum. All metrics are visually represented in Fig 14. The conventional volume similarity metric provides an absolute value for the volume difference. However, we chose not to include this metric, as it allows us to determine whether the model is underestimating or overestimating the size of the predicted volume, as discussed later in the report.

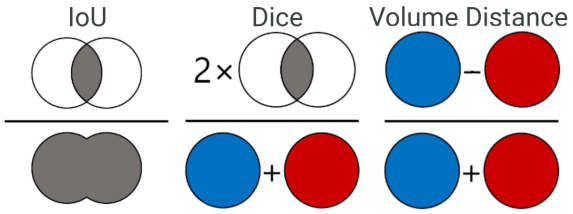


Figure 14: Graphical representations of the Intersection over union (IoU) score, Dice score and Volume Distance (VD) score. Note that the Volume Similarity (VS) score is defined as 1-VD.

The mean Intersection over Union (mIoU), ranging from 0 to 1, if has a score of 1 indicates perfect overlap between the predicted and ground truth segmentation, meaning there are no false positives or false negatives. Similarly, a Dice score of 1 also signifies perfect agreement between the two segmentations, indicating complete overlap. While both metrics measure the overlap between predicted and actual segments, the Dice score tends to be more sensitive to small objects because it gives more weight to correctly predicted positives. The mIoU, on the other hand, is stricter, as it penalizes false positives and false negatives more evenly. The volume similarity score evaluates the agreement in volume between the predicted and ground truth segments, with a score of 1 indicating identical volumes. If the volume similarity score is less than 1, the predicted mass is smaller than the ground truth, and if it is greater than 1, the predicted mass is larger.
Although we attempted fine-tuning these models, the results were inferior compared to retraining the entire model with a low learning rate. This is likely due to our limited computational resources, which prevented extensive hyperparameter tuning. We trained the models for 5 epochs, saving the model with the lowest loss on the training set. The results obtained for all the models over the test set are displayed in Tables 1, 2, respectively by 2D scores and 3D scores.

Additionally, for the 2D models, we conducted further experiments focusing on the most challenging parts of the slices, specifically the femur heads. We retrained all the models on a subset of the training set composed exclusively of axial slices of femur heads, and summarized the results in Table 3

Model	DiceScore	mIoU	TrainTime
Res50Unet	0.937 ± 0.121	0.897 ± 0.145	22 min
Res50Unet++	0.939 ± 0.131	0.904 ± 0.148	61 min
Res34Unet	0.938 ± 0.113	0.898 ± 0.136	14 min
Res34Unet++	0.940 ± 0.122	0.903 ± 0.145	28 min
DeepV3Unet	0.941 ± 0.107	0.901 ± 0.128	11 min
DeepV3Unet++	$\mathbf{0.952 \pm 0.095}$	$\mathbf{0.918 \pm 0.112}$	15 min
TotSeg	0.944 ± 0.062	0.899 ± 0.082	-
TotSegFine	0.846 ± 0.167	0.761 ± 0.197	200 min

Table 1: 2D scores of the chosen networks across the test set. The time is the time needed to train the neural net, and the error is one standard deviation of the results. The first six models takes as input 2D images while the last two 3D ones.

Model	DiceScore	mIoU	VS
Res50Unet	0.948 ± 0.030	0.905 ± 0.047	1.019 ± 0.017
Res50Unet++	0.959 ± 0.011	0.923 ± 0.018	1.008 ± 0.011
Res34Unet	0.957 ± 0.02	0.920 ± 0.034	1.023 ± 0.008
Res34Unet++	0.959 ± 0.021	0.924 ± 0.035	$\mathbf{1.002 \pm 0.016}$
DeepV3Unet	0.960 ± 0.014	0.924 ± 0.025	1.020 ± 0.011
DeepV3Unet++	$\mathbf{0.968 \pm 0.007}$	$\mathbf{0.939 \pm 0.013}$	1.014 ± 0.009
TotSeg	0.955 ± 0.009	0.914 ± 0.016	1.036 ± 0.013
TotSegFine	0.858 ± 0.028	0.753 ± 0.043	0.991 ± 0.037

Table 2: 3D scores of the chosen networks across the test set. The error is one standard deviation of the results. The first six models takes as input 2D images while the last two 3D ones.

for 2D scores, and Table 4 for 3D scores.

Model	DiceScore	mIoU	TrainTime
Res50Unet	0.958 ± 0.058	0.923 ± 0.078	2 min
Res50Unet++	0.943 ± 0.081	0.901 ± 0.110	5 min
Res34Unet	0.944 ± 0.095	0.903 ± 0.113	1 min
Res34Unet++	$\mathbf{0.960 \pm 0.055}$	$\mathbf{0.927 \pm 0.075}$	2 min
DeepV3Unet	0.937 ± 0.121	0.896 ± 0.136	1 min
DeepV3Unet++	0.945 ± 0.105	0.908 ± 0.12	1 min

Table 3: 2D results for the segmentation of the heads. All the models takes as input 2D images.

Model VS	Input	DiceScore	mIoU
Res50Unet	0.968 ± 0.008	0.939 ± 0.015	1.016 ± 0.014
Res50Unet++	0.962 ± 0.011	0.927 ± 0.019	1.011 ± 0.014
Res34Unet	0.961 ± 0.012	0.926 ± 0.022	0.989 ± 0.014
Res34Unet++	$\mathbf{0.970 \pm 0.006}$	$\mathbf{0.942 \pm 0.012}$	1.009 ± 0.015
DeepV3Unet	0.962 ± 0.008	$0.927 \pm 0.0.014$	0.987 ± 0.015
DeepV3Unet++	0.966 ± 0.007	0.935 ± 0.014	$\mathbf{0.995 \pm 0.016}$

Table 4: 3D results for the segmentation of the heads. All the models takes as input 2D images.

5 Discussion

Looking at Tables 1 and 2, relatively to DiceScore and mIoU we notice that the best-performing model is the 2D configuration DeepLabV3 with a U-Net++ backbone. The U-Net++ backbone did improve upon the results of other models as expected. This can be attributed to the dense connections that enable an enhancement of generalization of the model [40]. Also it seems that enbles the model to reduce the over/under-stimation regarding the volume similarity.

From an encoder architecture perspective, these results were anticipated. The DeepLabV3 architecture is specifically designed for semantic segmentation tasks, utilizing atrous convolutions to focus on segmentation at the pixel level. Atrous convolutions [6] enlarge the receptive fields, ensuring that each activation contains relevant information, unlike ResNets where some receptive fields may capture irrelevant background information. The combination of fine-grained semantic segmentation achieved by the U-Net++ architecture and the large receptive field of DeepLabV3 logically makes it the best choice for this type of task.

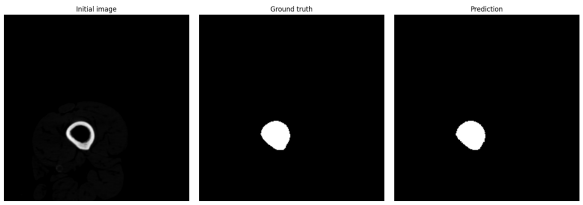


Figure 15: The highest Intersection over Union (IoU) achieved over the test set by the DeepLabV3Unet++ encoder is 0.983, corresponding to that particular image.

In Figure 15, we can see an example of accurate segmentation. The DeepLabV3 network has successfully segmented nearly every pixel, effectively ignoring the external background while correctly identifying the dark-colored bone marrow within the bone. This high level of accuracy is due to the network's deep stages, where the receptive field encompasses almost the entire image.

By examining the image segmented with the lowest quality among the various models (Figure 16 17), we can observe that a backbone using Deeplabv3Unet++ is capable of achieving results, although not satisfactory, that are three times superior to those obtained with ResNets blocks in terms of IoU.

Regarding 3D segmentation models, Total Segmentator demonstrated good performance, achieving IoU and Dice scores similar to those of 2D models. However, fine-tuning this model

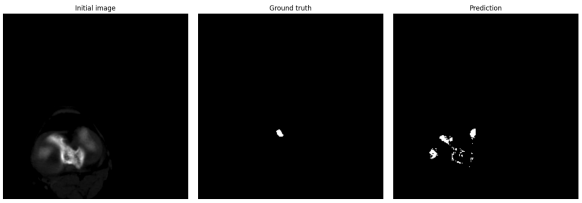


Figure 16: The worst Intersection over Union (IoU) achieved over the test set by the DeeplabUnet++ encoder is 0.1, corresponding to that particular image. Starting from the left we have: initial image, the ground truth and the prediction of our model.

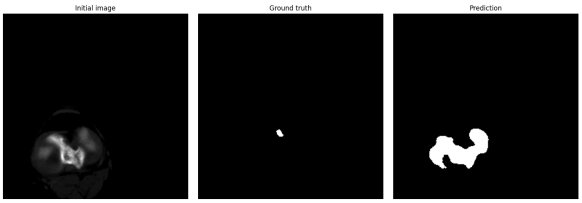


Figure 17: The worst Intersection over Union (IoU) achieved over the test set by the Resnet50-unet encoder is 0.027, corresponding to that particular image. Starting from the left we have: initial image, the ground truth and the prediction of our model.

did not yield improvements; instead, it diminished performance due to overfitting. This outcome was anticipated, given the insufficient data available to train a neural network with such a high number of parameters. The necessity of processing the entire 3D dataset, which consists of only 32 images with cross validation, coupled with the substantial parameter count in 3D convolutions, led to overfitting. Additionally, the developers of Total Segmentator and nnUNet recommend training from scratch using the same architecture with a different dataset, rather than fine-tuning from the latest checkpoint [37].
Upon examining the worst and best classified images 16 17 15, we observed that the models exhibited greater difficulty in segmenting the femoral heads, while the central part was almost always correctly predicted. This occurs because the number of slices related to the diaphysis is overrepresented compared to the extremities. Therefore, it was deemed interesting to fine-tune the 2D models specifically on the poorly classified parts in order to achieve better results on these segments as well.

Reviewing Tables 3 and 4 and the images in the above figures 18 - 27 , we observe an overall improvement in segmenting the heads. This outcome

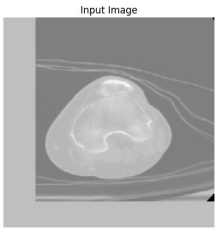


Figure 18: Femur axial slice of the head.

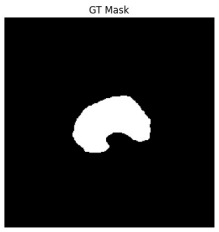


Figure 19: Ground truth image.

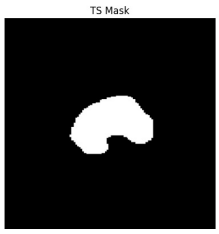


Figure 20: Total segmenator result.

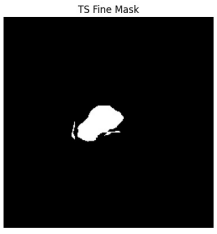


Figure 21: Total segmentator fine-tuned.

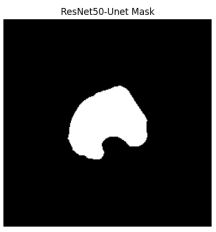


Figure 22: ResNet50-Unet result.

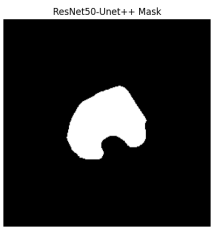


Figure 23: ResNet50-Unet++ result.

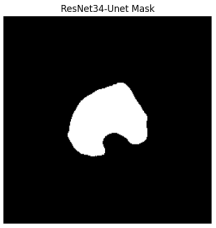


Figure 24: ResNet34-Unet result.

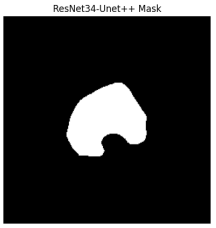


Figure 25: ResNet34-Unet++ result.

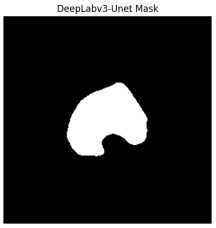


Figure 26: DeepLabv3-Unet result.

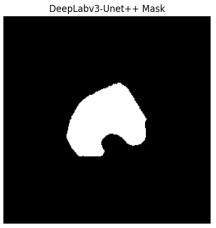


Figure 27: DeepLabv3-Unet++ result.

was anticipated since focusing on challenging data is a standard procedure. Surprisingly, ResNet34 outperformed DeepLabV3 with both backbones. The performance differences between the two in segmenting femur can be explained cosnidering the impact of network architecture and the focus of training data. DeepLabV3 leverages atrous convolution and spatial pyramid pooling to capture multi-scale context, making it highly effective for segmenting complex scenes with varied objects and detailed boundaries. This capability shines when training on full CT images, where understanding the entire femur's structure benefits from multi-scale feature capture and contextual information. However, when the segmentation task is narrowed to the femur heads, DeepLabV3's broader context capabilities become less crucial.

In contrast, ResNet34 encoder, known for its robust feature extraction through deep residual learning, excels when the task is specific and localized. Training ResNet34 on femur heads allows it to focus intensively on this region, effectively learning its nuances and specific features. Without the need to manage full-image complexity, ResNet34 efficiently concentrates on the precise characteristics relevant to femur head segmentation. This specialization likely accounts for its superior performance in this targeted task, demonstrating that a focused approach with a powerful feature extraction model can outperform more complex multi-scale architectures when dealing with localized segmentation challenges.

In conclusion, it appears that segmenting 2D images rather than using a 3D network for binary segmentation with a small dataset is the preferable strategy. It is important to highlight that extensive hyperparameter tuning could not be conducted due to limited resources. Furthermore, training on the entire femur results in quantitatively unacceptable segmentation outcomes for the heads.

6 Conclusion

Our findings demonstrate that the segmentation problem can be effectively addressed using both 3D UNet and 2D UNet models. The primary challenge we encountered was the limited size of the dataset. Due to this constraint, training 3D networks proved ineffective. To mitigate this, we employed a 2D approach by splitting the medical images into slices along the axial, sagittal, and coronal planes, and fine-tuning 2D UNet and UNet++ with pre-trained backbones. We found that DeepLabV3-UNet++ outperforms not only similar architectures, but also the pretrained TotalSegmentator. However, it struggled in predicting a correct mask for the femur heads. To overcome this problem, we managed to further improve the results, by training the 2D models from scratch with axial slices of the femur heads. Here, likely because of the more specific and localized task, the ResNet34 backbone proved to be more effective than DeepLabV3, while the UNet++ still produced better results than UNet.

In addition, we believe that with a sufficiently large dataset and computational resources, it would be feasible to train a 3D nnUNet capable of surpassing the performance of Total Segmentator and ResNet50. In order to properly train a nn-Unetthe authors of total segmentator used 1164 3D images leading to a size of 23 GB of size [36]. However, given the available data, we achieved better performance using 2D UNet on slices, with the best results obtained from the combination of DeepLabV3 and UNet++. Additionally, extensive hyperparameter tuning, which is currently beyond our resource capacity, would be necessary to maximize the performance of the 2D models.

References

[1] Jason Ansel et al. "PyTorch 2: Faster Machine Learning Through Dynamic Python Bytecode Transformation and Graph Compilation". In: *Proceedings of the 29th ACM International Conference on Architectural Support for Programming Languages and Operating Systems, Volume 2 (ASPLOS '24)*. ACM, Apr. 2024. DOI: 10.1145/3620665.3640366. URL: https://pytorch.org/assets/pytorch2-2.pdf.

[2] Eduardo Assunção et al. "Real-time weed control application using a jetson nano edge device and a spray mechanism". In: *Remote Sensing* 14.17 (2022), p. 4217.

[3] R. Beare, B. C. Lowekamp, and Z. Yaniv. "Image Segmentation, Registration and Characterization in R with SimpleITK". In: *Journal of Statistical Software* 86.8 (2018). DOI: 10.18637/jss.v086.i08. URL: https://doi.org/10.18637/jss.v086.i08.

[4] Pall Asgeir Bjornsson et al. "Automated femur segmentation from computed tomogra-

phy images using a deep neural network". In: *Medical Imaging 2021: Biomedical Applications in Molecular, Structural, and Functional Imaging*. Vol. 11600. SPIE. 2021, pp. 324–330.

[5] Changchun Cai et al. "Short-term electrical load forecasting based on VMD and GRU-TCN hybrid network". In: *Applied Sciences* 12.13 (2022), p. 6647.

[6] Liang-Chieh Chen et al. *Rethinking Atrous Convolution for Semantic Image Segmentation*. 2017. arXiv: 1706.05587 [cs.CV]. URL: https://arxiv.org/abs/1706.05587.

[7] Jia Deng et al. "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee. 2009, pp. 248–255.

[8] Mark Everingham et al. "The pascal visual object classes (voc) challenge". In: *International journal of computer vision* 88 (2010), pp. 303–338.

[9] Swarnendu Ghosh et al. "Understanding deep learning techniques for image segmentation". In: *ACM computing surveys (CSUR)* 52.4 (2019), pp. 1–35.

[10] Google. *Google Colaboratory*. https://colab.research.google.com/. Retrieved April 18, 2024. 2024.

[11] Kaiming He et al. *Deep Residual Learning for Image Recognition*. 2015. arXiv: 1512.03385 [cs.CV]. URL: https://arxiv.org/abs/1512.03385.

[12] Yong He et al. "Deep learning based 3D segmentation: A survey". In: *arXiv preprint arXiv:2103.05423* (2021).

[13] Andrew G Howard et al. "Mobilenets: Efficient convolutional neural networks for mobile vision applications". In: *arXiv preprint arXiv:1704.04861* (2017).

[14] Li-Ming Hsu et al. "3D U-net improves automatic brain extraction for isotropic rat brain magnetic resonance imaging data". In: *Frontiers in Neuroscience* 15 (2021), p. 801008.

[15] Pavel Iakubovskii. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2019.

[16] Fabian Isensee et al. "Extending nnu-net is all you need". In: *BVM Workshop*. Springer. 2023, pp. 12–17.

[17] Fabian Isensee et al. "nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation". In: *Nature methods* 18.2 (2021), pp. 203–211.

[18] Fabian Isensee et al. *nnU-Net: Self-adapting Framework for U-Net-Based Medical Image Segmentation*. 2018. arXiv: 1809.10486 [cs.CV]. URL: https://arxiv.org/abs/1809.10486.

[19] Tsung-Yi Lin et al. "Microsoft coco: Common objects in context". In: *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*. Springer. 2014, pp. 740–755.

[20] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[21] B. C. Lowekamp et al. "The Design of SimpleITK". In: *Frontiers in Neuroinformatics* 7 (2013), p. 45. DOI: 10.3389/fninf.2013.00045. URL: https://doi.org/10.3389/fninf.2013.00045.

[22] Peiqing Lv et al. "Deep supervision and atrous inception-based U-Net combining CRF for automatic liver segmentation from CT". In: *Scientific Reports* 12.1 (2022), p. 16995.

[23] Eri Matsuyama. "A deep learning interpretable model for novel coronavirus disease (COVID-19) screening with chest CT images". In: *Journal of Biomedical Science and Engineering* 13.7 (2020), pp. 140–152.

[24] Francesco Pivi Matteo Fusconi. *Link to the source code*. URL: https://github.com/MatteoFusconi/methods-for-3d-femur-segmentation.git.

[25] Shervin Minaee et al. "Image segmentation using deep learning: A survey". In: *IEEE transactions on pattern analysis and machine intelligence* 44.7 (2021), pp. 3523–3542.

[26] Hyeonwoo Noh, Seunghoon Hong, and Bohyung Han. "Learning deconvolution net-

work for semantic segmentation". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1520–1528.

[27] Keiron O'shea and Ryan Nash. "An introduction to convolutional neural networks". In: *arXiv preprint arXiv:1511.08458* (2015).

[28] Nikhil R Pal and Sankar K Pal. "A review on image segmentation techniques". In: *Pattern Recognition* 26.9 (1993), pp. 1277–1294. ISSN: 0031-3203. DOI: https://doi.org/10.1016/0031-3203(93)90135-J. URL: https://www.sciencedirect.com/science/article/pii/003132039390135J.

[29] Wahyu Pebrianto et al. "YOLOv3 with Spatial Pyramid Pooling for Object Detection with Unmanned Aerial Vehicles". In: *arXiv preprint arXiv:2305.12344* (2023).

[30] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. arXiv: 1505.04597 [cs.CV]. URL: https://arxiv.org/abs/1505.04597.

[31] Iqbal H Sarker. "Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions". In: *SN computer science* 2.6 (2021), p. 420.

[32] Pierandrea Sartori. "Comparison of 2D and 3D convolutional neural networks for medical imaging semantic segmentation". In: (2020).

[33] Abhishek Shivdeo et al. "Comparative evaluation of 3D and 2D Deep learning techniques for semantic segmentation in CT scans". In: *arXiv preprint arXiv:2101.07612* (2021).

[34] M Siddarth. "Building Resnet-34 model using Pytorch – A Guide for Beginners". In: *Analytics Vidhya* (2021). Accessed: 2024-07-27. URL: https://www.analyticsvidhya.com/blog/2021/09/building-resnet-34-model-using-pytorch-a-guide-for-beginners/.

[35] Jai Vardhan and Taraka Satya Krishna Teja Malisetti. "Breast cancer segmentation using attention-based convolutional network and explainable ai". In: *arXiv preprint arXiv:2305.14389* (2023).

[36] Jakob Wasserthal. *Dataset with segmentations of 117 important anatomical structures in 1228 CT images*. Oct. 2023. DOI: 10.5281/zenodo.10047292. URL: https://doi.org/10.5281/zenodo.10047292.

[37] Jakob Wasserthal et al. "TotalSegmentator: Robust Segmentation of 104 Anatomic Structures in CT Images". In: *Radiology: Artificial Intelligence* 5.5 (Sept. 2023). ISSN: 2638-6100. DOI: 10.1148/ryai.230024. URL: http://dx.doi.org/10.1148/ryai.230024.

[38] Wikipedia. *Segmentazione di immagini — Wikipedia, L'enciclopedia libera*. [Online; in data 27-luglio-2024]. 2018. URL: http://it.wikipedia.org/w/index.php?title=Segmentazione_di_immagini&oldid=100492877.

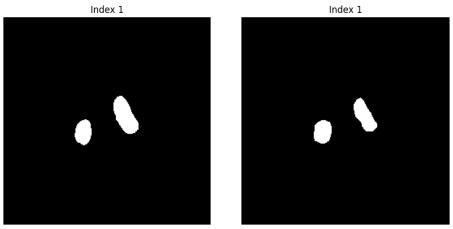
[39] Z. Yaniv et al. "SimpleITK Image-Analysis Notebooks: a Collaborative Environment for Education and Reproducible Research". In: *Journal of Digital Imaging* 31.3 (2018), pp. 290–303. DOI: 10.1007/s10278-017-0037-8. URL: https://doi.org/10.1007/s10278-017-0037-8.

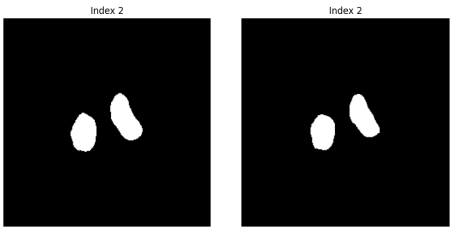
[40] Zongwei Zhou et al. *UNet++: A Nested U-Net Architecture for Medical Image Segmentation*. 2018. arXiv: 1807.10165 [cs.CV]. URL: https://arxiv.org/abs/1807.10165.

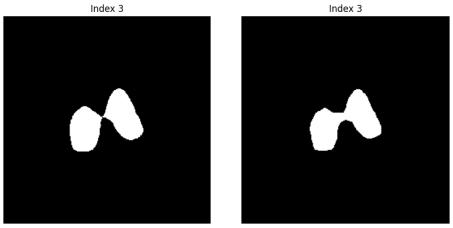
[41] Fuzhen Zhuang et al. "A comprehensive survey on transfer learning". In: *Proceedings of the IEEE* 109.1 (2020), pp. 43–76.

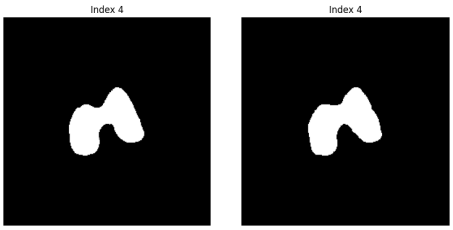
7 Appendix

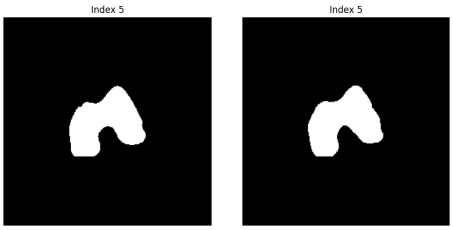
In this section we want to show as an example the resulted images of a full segmentation, done by DeepLabv3-UNet++ over the heads of the last femur in the dataset. On the right we will have the ground truth and on the left the predicted image. The sections for each femur are 40. We want to mention that the last ground truth image is the same as the one in 13.

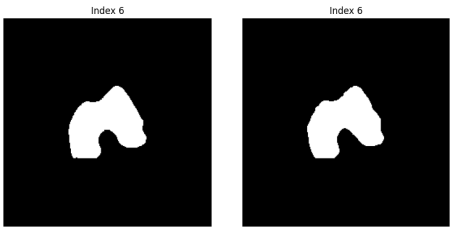


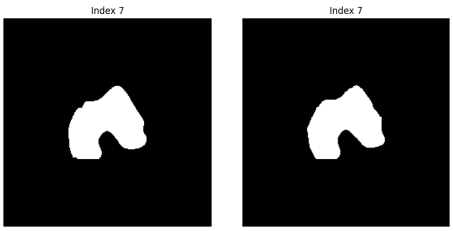


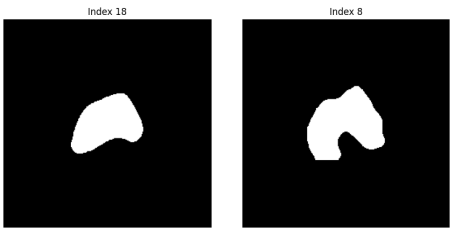


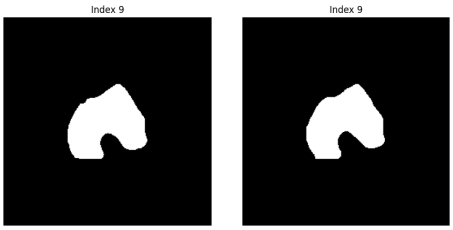


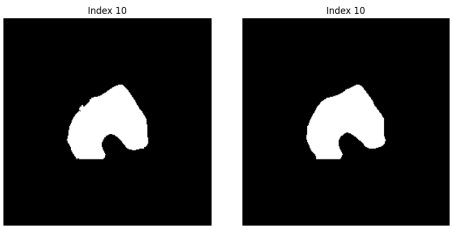


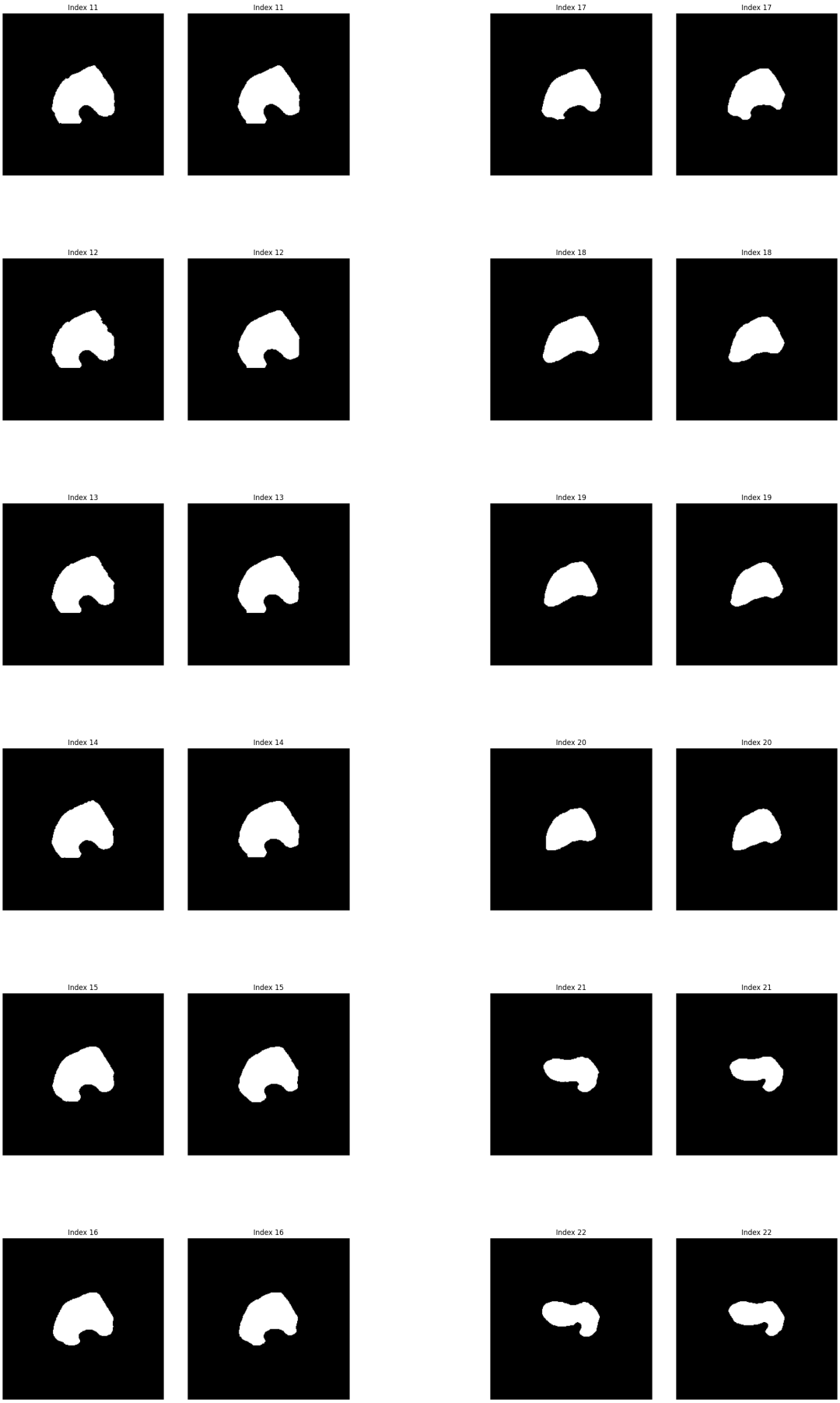


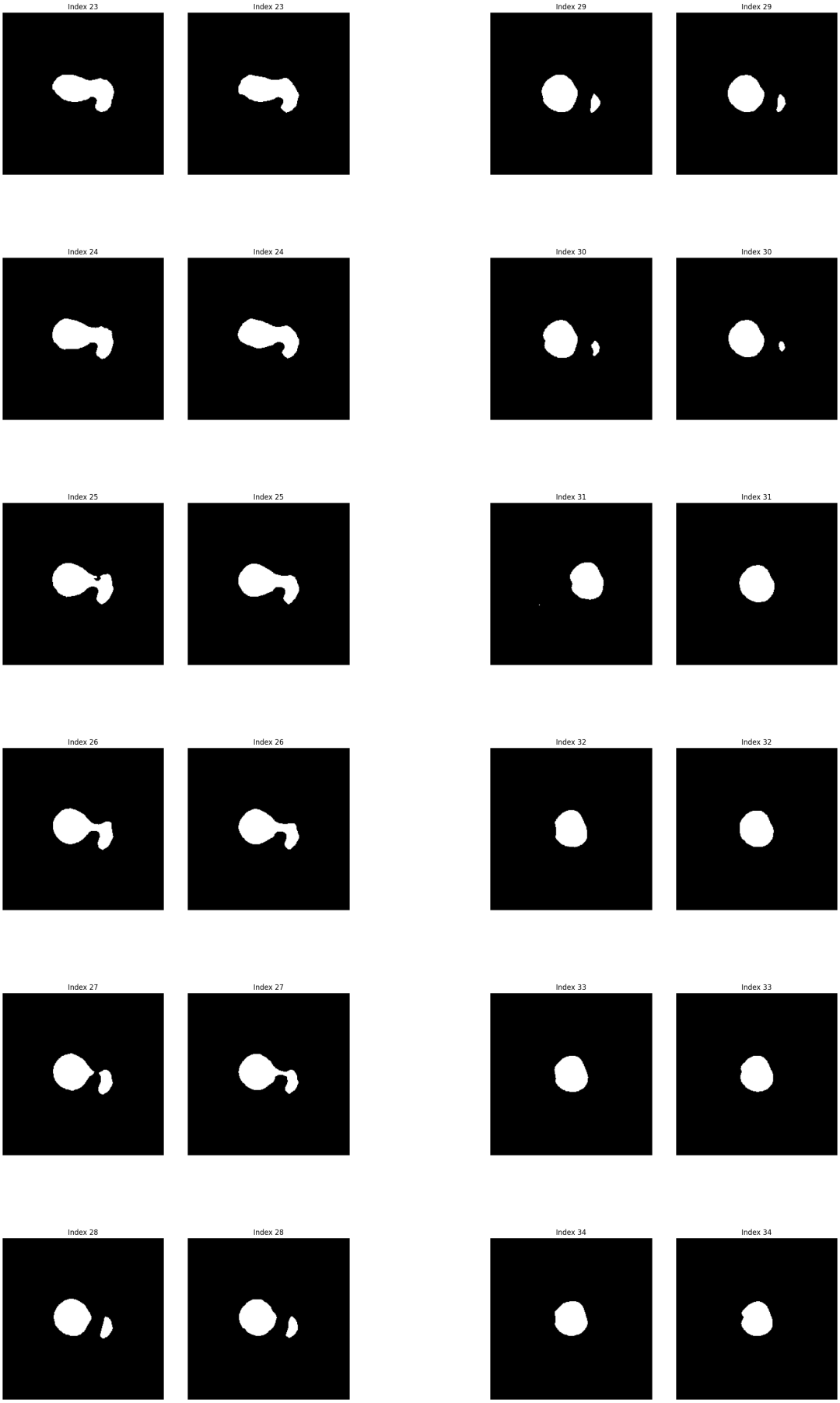


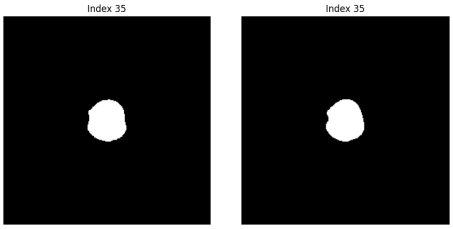


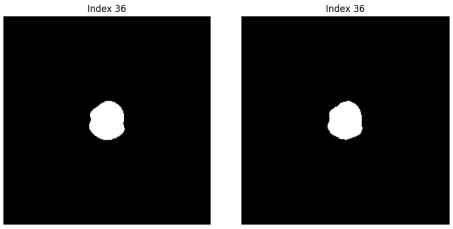


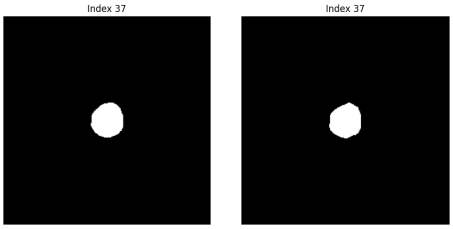


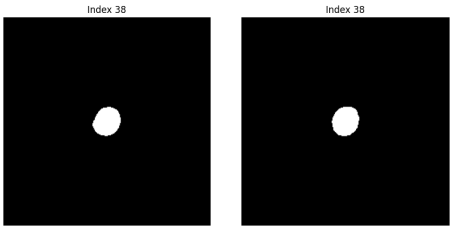

Index 11
Index 11
Index 17
Index 17
Index 12
Index 12
Index 18
Index 18
Index 13
Index 13
Index 19
Index 19
Index 14
Index 14
Index 20
Index 20
Index 15
Index 15
Index 21
Index 21
Index 16
Index 16
Index 22
Index 22

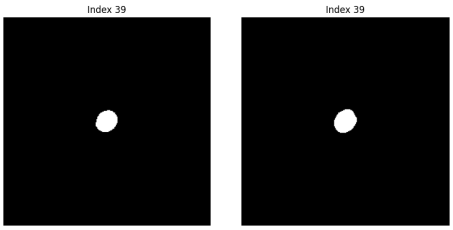

Index 23
Index 23
Index 29
Index 29
Index 24
Index 24
Index 30
Index 30
Index 25
Index 25
Index 31
Index 31
Index 26
Index 26
Index 32
Index 32
Index 27
Index 27
Index 33
Index 33
Index 28
Index 28
Index 34
Index 34

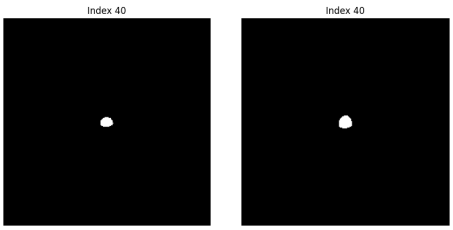

Index 35
Index 35


Index 36
Index 36


Index 37
Index 37


Index 38
Index 38


Index 39
Index 39


Index 40
Index 40