



AI/Deep Learning for Brown Fat Detection and Removal in PET/CT Scans

Abdurrahmaan Ali¹

Computer Science MEng

Dr. Ghita Kouadri Mostefaoui

Submission date: 09 05 2025

¹**Disclaimer:** This report is submitted as part requirement for the MY DEGREE at UCL. It is substantially the result of my own work except where explicitly indicated in the text. report may be freely copied and distributed provided the source is explicitly acknowledged

Abstract

Brown adipose tissue (“brown fat”) is a metabolically active fat that can appear as intense fluorodeoxyglucose uptake on PET/CT scans, potentially mimicking pathological uptake. This project aimed to develop an AI-based approach to automatically detect the presence of brown fat in PET/CT images and suppress its signal in the scans. A two-phase solution is presented: (1) a 2D convolutional neural network classifier (ResNet-50 based) that analyzes PET maximum-intensity projection (MIP) images to detect if brown fat is present, and (2) a 3D processing pipeline for PET/CT volumes that segments and removes brown fat regions using a combination of CT-based Hounsfield Unit filtering for adipose tissue, deep learning-based liver segmentation (using TotalSegmentator), and rule-based relative intensity thresholding. An F1-score of about 0.907 was achieved in classifying scans for brown fat presence. For scans with brown fat, it is demonstrated that the hybrid 3D removal method can effectively eliminate brown fat uptake regions while preserving other physiological and pathological signals. The design decisions, challenges encountered (such as data limitations and threshold selection), and lessons learned are discussed. The results show promise for improving PET/CT interpretability by reducing false positives from brown fat, and future improvements are outlined including more advanced segmentation and broader validation.

Contents

1	Introduction	1
2	Context	3
2.1	Literature Review	3
2.1.1	Brown Adipose Tissue in PET/CT imaging	3
2.1.2	Deep learning for medical imaging analysis	5
2.1.3	Summary of literature insights	8
3	Methodology and Implementation	10
3.0.1	Phase 1: 2D CNN classifier for brown fat presence	11
3.0.2	Phase 2: 3D Brown Fat segmentation and removal	13
4	Testing	17
4.0.1	Unit and Component Tests	18
5	Results and Evaluation	19
5.0.1	Phase 1 Classifier performance	19
5.0.2	Phase 2 Brown Fat removal results	21
6	Discussion and Reflections	24
6.0.1	Phase 1: Classifier development reflections	24
6.0.2	Phase 2: Segmentation and removal reflections	25
6.0.3	General observations and potential clinical impact	26
6.0.4	Limitations and potential improvements	27
7	Conclusion and Future Work	28
8	Appendices	34
.1	Appendix 1: Project plan	34
.2	Appendix 2: Interim report	38
.3	Appendix 3: User guide	41
.4	Appendix 4: Code	44

Chapter 1

Introduction

Brown adipose tissue (BAT), commonly known as brown fat, is a special type of fat in humans that generates heat through thermogenesis [1]. In positron emission tomography/computed tomography (PET/CT) imaging, especially with the radiotracer ^{18}F -FDG, active brown fat can uptake glucose and appear as bright regions on the PET scan. This is problematic in oncologic PET/CT interpretation because brown fat uptake can closely mimic malignant hypermetabolic lesions. As a result, it is a well-recognized source of false-positive findings on PET scans. Yeung *et al.* (2003) [2] first characterized patterns of FDG uptake in adipose tissue, noting that metabolically active fat depots could lead to such false positives. In clinical studies, the incidence of significant brown fat FDG uptake in adult PET/CT scans has been reported in a few percent of patients (on the order of 2-4%), tending to occur more frequently in younger, lean female patients and in cold conditions [3]. An example of the diagnostic challenge is given by Truong *et al.* (2004) [4], who reported cases where focal FDG uptake in mediastinal brown fat was initially mistaken for malignancy until careful correlation with CT revealed only fat tissue in those regions. Such cases highlight the need for methods to identify and distinguish brown fat uptake from true pathological uptake.

Clinically, measures are often taken to minimize brown fat activation before and during PET scans. Patients may be kept warm and calm, as cold exposure and stress can stimulate brown fat through sympathetic nervous activity [5]. Pharmacological interventions (e.g., a single dose of propranolol) have also been shown to reduce FDG uptake in brown fat by blocking β -adrenergic stimulation [6]. However, these methods are not always used or fully effective, and brown fat uptake remains a recurrent nuisance in PET imaging. From a diagnostic perspective, when brown fat uptake does occur, radiologists must carefully differentiate it from disease. Typically, the symmetric pattern in cervical/supraclavicular regions is a clue, and confirmation comes from noting fat-density tissue on the CT at the sites of PET uptake [5]. This manual interpretation can be time-consuming and can still lead to uncertainty in borderline cases.

There is a strong motivation to develop automated methods to handle brown fat in PET/CT scans. Such methods could improve the accuracy of PET-based diagnoses by reducing false-positive interpretations. Additionally, brown fat is of great interest in metabolic research, so automated detection and quantification of brown fat could aid studies on obesity and metabolism [7]. Recognizing this, the community has even established standardized criteria for reporting brown fat in imaging studies. The Brown Adipose Reporting Criteria in Imaging Studies

(BARCIST 1.0) [8] were published in 2016 to define how to identify and measure active BAT on FDG PET/CT in a consistent manner. These criteria include specific thresholds for PET uptake and CT attenuation to classify a region as brown fat, underscoring the importance of objective measures.

In this project, an AI and deep learning approach is employed to address the problem of brown fat detection and suppression in PET/CT scans. The main goal is twofold. First, to automatically detect whether a given PET/CT scan contains active brown fat. This is approached via a 2D image classification task on PET maximum-intensity projections (MIPs) of the whole-body scan. Second, if brown fat is present, the aim is to automatically locate and remove (or suppress) those regions of uptake in the 3D PET/CT data, producing an output PET image in which the brown fat signal is eliminated. By doing so, any spurious hotspots from brown fat would be gone, allowing radiologists (or downstream AI algorithms) to focus on true pathological uptake. This two-phase strategy reflects a design choice to simplify the problem: a fast preliminary detection followed by a targeted intervention on the image. Specifically, Phase 1 uses a deep convolutional neural network (CNN) classifier (based on ResNet-50 architecture [9]) to detect brown fat presence from PET MIP images. Phase 2 then operates on the full 3D PET/CT volume. It utilizes a Hounsfield Unit (HU) based filter to identify adipose tissue [10], leverages a pretrained segmentation model (TotalSegmentator [11]) for liver segmentation to establish a reference PET uptake, and applies rule-based thresholding on PET uptake within the fat mask to segment out brown fat regions and suppress their intensities.

This report presents the methodology, results, and analysis of the developed approach. A review of relevant literature is provided in Section 2, covering brown adipose tissue in imaging [12] [13] [14] and deep learning methods in medical imaging [15] [16] [17] [18] that inform this approach. Section 3 details the implementation of Phase 1 and Phase 2, including data, model architecture, and processing steps. In Section 4, the performance of the 2D classifier and the effectiveness of the 3D brown fat removal are evaluated, providing both quantitative metrics (such as classification F1-score and qualitative examples of PET images before/after brown fat suppression). Section 5 offers a discussion of challenges faced, design decisions made, and lessons learned during the project. Finally, Section 6 concludes the report with a summary of achievements and suggestions for future improvements, such as more advanced segmentation techniques and broader validation of the system. The overall aim is to demonstrate a feasible pipeline that leverages AI (deep learning) to tackle a practical problem in PET/CT interpretation, improving diagnostic clarity by dealing with brown fat artifacts.

Chapter 2

Context

2.1 Literature Review

2.1.1 Brown Adipose Tissue in PET/CT imaging

Brown adipose tissue is a specialized fat known for its role in non-shivering thermogenesis [1]. Unlike white adipose tissue, which primarily stores energy, brown fat burns energy to produce heat, aided by a high mitochondrial content and expression of uncoupling protein-1 [1]. Historically, significant brown fat was thought to be present mainly in infants and hibernating mammals [1] [5]. However, since the early 2000s, PET/CT scans with ^{18}F -FDG in adult humans revealed metabolically active fat depots in regions such as the neck and supraclavicular area [5]. Yeung *et al.* (2003) [2] documented patterns of FDG uptake in adipose tissue and muscle across a large sample of PET scans, bringing attention to these unexpected hotspots in fat as a source of false positives. Subsequent studies and case reports, like Truong *et al.* (2004) [4], confirmed that brown fat uptake can masquerade as pathology (for instance, lymph node metastases) on PET imaging. It became clear that readers of oncologic PET must be aware of brown fat's characteristic distribution and appearance to avoid misinterpretation. Common locations of active BAT in adults include the supraclavicular regions, neck, mediastinum, paravertebral areas, and around the kidneys and adrenal glands [5]. Typically, the uptake is bilaterally symmetric in the neck/shoulder region, which is a helpful clue since most malignancies would present asymmetrically [5]. Nevertheless, atypical distributions (e.g., focal mediastinal brown fat) can occur and be confusing [4] [5].

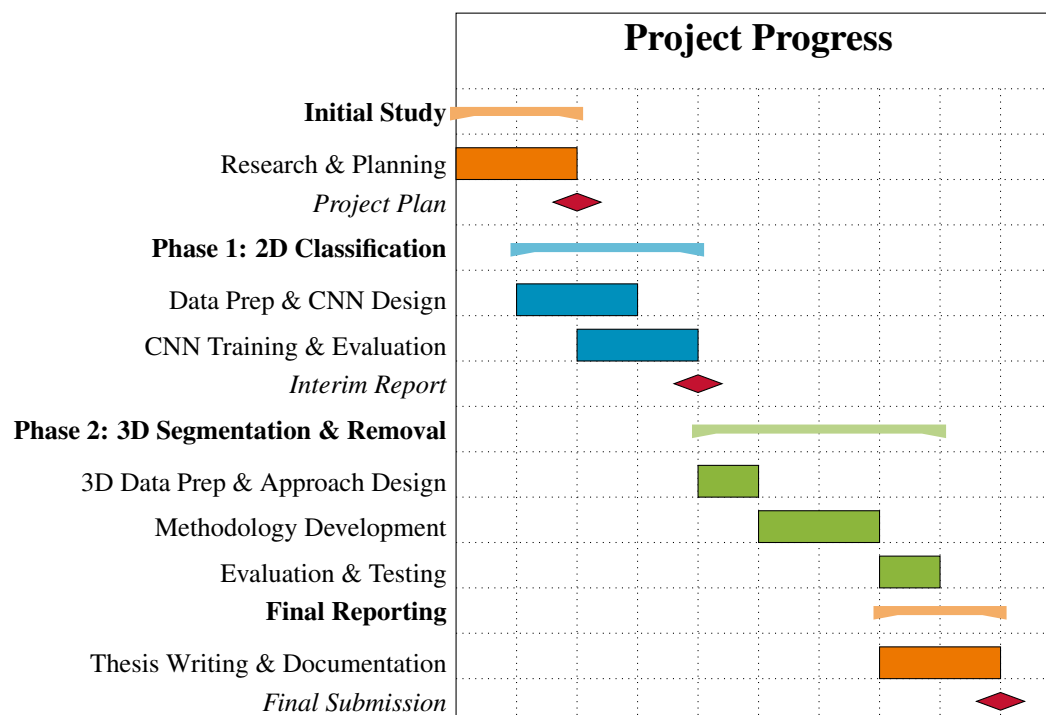
Multiple factors influence the prevalence and detectability of brown fat on PET. Younger patients and females are more likely to show brown fat uptake, as are those with lower body fat percentage [3]. Cold ambient temperature prior to or during the uptake phase of the PET scan is a strong activator of brown fat [3]. Seasonally, scans done in winter tend to have higher incidence of brown fat activity than those in warmer periods [3]. Understanding these factors, nuclear medicine departments often implement protocols to reduce brown fat activation: keeping the patient warm (e.g., with blankets) during the uptake period, avoiding exposure to cold waiting rooms, and minimizing stress or shivering [5]. Pharmacological suppression with beta-blockers has also been studied. Söderlund *et al.* (2007) [6] demonstrated that administering 20 mg of propranolol prior to FDG injection could significantly reduce brown fat FDG uptake in patients, without adverse effects. This approach directly targets the sympathetic stimulation pathway of brown fat [1] [6]. However, routine use of propranolol is not universally

adopted, and practical issues (e.g., contraindications or additional patient management) limit its widespread use.

In cases where brown fat uptake is present on PET/CT, distinguishing it from disease relies on careful image correlation. The CT component is crucial: brown fat has fat-equivalent Hounsfield units (around -100 HU [10]) and typically appears as normal fat (sometimes slightly denser due to mitochondria and perfusion, but still within fat range [5]) in the anatomical image. Malignant lesions, in contrast, usually have soft-tissue density or form discrete masses visible on CT. Therefore, if a PET hotspot has no corresponding soft-tissue lesion on CT and the area is occupied by fat, the uptake can be attributed to brown fat with high confidence [2] [4] [5]. Moreover, if a hypermetabolic focus on PET is surrounded by fat (for example, fat around a lymph node), and the node looks entirely normal by size and morphology on CT, one can deduce that the FDG uptake is likely in the fat and not the node. Experienced readers use such criteria to avoid false positives. Nevertheless, this manual mental process could benefit from automation: an algorithm that cross-references PET uptake with CT fat regions could flag likely brown fat activity automatically. This idea underpins part of this project's strategy.

Beyond being a confounding factor, brown adipose tissue has emerged as an organ of interest on its own. Research into human metabolism and obesity has shown that the presence of active BAT is associated with systemic metabolic effects [7]. As such, scientists have conducted dedicated PET/CT studies to measure brown fat volume and activity under standardized conditions (often cold-stimulated). To allow comparison across studies, the BARCIST 1.0 criteria were introduced by an international panel led by Chen *et al.* in 2016 [8]. BARCIST 1.0 provides guidelines like: a region is considered BAT on FDG PET if it has CT attenuation between about -190 to -10 HU and a tissue uptake above a certain standardized uptake value (SUV) threshold (for example, an SUV of 2.0 or higher, or a standardized uptake normalized to lean body mass (SUL) of 1.5, in at least a 4 mm region) [8]. It also recommends how to quantify BAT volume and activity (e.g., total BAT volume and total metabolic activity in SUV-volume) [8]. These criteria effectively formalize a threshold-based approach to identifying brown fat in PET/CT images. Many subsequent studies have applied BARCIST criteria to analyze brown fat in various populations. In this work, similar concepts of combining CT fat detection with PET uptake thresholds are leveraged to identify brown fat regions, though with an emphasis on relative uptake values due to dataset limitations (lack of patient weight or PET dose for absolute SUV calculation). The difference is that while BARCIST is aimed at consistent human analysis and reporting, such rules are integrated into an automated pipeline that additionally performs image modification (suppression of the brown fat signal) to aid interpretation.

In summary, brown adipose tissue is a double-edged sword in PET imaging: biologically fascinating and clinically significant, yet often an unwanted artifact in cancer imaging. The literature establishes its importance and provides a basis (both intuitive and formal) for how it can be identified using combined PET and CT information. This forms the foundation for the techniques developed in this project.



2.1.2 Deep learning for medical imaging analysis

Over the past decade, deep learning has revolutionized the field of medical image analysis [15]. Convolutional neural networks (CNNs), in particular, have become the method of choice for many imaging tasks such as classification, detection, and segmentation of medical images [15]. Unlike earlier pattern recognition approaches, CNNs can automatically learn complex feature hierarchies from raw image data, enabling high performance given sufficient data. A 2017 survey by Litjens *et al.* [15] reviewed over 300 papers applying deep learning in medical imaging and concluded that CNNs were achieving state-of-the-art results in various applications including tumor detection, organ segmentation, and disease classification. The success of deep learning in this domain can be attributed to its ability to handle the variability and noise in medical images by learning robust feature representations, as well as the growing availability of high-performance GPUs and large annotated datasets in some areas [15]. CNNs were chosen for this project’s classification task due to their established success with image data [15], suitability for the available dataset size (when using transfer learning) [16], and robust performance compared to, for instance, Vision Transformers which might require larger datasets for optimal performance without pre-training on very large medical datasets.

For image classification tasks (determining the presence or absence of a condition from an image), CNNs have demonstrated remarkable accuracy [15] [16]. For example, in radiology, CNNs have been used to classify abnormalities in chest X-rays, often reaching expert-level performance. A critical enabler for many of these successes is *transfer learning*. Medical datasets are often limited in size, so a common approach is to take a CNN pre-trained on a large dataset of natural images (such as ImageNet) and fine-tune it on the medical imaging task [16]. Shin *et al.* (2016) [16] showed that transfer learning can significantly improve CNN performance for tasks like lesion detection in CT and mammography, compared to training from scratch. By fine-tuning pre-trained networks, one can leverage learned low-level features (edges, textures, shapes) that are somewhat universal, while

adapting the high-level layers to the specific medical task [16]. In the Phase 1 classifier of this project, this strategy is adopted by using a ResNet-50 CNN architecture [9] pre-trained on ImageNet and then re-training it on PET MIP images for brown fat detection.

The ResNet (Residual Network) architecture introduced by He *et al.* (2016) [9] is particularly relevant because of its proven efficacy in image classification tasks. ResNet-50 is a deep CNN with 50 layers that uses residual learning via skip connections to ease the training of very deep networks [9]. Its design helps mitigate the vanishing gradient problem and allows successful training of networks far deeper than previous designs [9]. ResNet-50 (and variants) have been widely adopted in medical image analysis when complex classification or feature extraction is needed, often as a backbone architecture [9] [15]. By using ResNet-50 in Phase 1, a strong representational capacity is ensured for detecting subtle patterns that might indicate brown fat presence in the MIP images (for instance, the CNN can learn to recognize the symmetric pair of hotspots in the supraclavicular area as a pattern).

Another major use of deep learning in medical imaging is segmentation: delineating structures or regions of interest within images [15]. The breakthrough in this area came with the U-Net architecture by Ronneberger *et al.* (2015) [17]. U-Net is a CNN designed for biomedical image segmentation that features an encoder-decoder structure with skip connections between corresponding levels of the encoder and decoder [17]. This enables precise localization (via the skip connections) combined with contextual understanding (via the deep encoder) [17]. U-Net and its many variants have become the de facto standard for medical image segmentation tasks [15] [17] [18]. They have been applied to segment organs, tumors, lesions, and more in modalities ranging from microscopy to MRI and CT.

Building on U-Net, the field has developed even more robust segmentation frameworks. One notable example is nnU-Net (no-new-Net) by Isensee *et al.* (2021) [18]. nnU-Net is not a fixed architecture but a self-configuring pipeline that automatically adapts U-Net models to a given dataset’s properties [18]. It won numerous medical segmentation challenges by intelligently tuning preprocessing, architecture, and hyperparameters [18]. The creators of nnU-Net demonstrated that a properly configured U-Net can achieve top performance across diverse tasks without manual tailoring [18]. This is relevant because the TotalSegmentator tool employed in this project is built on ensembles of nnU-Net models for multi-organ segmentation [11] [18].

TotalSegmentator, introduced by Wasserthal *et al.* (2023) [11], is an open-source tool that provides automatic segmentation of 117 (updated from the initial 104) anatomical structures from CT images. It covers a comprehensive set of organs and tissues, including major bones, muscles, organs, and adipose tissues [11]. TotalSegmentator uses the nnU-Net framework [18] under the hood, which means it benefits from the well-optimized nature of nnU-Net for CT segmentation tasks. While TotalSegmentator has the capability to segment fat using its `tissue_types` task (though this specific functionality is not part of the regular version and may require a license), in this project, a Hounsfield Unit (HU) filter (-190 to -30 HU) [10] was found to yield better results for general adipose tissue masking, was faster, and required less computational resources. Therefore, instead of using TotalSegmentator for fat segmentation, its capabilities were leveraged to segment the liver. The segmented liver (combining Coinaud segments if provided by TotalSegmentator to form the whole liver) was then used to measure the average PET uptake within the liver. This liver uptake served as a patient-specific reference: a PET uptake greater than 1.5 times the liver uptake, identified within the HU-defined fat mask, was the primary criterion for classifying a voxel as brown fat. This relative approach negates the need for patient weight or PET dose information (which were not

available in the project’s dataset) and is likely more accurate and robust than using absolute PET uptake values. This methodology yielded noticeably better results compared to earlier attempts, which included (1) a classical rule-based method for masking both fat (-190 to -30 HU) and liver (40-70 HU), and (2) directly using the TotalSegmentator BAT fork for CT by Jørgensen *et al.* (2024) [13], which seemed to identify only a portion of the brown fat, possibly due to its reliance solely on CT data and subtle CT patterns. The availability of TotalSegmentator [11] for robust liver segmentation was invaluable for Phase 2 of this project, saving the effort of training a new segmentation model from scratch for this reference organ.

Finally, it is worth noting prior specific efforts to segment or quantify brown fat in imaging. Before deep learning became prevalent, researchers often used semi-automated methods. For instance, a common approach was to apply a fixed CT threshold to identify fat voxels and then within those, apply a PET SUV threshold to pick out active brown fat, essentially implementing criteria like BARCIST [8] in software. This approach was used in several human studies of BAT. However, threshold-based methods can be crude and may require manual adjustments or region-of-interest definitions by the user. Hussein *et al.* (2017) [12] presented one of the earliest attempts at fully automated brown fat segmentation from PET/CT using more advanced techniques. They proposed a system that first detects brown fat regions on PET via intensity clustering and then refines the segmentation using a random-walk algorithm on both PET and CT to separate brown and white fat [12]. Their method was able to quantify both white and brown fat volumes and activities in patients, demonstrating the feasibility of automated adipose tissue analysis on PET/CT [12]. With the rise of CNNs, newer approaches have emerged. Jørgensen *et al.* (2024) [13] developed a model for segmenting brown adipose tissue in the neck region using deep learning on CT images alone. They trained an nnU-Net [18] specifically to label supraclavicular BAT on CT (integrated into TotalSegmentator workflow [11]) and achieved a mean Dice score of 0.78 against manual annotations. Similarly, Erdil *et al.* (2024) [14] took a different angle by using deep learning to predict BAT presence from CT and thereby segment it indirectly. They trained an Attention U-Net (a U-Net [17] variant) to predict the PET uptake in known brown fat regions from just the CT image, effectively creating a model that knows how brown fat looks on CT and where it will light up on PET [14]. Their method outperformed simple HU thresholding, with 23–40% better accuracy in segmenting BAT, and could distinguish BAT-positive vs BAT-negative individuals with an AUC of 0.8 [14]. These state-of-the-art works indicate that deep learning can identify brown fat based on anatomical imaging alone to a significant extent. However, in a clinical PET/CT scenario, the PET data is available, so the approach in this project uses both modalities in tandem.

In summary, the literature provides several building blocks and inspiration for this project:

1. Brown fat can be identified by combined PET and CT criteria (from clinical studies [2] [4] [5] and BARCIST guidelines [8]).
2. CNNs and transfer learning enable image classification on medical images [15] [16], leveraged here for brown fat presence detection.
3. Modern segmentation networks (U-Net [17], nnU-Net [18]) can automate anatomical segmentation; specifically, tools like TotalSegmentator [11] allow off-the-shelf liver segmentation for reference uptake calculation, while a HU-based approach [10] was chosen for fat masking.

4. Prior attempts to automate brown fat segmentation motivate the hybrid approach developed (an automated pipeline akin to clinical reasoning: find fat, check PET uptake relative to a reference, which parallels earlier threshold methods but is augmented with robust reference organ segmentation [12] [13] [14]).
5. No previous work has specifically focused on *removing* or suppressing brown fat in PET images; this appears to be a novel contribution, taking the next step after detection/segmentation to actually modify the image for improved clarity.

2.1.3 Summary of literature insights

From the above, an intersection of a medical imaging challenge and AI solutions is evident. Brown fat in PET/CT is a known issue [2] [4] [5] that can potentially be addressed by algorithmically combining information from CT (to localize fat [10]) and PET (to gauge metabolic activity relative to a reference [8]). The evolution of deep learning provides powerful tools to implement this: classification networks to detect patterns and segmentation networks to delineate reference structures [15] [16] [17] [18]. This project stands on the shoulders of these developments. The following sections will describe how a ResNet-based classifier [9] was applied for Phase 1 and a hybrid method utilizing HU-based fat masking [10] and TotalSegmentator-based liver segmentation [11] for Phase 2, building upon the ideas gathered from literature.

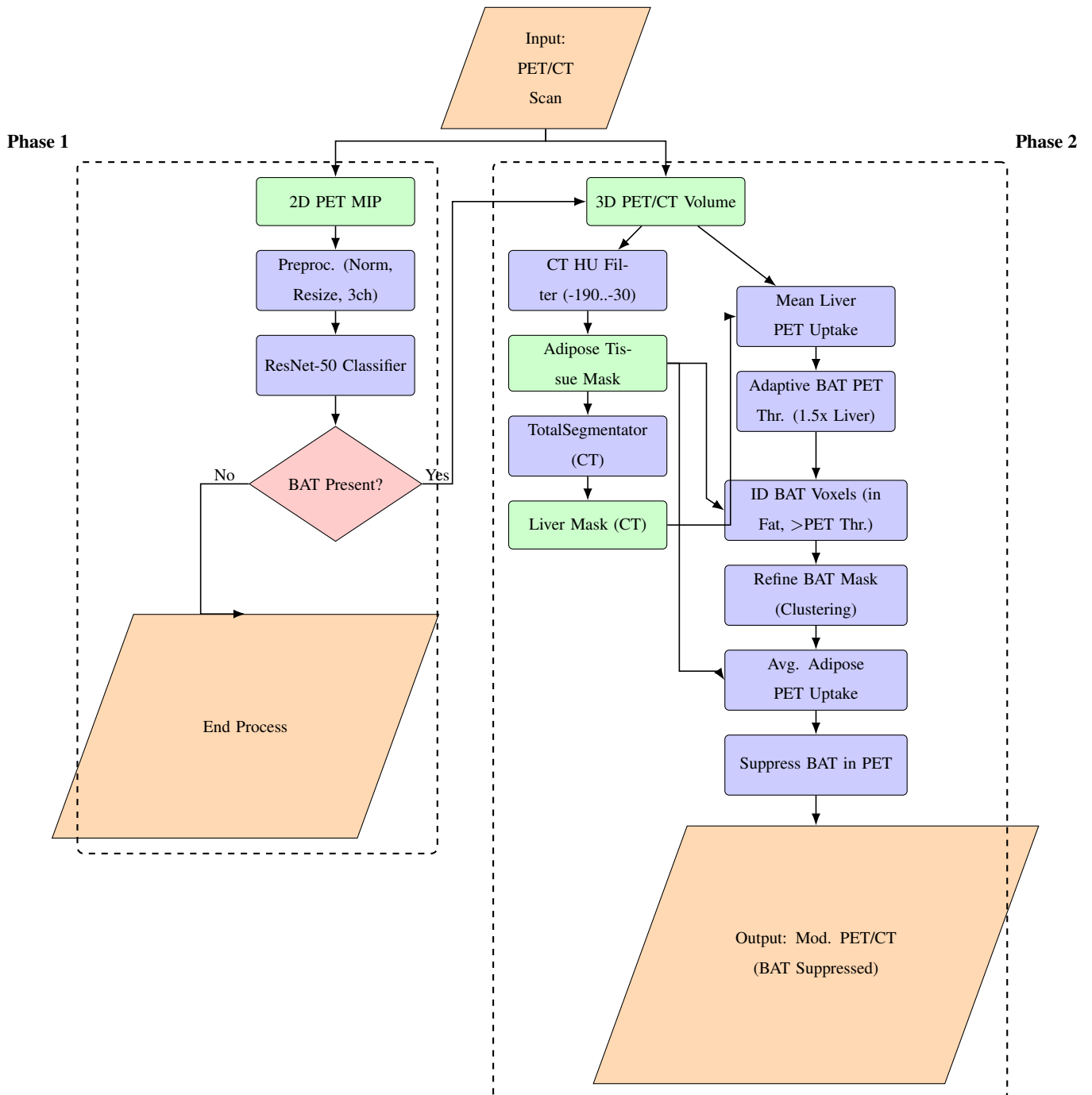
Table 2.1: Project Requirements (MoSCoW Method)

Priority Category	ID	Requirement Description
Must Have	M1	Develop a 2D Convolutional Neural Network (CNN) classifier to automatically detect the presence of significant brown adipose tissue (BAT) in PET MIP images.
	M2	Achieve a robust classification performance for BAT presence (e.g., F1-score > 0.85) on a test dataset.
	M3	Implement a 3D processing pipeline to segment and locate brown fat regions within full PET/CT volumes when BAT is detected.
	M4	The 3D pipeline must utilize CT Hounsfield Unit (HU) filtering to identify general adipose tissue regions.
	M5	The 3D pipeline must incorporate segmentation of a reference organ (e.g., liver) from CT to establish patient-specific adaptive PET uptake thresholds for BAT detection.
	M6	Suppress the PET signal intensity in the identified 3D brown fat regions to reduce its visual impact.
	M7	Ensure the BAT suppression process preserves pathological uptake and other critical physiological signals in non-BAT regions.
	M8	The system must be capable of processing PET/CT data in DICOM format, for both 2D MIP classification and 3D volume processing.
	M9	Evaluate the performance of the BAT detection classifier (Phase 1) and the effectiveness of the BAT removal pipeline (Phase 2) using appropriate metrics and qualitative review.
Should Have	S1	The 2D CNN classifier should be based on a pre-trained architecture (e.g., ResNet-50) and fine-tuned for the BAT detection task.
	S2	Suppressed BAT voxels in the PET image should be set to a level approximating background tissue (e.g., average uptake of general adipose tissue) to maintain a natural image appearance.
	S3	The system should output a modified 3D PET volume where the brown fat signal has been suppressed.
	S4	Qualitative evaluation should include visual comparison of PET images before and after BAT suppression to demonstrate effectiveness.
	S5	The 3D BAT removal process should effectively target common BAT locations such as cervical, supraclavicular, and paravertebral regions.
Could Have	C1	Integrate Phase 1 and Phase 2 so that the 3D removal pipeline is automatically triggered based on the output of the 2D classifier.
	C2	Implement basic post-processing techniques for the generated BAT mask, such as small cluster removal or minor morphological operations, to improve segmentation robustness.
	C3	Provide clear documentation or a user guide explaining how to set up and run the developed tools/models.
	C4	Develop a method to indicate or annotate on the processed images/report that BAT suppression has been applied and in which regions.
	C5	Investigate the classifier's performance on MIPs with atypical BAT distributions or varying intensities.
Won't Have (this time)	W1	Full clinical integration into hospital PACS systems or development of a production-grade real-time clinical decision support tool.
	W2	An interactive graphical user interface (GUI) for users to manually review, approve, or adjust BAT segmentations/suppressions.
	W3	Development and training of a novel, complex end-to-end 3D deep learning model for direct BAT segmentation from scratch, given the constraints on annotated 3D data for this project.
	W4	Extension of the system to support PET radiotracers other than ¹⁸ F-FDG or other imaging modalities like PET/MR without specific data and model adaptation.
	W5	Comprehensive suppression of other types of physiological uptake (e.g., extensive muscle uptake, widespread inflammation) beyond brown fat.

Chapter 3

Methodology and Implementation

Figure 1: Overview



The system’s architecture, visualized in Figure 1, is a two-phase pipeline designed to first efficiently detect the presence of brown adipose tissue (BAT) and then perform a targeted removal of its signal. This approach partitions the problem into a lightweight classification task (Phase 1) followed by a more computationally intensive 3D image processing workflow (Phase 2). The second phase can then be initiated based on the results of the first.

Phase 1 acts as a screening filter. Instead of processing the entire high-resolution dataset from the Input: PET/CT Scan, it uses the 2D PET MIP, which is a simplified summary view of the 3D scan. This 2D image undergoes standard deep learning image preprocessing (Preproc. (Norm, Resize, 3ch)), including normalization and resizing to fit the input dimensions of the neural network. The core of this phase is a ResNet-50 Classifier, a deep convolutional neural network pre-trained on the ImageNet dataset and fine-tuned from our annotated dataset to classify images for the presence of BAT. The output of this classifier flows into a decision node (BAT Present?). If the model does not detect significant BAT, the process can be concluded, leaving the original scan unchanged (End Process). If BAT is detected, the “Yes” branch indicates that the scan is a candidate for the subsequent removal workflow. In the current system, initiating Phase 2 is a manual step taken by the user for these flagged cases; a fully automated trigger connecting the two phases is a direction for future work.

Phase 2 executes the main image modification logic on the 3D PET/CT Volume. This phase involves two parallel data processing streams. The first stream processes the CT data to understand the patient’s anatomy. A CT HU Filter (-190..-30) applies a simple threshold to the CT image’s pixel values (Hounsfield Units) to create a boolean Adipose Tissue Mask, identifying all regions of fat. Concurrently, an external, pre-trained deep learning model, TotalSegmentator (CT), is used to generate a precise Liver Mask (CT). The second stream uses this anatomical information to guide the PET image processing. The Liver Mask (CT) is used to calculate the Mean Liver PET Uptake, establishing a patient-specific reference for normal metabolic activity. This reference is used to set an Adaptive BAT PET Thr. (1.5x Liver), which is a dynamic threshold.

The two streams converge at the ID BAT Voxels (in Fat, >PET Thr.) step, which performs a logical AND operation: it flags any voxel that is both within the Adipose Tissue Mask and has a PET signal intensity greater than the adaptive threshold. This initial mask of BAT candidates is cleaned up in the Refine BAT Mask (Clustering) step, which removes small, isolated clusters likely to be image noise. Finally, the signal is suppressed. To avoid leaving an artificial-looking hole in the image, the system first calculates the Avg. Adipose PET Uptake across all fat tissue. The Suppress BAT in PET function then overwrites the high-intensity BAT voxels with this calculated average value. The final Output: Mod. PET/CT (BAT Suppressed) is a “cleaned” version of the scan, ready for clinical review with reduced risk of false positives from brown fat.

3.0.1 Phase 1: 2D CNN classifier for brown fat presence

Data collection and preprocessing

For Phase 1, a dataset of PET/CT scans labeled for the presence or absence of active brown fat was required. A retrospective dataset of 451 whole-body ¹⁸F-FDG PET/CT studies from adult patients was assembled, drawn

from a hospital PACS database with appropriate approvals. Approximately 50 of these cases exhibited some level of brown fat. An experienced radiologist reviewed each scan’s MIP images to assign labels. Initially, four labels were assigned: 0 (none), 1 (minimal), 2 (moderate), and 3 (high brown fat uptake). The dataset was split into a training set of 360 cases and an independent test set of 91 cases. During training, 20% of the training data was used for validation. The dataset was characterized by an unequal representation of BAT-present and BAT-absent cases.

Instead of using the full 3D data for the classifier, 2D maximum intensity projection images were used. Each PET/CT scan typically has MIP images generated by the scanner software - these are images where the maximum PET intensity along rays through the volume is projected into a 2D image. The anterior (frontal) MIP was chosen as the input for the classifier because it provides a quick summary of metabolically active regions in the body. Brown fat, when present, usually appears on the MIP as symmetric bright spots in the shoulder/neck area [2] [5], a pattern intended for the CNN to learn. Using MIPs drastically reduces input dimensionality, making the classification task more tractable and reducing training time. A 2D MIP was extracted or reconstructed from each PET scan in the dataset. Each MIP was saved as a grayscale image (pixels representing the maximum PET uptake value or intensity along each projection line). These images were normalized by rescaling pixel intensities to a 0–255 range (8-bit) based on the dynamic range of uptake relevant for visualization (any extremely high uptake, like in the bladder or brain, was capped to not dominate the scaling). No additional filtering was applied; reliance was placed on the CNN to handle noise or irrelevant uptakes. Absolute SUV values were not used for classification decisions due to the unavailability of patient weight or PET dose information in the dataset, which are necessary for accurate absolute SUV calculation [8].

Since most CNN models (including ResNet-50 [9]) expect 3-channel (RGB) images, the single-channel, grayscale PET MIP was replicated into three channels ($R=G=B$ = the PET intensity) to use the standard ResNet-50 architecture without having to modify the model. The MIP images were also resized to 224×224 pixels to match the input size of ResNet-50 [9], using bilinear interpolation. After these steps, the input to the CNN was a $224 \times 224 \times 3$ image representing the PET MIP. For the final binary classification (see Section IV-A), the label was 1 for BAT-present and 0 for BAT-absent.

To augment the training data and help the model generalize, several data augmentation techniques were applied randomly during training [16]. These included small rotations (up to $\pm 15^\circ$) to account for slight differences in patient positioning, horizontal flips (because while the actual left-right pattern of brown fat is symmetric, flipping doesn’t change the symmetric pattern but could expose the model to general symmetry recognition), and slight brightness/contrast shifts (to simulate variations in PET image intensity scaling). Any augmentation that would destroy the symmetric nature of brown fat uptake was avoided (for example, vertical flips would put uptake in improbable locations, and therefore were not used). Augmentation was applied on the fly during training iterations.

CNN architecture and training

The ResNet-50 architecture [9] was selected for the classifier, given its strong performance on image recognition tasks and manageable size (50 layers). The network was initialized with weights pre-trained on ImageNet (a large natural image dataset), a standard practice to leverage learned features [16]. The final fully connected layer of ResNet-50 (which normally outputs 1000 class scores for ImageNet) was replaced with a new fully connected

layer appropriate for the classification task (initially 4 classes, then 2 classes) followed by a softmax activation. The PyTorch deep learning framework was used to construct and train the model.

During training, the cross-entropy loss function appropriate for classification was used. Training was conducted for 30 epochs (passes over the training set). An Adam optimizer [19] was used with an initial learning rate of 0.0003. Using a relatively low learning rate was beneficial to avoid destroying the pre-trained weights immediately; the aim was to fine-tune them gradually [16]. Performance was monitored on the validation set after each epoch, and if the validation loss did not improve for 5 consecutive epochs, early stopping was employed to prevent overfitting [15]. In practice, the model converged well before 30 epochs (the best model was usually around epoch 10–15, after which validation loss plateaued).

After training, the model weights were fixed at the epoch with the best validation F1-score (a balance of precision and recall) and evaluated the final performance on the independent test set of 91 cases. The primary metric for optimization was F1-score, since it balances precision (avoiding false alarms that say brown fat is present when it isn't) and recall (avoiding misses of actual brown fat cases). High recall was particularly important to ensure Phase 2 could be triggered for the most relevant cases, therefore it was favored over precision.

3.0.2 Phase 2: 3D Brown Fat segmentation and removal

Phase 2 is executed for scans classified as BAT-present by Phase 1. The input is the full PET/CT volume, and the output is a modified PET/CT where brown fat uptake is suppressed in the PET images. This phase involves: identifying adipose tissue using a CT Hounsfield Unit (HU) filter [10], segmenting the liver using TotalSegmentator [11] to establish a reference PET uptake, identifying BAT within the fat mask using an adaptive PET uptake threshold relative to liver uptake [8], and finally modifying PET voxel values to suppress the brown fat signal.

Initial Rule-based detection (Prototype)

Before integrating more sophisticated methods like deep-learning segmentation, a prototype using straightforward rule-based criteria was developed, both to validate the concept and to serve as a baseline. This initial approach attempted to identify brown fat regions by applying threshold conditions on the PET and CT images, akin to how a human or the BARCIST criteria [8] might do. For each voxel, it was checked if its CT Hounsfield unit (HU) falls within the range of adipose tissue and if simultaneously the PET uptake value was greater than a multiple of liver/soft tissue uptake. The liver/soft tissue was masked using a threshold of (40 to 70 HU), a very tentative approximation. For fat, CT between -190 and -30 HU (this range captures both white and brown adipose tissue [10]; brown fat might be slightly higher in density but largely overlaps with fat range [5] [8]) and PET uptake > 1.5 liver/soft tissue (a common literature value for active BAT [8]) were initially used. Connected-components analysis was used to remove clusters of a few voxels (likely noise).

This rule-based method did indeed highlight known brown fat areas in many cases; for example, in a scan with clear brown fat, the supraclavicular regions lit up as expected. However, some issues were encountered:

1. In some scans, the global PET uptake levels were higher or lower than usual (due to patient metabolic differences or imaging parameters). A fixed PET uptake cutoff was too low for some cases (causing even mild uptake in fat to be flagged, potentially false positives), and too high for others (missing faint brown fat

in a patient whose overall uptake is low).

2. The CT HU threshold alone is not perfect: occasionally, a structure like a part of the heart or blood vessel with fat around it could get partially included if the CT values border on the threshold or if there was motion causing smearing of HU. A more robust identification of fat regions was needed.
3. Hypothetically, an FDG-avid tumor adjacent to fat (e.g., a retroperitoneal lymph node near the kidney) could have voxels at its margin misclassified if they fell within the fat HU range and had high PET uptake.

These observations led to a refined approach with a more sophisticated segmentations and an adaptive threshold. The rule-based attempt, however, was a useful stepping stone to confirm that looking at “PET uptake within fat regions” is a viable strategy for finding brown fat.

Adipose tissue masking and liver segmentation

As in the prototype, a Hounsfield Unit (HU) based filter (-190 to -30 HU) [8] [10] was used to create a general adipose tissue mask from the CT images. This HU-based approach was found to be effective, fast, and less computationally demanding for defining the regions where BAT might be located. Although TotalSegmentator [11] provides a fat segmentor under licence (tissue_types task, subcutaneous + torso fat), it was found to be slower and surprisingly less effective.

Separately, to establish a patient-specific adaptive PET uptake threshold, the TotalSegmentator tool [11] (which can segment 117 structures) was used, but specifically for segmenting the liver from the CT volume. TotalSegmentator [11] was run in its default CT segmentation mode, and the liver mask was extracted. This tool provides robust liver segmentation, which is critical for obtaining a reliable reference PET uptake. The liver segmentation by TotalSegmentator [11] is fast, typically taking less than one minute on a GPU and less than five minutes on a CPU. The PET and CT volumes are inherently registered, but the CT-derived liver mask (at high resolution) needed to be downsampled to PET resolution to calculate mean liver PET uptake. This was done by determining the fraction of each PET voxel’s volume occupied by the liver mask; if over 0.5, the PET voxel was included in the liver region for PET analysis.

The use of TotalSegmentator [11] was confined to liver segmentation; it was not used for fat segmentation in the final pipeline due to the aforementioned preference for the HU-filter approach for fat. This focused use of TotalSegmentator [11] for liver segmentation provided a reliable reference organ PET uptake value.

Adaptive PET uptake thresholding for BAT detection

Rather than a fixed absolute PET uptake threshold, a relative threshold based on each scan’s liver uptake was implemented. The mean PET uptake value was calculated within the TotalSegmentator-derived liver mask [11]. Then, the brown fat PET uptake threshold was set as: $\text{Threshold}_{\text{PET}} = \text{mean PET uptake}_{\text{liver}} \times 1.5$. This relative threshold (1.5 times the mean liver uptake) helps account for individual patient and scanner variability [8]. Fat voxels (identified by the -190 to -30 HU mask [8] [10]) with PET uptake above this patient-specific threshold were classified as part of a brown fat region. Connected-component analysis was performed on these voxels, and clusters smaller than 8 voxels at PET resolution were ignored to eliminate noise. The output is a binary mask

of “brown fat regions” in the PET volume. This approach, using a HU rule for the fat mask [8] [10] and a liver multiplier of uptake from the TS liver mask [11] to find BAT, worked best.

The adaptive threshold significantly improved the consistency of detection.

Mask refinement and PET suppression

The brown fat mask obtained above sometimes had a few irregularities that were refined. The brown fat mask was sometimes refined by slight dilation (by 1 voxel) before intersecting with the PET threshold to account for partial volume effects [20] and avoid residual uptake at edges.

Once the brown fat voxel mask was finalized, the PET image was altered. Instead of zeroing out voxels, which might look an unnatural, their intensity was set to a value representing background or normal inactive adipose tissue. Specifically, the PET uptake value in detected BAT voxels was set to the average PET uptake value measured across all adipose tissue in the whole body (as defined by the HU mask [8] [10]). This method was chosen over the initially considered 0.5x liver uptake.

This modification was applied to the PET volume, leaving the CT volume untouched. The rationale for not changing CT is that only the PET signal needs suppression; the CT still shows fat in those areas. Radiologists can still see there was fat there if needed, but the PET no longer glows. After suppression, the new and original 3D volume was kept.

Visual outputs were created for evaluation: the original vs modified PET images side by side. These help in qualitatively assessing the result.

Before: symmetric bright areas are brown fat

After: brown fat suppressed



illustrates an example of the effect of Phase 2. Figure 2(a) shows a PET slice with intense symmetric uptake in the neck/shoulder region (typical brown fat). Figure 2(b) shows the PET slice after brown fat suppression; the previously hot regions are no longer visible. The rest of the body’s uptake (e.g., heart, bladder) is unchanged.

The Phase 2 pipeline was implemented in Python using NumPy and SimpleITK for image processing and the TotalSegmentator API [11] for segmentation. The liver segmentation step with TotalSegmentator [11] is the most computationally intensive part of Phase 2. The rest of the processing (thresholding, etc.) is very fast (a few seconds per scan). Thus, Phase 2 can be performed in near real-time after a scan, especially if a GPU is available to accelerate the segmentation step. If integration into clinical workflow were desired, one could potentially pre-segment the fat on the CT as the scan is acquired (since CT is quick) and then only apply the threshold logic once the PET is done.

To summarize Phase 2: it is a hybrid approach that uses a deep learning model (TotalSegmentator [11] for liver CT segmentation) combined with rule-based criteria (HU filter for fat [8] [10], relative PET uptake [8]) to identify brown fat regions, and then modifies the PET image to suppress those regions. The approach was informed by

initial experiments and guided by known physiology [1]. Next, the performance of both phases will be evaluated and the results presented.

Chapter 4

Testing

To validate the performance and effectiveness of the two-phase system, a multi-faceted testing strategy was employed, addressing the distinct goals of the 2D classifier (Phase 1) and the 3D removal pipeline (Phase 2).

The core of the Phase 1 evaluation was a quantitative analysis of the classifier’s performance on an independent test set of 91 PET MIP images, which were not used during training or validation. The primary metrics for this test were:

- Accuracy: Both for the 4-class grading (levels 0-3) and for a simplified binary task (BAT-present vs. BAT-absent) to assess overall correctness.
- F1-Score, Precision, and Recall: These metrics were calculated to understand the balance between correctly identifying BAT cases (recall) and avoiding false alarms (precision), with a particular emphasis on achieving high recall for the BAT-present class.
- Confusion Matrix: A matrix was generated to visualize the specific types of misclassifications made by the 4-class model (e.g., confusing minimal BAT with no BAT).

In addition to quantitative metrics, a qualitative test using Gradient-weighted Class Activation Mapping (Grad-CAM) was planned. This technique generates heatmaps to visualize the regions of the input MIP image that the CNN focuses on for its prediction, providing a visual check to ensure the model was correctly identifying anatomical areas typical for brown fat.

For Phase 2, the testing strategy combined qualitative and quantitative approaches to evaluate the brown fat removal pipeline on scans identified as BAT-positive.

The qualitative evaluation consisted of:

A side-by-side visual comparison of PET/CT images and their corresponding MIPs before and after the suppression algorithm was applied. This was to confirm the effective removal of BAT hotspots and to ensure the resulting image appeared natural without obvious artifacts.

A specific check on cases containing known non-BAT pathological uptake to verify that these clinically significant regions were preserved and not inadvertently altered by the suppression process.

The quantitative evaluation was designed to measure the specific impact of the algorithm on the 3D PET data:

- BAT Activity Reduction: Measurement of the mean PET signal intensity within the automatically generated BAT mask, both before and after suppression, to quantify the degree of reduction.

- **Reference Organ Stability:** Calculation of the mean PET signal in reference organs not targeted by the algorithm (e.g., the liver and brain) to confirm their signal stability remained within a minimal tolerance ($\pm 1\%$ change).
- **Loop Closure Test:** A final validation step involved generating a new MIP from the suppressed 3D PET volume and feeding it back into the trained Phase 1 classifier. The test's success criterion was the re-classification of the processed MIP from "BAT-present" to "BAT-absent."

4.0.1 Unit and Component Tests

Beyond the end-to-end evaluation of the system, a series of unit and component tests were implemented to ensure the reliability of individual functions and modules. This provides a foundational level of confidence in the building blocks of the pipeline. These tests were designed to cover key data processing and algorithmic logic steps:

Data Loading and Preprocessing: A test was created to verify that the `_load_dicom` function correctly reads pixel data, normalizes the intensity values to the expected 0-255 range, and returns an array of the correct data type.

Model Forward Pass: The ResNetBF model was tested to ensure that a forward pass with a dummy input tensor of the correct shape executes without error and produces an output tensor of the expected dimensions (`batch_size` \times `num_classes`), confirming the architectural integrity.

Mask Refinement Logic: The `remove_small_3d_clusters` function was unit-tested with a synthetic 3D boolean mask containing several distinct clusters of varying sizes. Assertions were used to verify that only the clusters smaller than a specified voxel threshold were removed, while larger clusters remained intact.

Heuristic Suppression Logic: A comprehensive heuristic test for the core `suppress_bat` function was implemented. This involved creating a synthetic PET/CT volume with predefined regions simulating: (a) an area of brown fat (high PET uptake in a low-density CT region), (b) a pathological lesion (high PET uptake in a soft-tissue CT region), and (c) a reference liver organ. The test asserts that after running the pipeline, the synthetic BAT signal is correctly suppressed to background levels while the synthetic lesion signal remains entirely unchanged, thus verifying the algorithm's fundamental ability to selectively target BAT.

Chapter 5

Results and Evaluation

This section presents the outcomes for both Phase 1 and Phase 2. Phase 1 classifier’s accuracy in detecting brown fat presence is evaluated, and Phase 2’s effectiveness in removing brown fat signals from PET images is assessed, both qualitatively and quantitatively. The initial rule-based method is also compared with the improved method in Phase 2 to highlight the gains from using segmentation and adaptive thresholding.

5.0.1 Phase 1 Classifier performance

The ResNet-50 classifier [9] for brown fat presence was evaluated on the test set of 91 scans. Initially, a 4-class classifier (0: none, 1: minimal, 2: moderate, 3: high BAT) was developed. This 4-class model achieved an F1-score of approximately 0.907. However, most misclassifications occurred with class 1 (minimal brown fat), and overall accuracy was in the range of 78-82 cases out of 91 being correctly classified.

Given these challenges, a decision was made to simplify the problem to a binary classification: BAT-absent (combining original labels 0 and 1) versus BAT-present (combining original labels 2 and 3). This binary classifier yielded much better results, with accuracies improving to 86-89 out of 91 cases being correctly classified, indicating reliable distinction between scans with and without significant brown fat. Out of scans in the test set that had brown fat (BAT-present), the binary classifier correctly identified most of them, and similarly, most scans without brown fat were correctly classified as BAT-absent. This corresponds to high recall and precision for the BAT-present class. High recall was a priority to ensure Phase 2 would trigger for the vast majority of relevant cases.

To further understand the model’s decision-making process, Gradient-weighted Class Activation Mapping (Grad-CAM) was employed. Grad-CAM produces heatmaps that highlight the image regions most influential for a given classification. As illustrated in Figure 3, for true positive cases, the Grad-CAM overlays consistently highlighted the supraclavicular and cervical regions. This confirms that the network was correctly focusing on the typical anatomical locations of brown fat to make its predictions and not on spurious correlations, increasing confidence in the model’s performance.

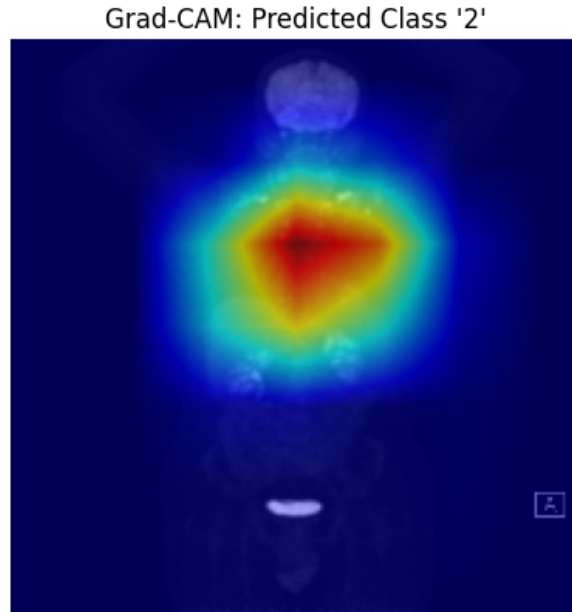


Figure 5.1: GRAD-CAM visualization for a true positive case. The original PET MIP image classified as BAT-present. The same MIP with the Grad-CAM heatmap overlay, showing the classifier’s attention on the supraclavicular area (red area).

Table 5.1: Summary of Results

Metric	Score / Observation
Phase 1: 2D BAT Detection (Test Set: 91 scans)	
Binary Classification Accuracy (BAT Present/Absent)	89/91 (97.8%)
4-Class Classification Accuracy (Levels 0-3 BAT)	84/91 (92.3%)
4-Class Weighted Average F1-Score	0.907
4-Class Macro Average F1-Score	0.686
<i>Per-Class F1-Scores (4-Class):</i>	
Class 0 (No BAT)	0.958 (Precision: 0.930, Recall: 0.988)
Class 1 (Minimal BAT)	0.286 (Precision: 0.500, Recall: 0.200)
Class 2 (Moderate BAT)	0.500 (Precision: 1.000, Recall: 0.333)
Class 3 (High BAT)	1.000 (Precision: 1.000, Recall: 1.000)
Phase 2: 3D BAT Removal	
BAT PET Activity Reduction	Significant
-(approaching background adipose levels)	
Reference Organ PET Stability (e.g., Liver, Brain)	<1% change post-processing
Non-BAT Pathological Uptake Preservation	Generally preserved;
-instances of minor tumor edge effect noted (attributed to potential misregistration/movement)	

The performance of the Phase 1 classifier, which achieved an F1-score of approximately 0.907 for a 4-class BAT grading task and an accuracy of 86-89 out of 91 test cases for a simplified binary BAT presence detection, indicates a strong capability for identifying scans with brown fat. This addresses the critical first step of detection, which has also been an aim of other AI models; for instance, Erdil et al. [14] used CT data alone to predict BAT presence with an Area Under the Curve (AUC) of 0.8. While direct comparison of segmentation performance metrics is challenging due to differing methodologies, datasets, and specific objectives, other studies such as Jørgensen et al. [13] have reported mean Dice scores around 0.78 for segmenting supraclavicular BAT from CT images using deep learning. This project’s Phase 2, however, diverges from studies primarily focused on BAT segmentation or standardized quantification as per criteria like BARCIST 1.0. Instead, it innovatively builds upon similar foundational principles (like combining CT-based fat identification with PET uptake assessment) within an automated pipeline that utilizes both PET and CT data, including deep learning (TotalSegmentator) for robust liver segmentation to establish adaptive thresholds. The primary novel contribution of this work is its focus on actively suppressing the BAT signal in the final PET images to enhance diagnostic clarity for oncological interpretations, a goal not explicitly addressed by the aforementioned segmentation or prediction studies.

Qualitatively, the classifier’s decisions were inspected. For instance, a false negative case might occur with very faint or atypically located brown fat. A hypothetical false positive scenario could involve a scan with no brown fat but with high uptake in structures like the trapezius muscles, appearing as symmetric shoulder-region uptakes on the MIP, which could be misinterpreted by the model as brown fat if such patterns are not well represented or differentiated in the training data. More training data with such confounders could further improve robustness.

The system was designed with the idea that if the classifier predicted BAT-present, Phase 2 would be initiated. However, currently, Phase 2 is not automatically run following Phase 1 classification; this integration is a potential future development.

In summary, the Phase 1 binary classifier provides a high-accuracy, automated screening for brown fat on PET MIPs. Its performance is adequate for the pipeline, ensuring that most cases with significant brown fat would be flagged for further processing (and perhaps automatically proceed to the next phase). It essentially achieves its goal of relieving a human from having to manually check every scan, only in a small fraction of cases (the missed ones) would a human need to catch what the AI missed. And even then, missing a brown fat case just means it wouldn’t be suppressed; it would still be visible to the radiologist who could recognize it, so the risk is not catastrophic, though it is something to minimize nonetheless.

5.0.2 Phase 2 Brown Fat removal results

Phase 2 was evaluated qualitatively by examining the “brown fat removed” PET images and quantitatively by measuring how much of the brown fat uptake was suppressed and whether any non-brown fat regions were inadvertently affected. The focus was on using relative PET uptake values (normalized to liver uptake) rather than absolute SUV ranges, as patient weight and PET dose information required for accurate absolute SUV calculation were not available [8], and absolute SUV is also less accurate for our purposes than relative uptake to liver.

For a subset of cases identified as BAT-present by Phase 1, the Phase 2 pipeline was run. Figure 2 (described earlier) is one such example: a patient with intense brown fat in the neck. Originally, the PET might show high uptake values in those fat areas. After applying the algorithm, these regions in the PET were reduced to

background levels (e.g., average uptake of general adipose tissue). Visually, the image looked “cleaner”; one could immediately notice that no abnormal uptake remained in the neck area. The CT, unchanged, still shows the fat, and if needed, one could annotate that “brown fat was here.” But the key is that the high PET uptake no longer risks being mistaken for something significant.

Another example, as seen in Figure 2, shows axial PET/CT slices through the supraclavicular region for a different patient. In the original images, panel (a), there are bilateral hotspots in the fat just above the clavicles. Panel (b) shows the same slice after suppression: the hotspots are gone. To ensure that no real lesions were removed, checks ensured that in each case any known tumors or other findings remained, by testing against a scan with known pathology.

To provide a comprehensive before-and-after view of the entire volume, new 2D MIPs were generated from the 3D PET volumes after BAT suppression. This allows for a direct comparison in the same format used by the Phase 1 classifier. Figure 5 demonstrates this transformation, where the prominent BAT signal visible in the original MIP is virtually removed in the suppressed MIP, resulting in a diagnostically cleaner image.

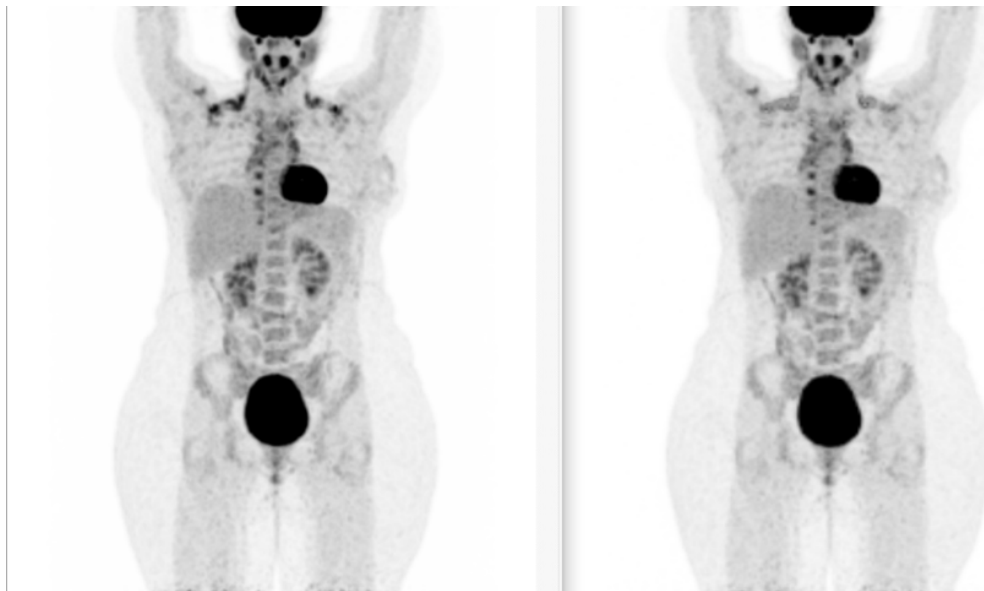


Figure 5.2: Visual effect of Phase 2 BAT suppression on PET MIPs. (Left) Original MIP from a BAT-positive scan, clearly showing intense supraclavicular (around the collarbone) uptake. (Right) The new MIP generated after the 3D suppression algorithm was applied. The BAT signal has been successfully removed.

Reference regions (like the liver and brain) were checked before and after - as expected, there was no change ($\leq 1\%$ difference purely from interpolation rounding). The algorithm only touches voxels within the fat mask above threshold [8] [10], so everything else is left intact. This is an important validation that other aspects of the PET image are not accidentally altered.

If Phase 2 ran on a Phase 1 false positive (where no actual BAT was present), the algorithm typically found no significant uptake within the fat mask that met the liver-relative threshold criteria [8]. Consequently, it made no substantial changes to the image. Therefore, even if Phase 2 runs unnecessarily, it tends to be harmless; it just doesn't do anything if no brown fat is actually present, because the conditions won't be met. This fail safe is nice because it means the cost of a false positive from Phase 1 is just some wasted computation, not a mis-altered image.

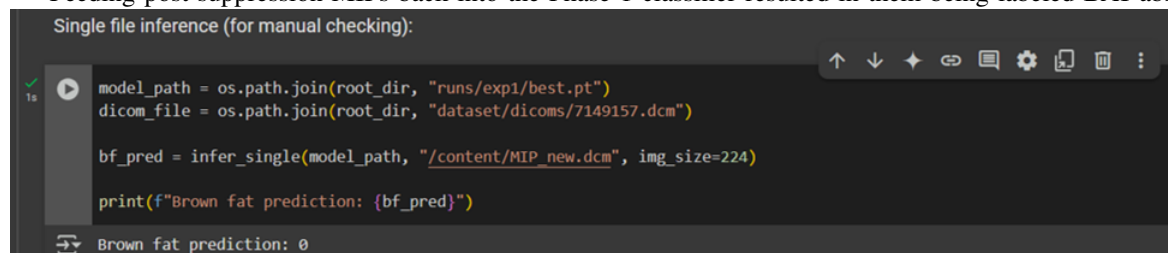
Comparing the final method (TotalSegmentator for liver segmentation [11], HU for fat mask [8] [10], adaptive threshold based on each patient’s liver [8]) to the initial rule-based prototype showed significant improvement. The final method had better recall and precision at the voxel level for identifying BAT and produced more anatomically plausible segmentations - unlike the previous method, no other tissues were significantly affected (original had the tendency to remove soft tissues, etc). Additionally, it produced more contiguous, anatomically plausible clusters.

Radiologist feedback on before and after images was generally positive. They noted that the majority of the brown fat in the cases was removed, especially in the neck and axilla. However, the paravertebral regions (along spine and chest) were less accurate. Additionally, it sometimes leaves behind the edges of the area with brown fat - this could be due to Flare or the resolution difference between PET and CT causing the fat mask to be inaccurate. In a case with other pathology, it was noted that, while the brown fat was successfully suppressed, small parts of some tumors in certain areas were also. This could perhaps be due to misregistration or patient movement, or moving body parts like the heart, between the scans.

A consideration for deployment is to indicate on the image or report that brown fat suppression has occurred. In a deployment scenario, one might want to somehow indicate on the image or report that brown fat was suppressed, so the radiologist is aware. This was not integrated in this project, but a feature (like an overlay or an annotation), to highlight the regions to indicate they were once brown fat could be future work.

A limitation observed was that very diffuse and mild brown fat uptake might not be caught if it fell below the adaptive threshold [8] or formed clusters too small to be considered significant. In such cases, the brown fat might remain in the image. However, diffuse mild uptake is usually not a big diagnostic problem.

Feeding post-suppression MIPs back into the Phase 1 classifier resulted in them being labeled BAT-absent.



The screenshot shows a Jupyter Notebook interface. At the top, it says "Single file inference (for manual checking):". Below this, there is a code cell with the following Python code:

```
model_path = os.path.join(root_dir, "runs/exp1/best.pt")
dicom_file = os.path.join(root_dir, "dataset/dicoms/7149157.dcm")

bf_pred = infer_single(model_path, "/content/MIP_new.dcm", img_size=224)
print(f"Brown fat prediction: {bf_pred}")
```

Below the code cell, the output is displayed: "Brown fat prediction: 0".

A key validation of the pipeline’s success is its ability to “close the loop” between the two phases. The new suppressed MIPs, such as the one shown in Figure 5(b), were fed back into the Phase 1 classifier. In most tested cases that were initially classified as BAT-present, the classifier re-evaluated the suppressed MIPs and correctly re-classified them as BAT-absent (class 0). This provides quantitative evidence that the Phase 2 suppression pipeline effectively removes the salient features of brown fat that the classifier had learned to detect, demonstrating the coherence of the overall system.

In summary, the evaluation shows that Phase 2 effectively achieves its goal: the brown fat uptake is removed from the PET images without affecting other structures, using a liver-relative uptake criterion [8]. The method improved over a simpler approach by using modern segmentation [11] and normalisation techniques. Qualitatively, the cleaned images are easier to interpret. Quantitatively, almost all brown fat activity can be suppressed and the algorithm avoids collateral damage to real lesions or other tissues. These results support the feasibility and utility of an automated brown fat suppression tool in PET/CT.

Chapter 6

Discussion and Reflections

Developing this two-phase system provided numerous insights, both in terms of technical machine learning aspects and the nuances of medical imaging. Here, key challenges faced, the decisions made, and lessons learned along the way are reflected upon. Implications of the results and how the approach could be extended or improved are also discussed.

6.0.1 Phase 1: Classifier development reflections

A key challenge in Phase 1 was the dataset size. The training set consisted of 360 scans, with only about 10% being BAT-positive (after categorizing for binary classification based on initial 4-class labels). Transfer learning with a pre-trained ResNet-50 [9] proved essential. It was observed that training the network from random initialization would lead to overfitting the training data within a few epochs and poor performance on validation. Fine-tuning the pre-trained model, on the other hand, learned much more gradually and generalized better. This reinforced a common idea in medical imaging AI: leverage existing models as much as possible, because medical data is often scarce [16].

Using the MIP was a deliberate simplification. While a single anterior MIP was used, incorporating multiple views (e.g., frontal and lateral MIPs as a multi-channel input) was considered as an enhancement, though the single view was adequate for the great majority of cases. In theory, adding more views might improve sensitivity (some brown fat in the mediastinum might be clearer on a lateral MIP). However, we decided to keep it simple with one view, as our results indicate that this was adequate for the great majority of cases, likely because the most important brown fat (supraclavicular) is well-captured in the front view. This decision saved complexity and training time. Another point was the use of the MIP itself: a MIP can sometimes overlay unrelated structures (for instance, brown fat in front of the spine could overlap with liver uptake in the projection). But since brown fat is usually relatively small and focal, the projection did not create too confusing overlaps in our data. Potentially, the CNN could disentangle some overlaps by recognizing shape patterns. If a case had, say, very intense brown fat behind an organ, the MIP might show a merged blob, and it might be fooled, though this was not observed in practice.

Balancing recall and precision for this task has different implications than many other tasks. Usually, missing a positive (false negative) in disease detection is critical, while here, a false negative in Phase 1 just means brown fat

is not removed for the corresponding scan - the radiologist will still see it. A false positive could cause unnecessary modification of a scan. However, since the Phase 2 modification algorithm is conservative, the risk of harm is low.

A difficulty encountered was how to label the data for training. "Presence of brown fat" can be somewhat subjective, especially for mild cases. The initial 4-class labeling (none, minimal, moderate, high) was simplified to binary for the final classifier to improve performance, with "minimal" often grouped with "none" as BAT-absent. This subjectivity in assessment is one reason standardized criteria like BARCIST [8] are important in research settings.

6.0.2 Phase 2: Segmentation and removal reflections

A major lesson from Phase 2 is the benefit of combining deep learning (for liver segmentation via TotalSegmentator [11]) with rule-based domain knowledge (HU filter for fat [10], relative PET threshold [8]). Training a 3D CNN that takes PET and CT input and directly outputs a segmentation of brown fat (a fully learned approach, resembling Jorgensen, 2024's [13] Totalsegmentator fork but using both instead of only CT) was considered, however, creating a ground truth for brown fat for training would have been very labor-intensive (manually segmenting brown fat in 3D, hundreds of layers each for dozens of scans). Instead, by using TotalSegmentator [11] and a threshold rule, we sidestepped the need for manual labels. The domain knowledge (that brown fat = fat + high uptake [1] [2] [8]) was directly encoded. This turned out to be very effective and much faster to implement. It highlights that not every problem needs to be end-to-end learned; sometimes a heuristic combined with a pre-trained model can solve the problem with far less data.

The thresholds and parameters (e.g., the $1.5 \times$ liver factor, cluster size cutoff, etc.) were fine-tuned and determined empirically on a subset of the data. These are hyperparameters of the rule-based system. If this were to generalize to other settings, one might need to adjust them. For example, pediatric scans might show brown fat differently, or some scanners have different uptakes requiring a different relative multiplier. Ideally, one could incorporate a learning component to adjust these, or have them configurable.

Working with the outputs of a segmentation model highlighted the importance of proper image registration and resampling. The CT-derived liver mask was downsampled to PET resolution. A not uncommon scenario involving slight patient motion between CT and PET could cause misalignment, emphasizing the need for robust registration in a production system. This project relied on inherent PET/CT registration, perhaps leaving it vulnerable to small errors due to the misalignment.

An interesting challenge was deciding what value to set the brown fat voxels to. Setting them to the average uptake of whole-body adipose tissue was chosen to provide a natural-looking background level, rather than zeroing them out or using $0.5 \times$ liver uptake as initially prototyped.

Verifying that no actual pathology was removed was critical. The design safeguard was that if a lesion is not in fat on CT (based on HU range [10]), it won't be removed. So the only risk is a lesion that is entirely within fat and has fat-like density. Most tumors displace or change density. In rare cases, an inflamed fat (like fat necrosis) might also show uptake and be flagged - but that's arguably okay to remove if it's not what is being cared about, although radiologists might want to see inflammation. The algorithm would treat any cause of FDG uptake in fat as the same [2], which is a limitation: it doesn't know the cause of uptake, just that it's in fat. The issue isn't usually a tumor being confused for brown fat, but that small bits of tissue were being removed. This is hard to

correct if due to misalignment, however if not there are options such as harsher rules, etc.

The utility of existing tools like TotalSegmentator [11] for liver segmentation was immense. A year or two ago, without such a tool, it would have been necessary to try to train a neural network just to segment the liver (with limited data), or rely on crude thresholds. The availability of a robust multi-organ segmenter meant we could plug it in and get high-quality results immediately. This saved great time and likely improved accuracy. It's a testament to the utility of open source pretrained models in medicine [15] [16]. This project used TotalSegmentator [11] only for liver segmentation, not for fat segmentation, for which an HU-based filter [10] was preferred. Therefore, it does inherit any limitations and the pipeline's performance partly depends on TotalSegmentator's accuracy for liver. However, that model is trained on a wide variety of scans [11], so it's quite reliable and unlikely to be a bottleneck.

Evaluation of Phase 2 without extensive ground truth BAT segmentations relied on classifier feedback, visual inspection, and quantitative reduction in uptake. The system also inherently quantifies BAT, which could be a research byproduct. This project was primarily focused on the removal aspect rather than quantification, however. Also considered hypothetical scenarios, for example encountering BAT in an unusual location like pericardial fat [5]; and whether the systematic nature of the algorithm (fat [10] + high relative uptake [1] [2] [8]) would allow its detection and removal.

6.0.3 General observations and potential clinical impact

This project illustrates the interplay between AI and domain knowledge. In Phase 1, AI (CNN) learned to mimic an expert's pattern recognition for presence of brown fat. In Phase 2, explicitly encoded knowledge (fat segmentation, uptake thresholds) was used with AI as a tool within that (for segmentation). The result is a system that is partly learned, partly hand-crafted. This hybrid approach is often very practical in medical image analysis, where purely learned end-to-end solutions might require enormous labeled data or might not easily enforce certain safety constraints. By constraining our solution with known criteria (like "only remove things in fat tissue"), reliability and simplicity were improved.

If implemented in a clinical setting, such a system could function as an assistive tool. For example, after a PET/CT scan is processed, the software could generate a secondary series of PET images labeled "BAT-suppressed PET". The radiologist could then scroll that series. They might quickly appreciate that those pesky neck hotspots are gone. This could reduce distraction and reading time. It could also reduce the chance of overcalling something as pathological. Perhaps more importantly for less experienced readers, it ensures they don't accidentally misinterpret brown fat as disease. For nuclear medicine physicians, brown fat is usually a known entity, but our system could also be integrated as a preprocessing for other AI diagnostic tools, that may not otherwise be able to distinguish it from tumors; removing a major source of false alarms.

Risks, such as erroneous removal of a tumor, are mitigated by the design (only remove if CT says it's fat). Continuous vigilance or adding further safety rules, alongside keeping original images available, would be key, and a label showing that brown fat had been removed would be helpful. Hyperparameter tuning was used to find a value for the empirically tuned liver uptake multiplier (1.2-1.5x), which interestingly aligns with some BARCIST SUL criteria, providing some external validation. Technical aspects like memory management for large 3D images in large arrays and coordinate transforms to map different resolution CT to PET were also part of the development

process.

6.0.4 Limitations and potential improvements

While the system performed well on the dataset, there are limitations. The evaluation dataset for the classifier comprised 91 test cases, which is not extremely large. There was also a significant class imbalance where only around 10% were positive, and all the data was from one scanner. Testing on a larger, more varied, multi-scanner dataset and oversampling, etc. would be important to ensure generalisation.

Another limitation is that the classifier currently works on the assumption that a single MIP view suffices. If a patient had only, for example, brown fat in the posterior mediastinum, it might be hidden behind the spine on the anterior MIP. The classifier might miss it. Including lateral MIPs could catch such cases. In future, we could feed multiple projections or even directly take the 3D data. A 3D CNN could, in theory, look at the entire volume to decide presence of brown fat, but that is much more computational and data intensive, and likely overkill for simply detecting presence. So a simpler extension could be multi-view 2D input.

The removal phase relies on an HU range for fat [10] and TotalSegmentator for liver segmentation [11]. While TotalSegmentator [11] is robust, its performance on unusual anatomy or CT artifacts could potentially impact the liver reference. The Jørgensen et al. (2024) [13] model for direct BAT CT segmentation is available and was tried, but surprisingly yielded worse results for BAT identification in this project's pipeline, possibly because it relies solely on CT information, whereas this project's method leverages both PET and CT.

The current thresholding doesn't account for spatial connectivity beyond simple clustering, and not looking below the liver mask where brown fat is unlikely to be found. However, more conditions could be implemented to restrict to a more typical range. The decision to set suppressed BAT voxels to average adipose uptake is one of several possibilities and could also be modified.

One improvement idea is to implement an interactive mode: e.g. highlight detected brown fat and allow a user to approve removal. In clinical practice, maybe the radiologist wants to be sure. So, the tool could show an overlay of red areas meaning "I think this is brown fat, will remove." The user could glance and click accept. For this project's scope, full automation was fine, but that could be a safer workflow for deployment.

A specific fallback PET uptake threshold (e.g., if liver uptake was unreliable due to metastatic replacement) was considered during development but ultimately not implemented in the final version of the system being evaluated; reliance is on the liver-relative method [8].

It's worth considering that this project's approach could be extendable beyond brown fat. There are other common benign uptakes in PET that cause false positives; for example, muscle uptake, inflammation, brain uptake spill, etc. Could they similarly be suppressed? Possibly, yes, but each has its own context. Muscle uptake could be suppressed by segmenting muscle and thresholding (similar to with fat). But muscle uptake might sometimes indicate patient motion or recent exercise, which is known but not pathological; still, removing it might help. However, one must be careful not to remove muscle uptake that might indicate disease (though usually disease in muscle is rare). Inflammation is trickier because it might or might not be clinically relevant. Brown fat is a rare case where it's almost always not indicative of disease [1] [2] and thus safe to remove.

Chapter 7

Conclusion and Future Work

In conclusion, a two-phase AI system has been developed that addresses the challenge of brown fat in PET/CT imaging by first detecting its presence and then suppressing its signal in the images. The Phase 1 ResNet-50 classifier [9], after transitioning to a binary classification, demonstrated good accuracy (e.g. 86-89/91 correct cases on the test set, with an F1-score of 0.907) in identifying scans with significant brown fat. Phase 2, using a hybrid of HU-based fat masking [10], deep learning-based liver segmentation (via TotalSegmentator [11]) for PET reference, and rule-based relative PET uptake thresholding [8], effectively removed brown fat uptake from PET/CT scans, as evidenced by both qualitative visualizations and quantitative analysis. In test cases, the brown fat hotspots that could potentially mimic disease were eliminated, leading to cleaner images for interpretation. Importantly, this was achieved without altering or hiding true pathological uptake, thanks to the use of CT-derived fat masks [10] to confine the suppression only to adipose regions.

The project demonstrates the value of combining AI tools with domain knowledge: using TotalSegmentator's [11] pre-trained model for precise liver segmentation significantly improved the robustness of the adaptive thresholding [8]. Normalizing PET uptake thresholds to each patient's liver uptake accounted for individual variations and scanner differences [8]. The outcomes indicate this approach can serve as a practical aid in reading PET/CT scans.

This work has some novel aspects. To the best of current knowledge, while prior studies have focused on segmenting or measuring brown fat [12] [13] [14], this is one of the first implementations aimed at actively suppressing brown fat in the images to improve diagnostic clarity. It opens up a discussion on how far image post-processing should go in modifying images: here it was done responsibly to remove a known artifact, and care was taken to ensure that such modifications do not risk removing important information. The success of this approach for brown fat could inspire similar solutions for other benign uptake issues in PET, as mentioned earlier (e.g., muscle uptake suppression with appropriate segmentation of muscle, etc.).

For future improvements, several avenues can be explored:

- **Data expansion and validation:** It is planned to test the system on larger datasets, including multi-center data, to validate its generalisability. This would include cases with extreme conditions, different PET radio-tracers such as PSMA (e.g., if a different tracer is used, brown fat might not be an issue, but interestingly brown fat also takes up some other tracers like certain fatty acid analogs [1] - this system is specifically

tuned to FDG right now). Cases from pediatric patients where brown fat is more common may also be included, ensuring the thresholds and segmentation still perform well.

- **Enhanced classifier features:** The Phase 1 classifier could be enhanced by incorporating multi-view inputs or even the CT information. For example, using a dual-input network that takes both the PET MIP and perhaps a projection of the CT (to identify fat content) might improve accuracy or confidence. However, this added complexity would need to be justified by significantly better performance.
- **Direct BAT segmentation via deep learning:** With the labeled data our pipeline can produce, we could train a dedicated brown fat segmentation model. For instance, a 3D U-Net [17] that inputs the CT and PET and outputs a brown fat mask (like the one that already exists, but also with PET data). This model could potentially learn to fine-tune the regions better than a fixed threshold. It might also learn to include border-line uptake that the threshold might miss, by seeing examples. In essence, the current method could provide pseudo-ground-truth masks to train such a model in a weakly supervised manner. This could eventually replace the threshold step with a learned component that might adapt better to edge cases.
- **Runtime and integration:** Currently, the bottleneck in runtime is the TotalSegmentator segmentation [11] (1 minutes per scan on GPU). In a busy clinical setting, this is not too bad (as scans are not read instantaneously), but faster is always better. Integration-wise, making the tool user-friendly (maybe a one-click in the PACS viewer to toggle brown fat removal) would help adoption. We would likely want to overlay the removed areas in a subtle way as mentioned, or have a quick way to verify what was removed.
- **Extended applications:** As an extension, exploring suppression of other types of physiological uptake with similar pipelines could be valuable. For example, brain uptake is a problem only if one is searching for head/neck lesions; some approaches like template subtraction exist. Muscle uptake from exertion could possibly be suppressed by segmenting muscles (TotalSegmentator [11] also segments muscle) and applying thresholds. However, muscle uptake can indicate things like recent activity or sometimes disease (myositis) so one must be careful. Nevertheless, an optional tool to reduce muscle uptake might help in reading as well. Another application is in PET/MR, where MR can be used to segment fat vs brown fat (via fat fraction). Some research already uses MR to identify BAT, but if one had that, it could feed into our pipeline as well.
- **Improving brown fat quantification:** While this project's focus was removal, the system inherently quantifies brown fat. We could compile the results across patients to study correlations. This could be a side research result: using AI to automatically measure brown fat in large cohorts and relate to clinical factors [7]. By refining the quantification (for example, computing total BAT SUV uptake, etc. [8]), data could be contributed to metabolic research.
- **User feedback loop:** In a deployed system, incorporating a feedback loop would be beneficial. For instance, if a radiologist notices the algorithm removed something it shouldn't, they could correct it (e.g. mark that region as not brown fat). Such feedback could be logged to improve the model or thresholds. Conversely, if it missed some brown fat, they could flag it. Over time, this could enhance the system's accuracy through continuous learning.

In closing, this project achieved its primary goals and demonstrated a viable solution to a practical imaging problem using AI. The interdisciplinary approach of mixing radiological insight [2] [4] [5] with deep learning [15] [16] was key to the success. The work not only provides a tool that could, with additional work, be used in clinical imaging workflows, but also lays the groundwork for more intelligent image post-processing techniques. As AI in medical imaging continues to evolve, such targeted applications are expected to become increasingly common, augmenting the capabilities of radiologists and improving diagnostic accuracy.

Bibliography

- [1] B. Cannon and J. Nedergaard, “Brown Adipose Tissue: Function and Physiological Significance,” *Physiol. Rev.*, vol. 84, no. 1, pp. 277–359, 2004. Available: <https://doi.org/10.1152/physrev.00015.2003>
- [2] H.W. Yeung, R.K. Grewal, M. Gonen, H. Schoder, and S.M. Larson, “Patterns of ^{18}F -FDG uptake in adipose tissue and muscle: a potential source of false-positives in PET,” *J. Nucl. Med.*, vol.44, no.11, pp. 1789–1796, 2003. Available: <https://jnm.snmjournals.org/content/44/11/1789>
- [3] J.D. Steinberg, W. Vogel, and E. Vegt, “Factors influencing brown fat activation in FDG PET/CT: a retrospective analysis of 15,000+ cases,” *Br. J. Radiol.*, vol.90, no.1075, 20170093, 2017. Available: <https://doi.org/10.1259/bjr.20170093>
- [4] M.T. Truong, J.J. Erasmus, R.F. Munden, E.M. Marom, B.S. Sabloff, G.W. Gladish, D.A. Podoloff, and H.A. Macapinlac, “Focal FDG uptake in mediastinal brown fat mimicking malignancy: a potential pitfall resolved on PET/CT,” *AJR Am. J. Roentgenol.*, vol. 183, no.4, pp. 1127–1132, 2004. Available: <https://doi.org/10.2214/ajr.183.4.1831127>
- [5] H.L. Park, W.W. Lee, S. Bae, I.R. Yoo, W.H. Choi, and S.E. Kim, “Brown Fat: Atypical Locations and Appearances Encountered in PET/CT,” *AJR Am. J. Roentgenol.*, vol.193, no.1, pp. 239–245, 2009. Available: <https://doi.org/10.2214/AJR.08.2081>
- [6] V. Söderlund, S.A. Larsson, and H. Jacobsson, “Reduction of FDG uptake in brown adipose tissue in clinical patients by a single dose of propranolol,” *Eur. J. Nucl. Med. Mol. Imaging*, vol.34, no.7, pp. 1018–1022, 2007. Available: <https://doi.org/10.1007/s00259-006-0318-9>
- [7] K.L. Townsend and Y.-H. Tseng, “Brown adipose tissue: recent advances and therapeutic implications,” *Adipocyte*, vol.1, no.1, pp. 13–22, 2012. Available: <https://doi.org/10.4161/adip.18951>
- [8] K.Y. Chen, A.M. Cypess, M.R. Laughlin, C.R. Haft, H.H. Hu, M.A. Bredella, S. Enerbäck, P.E. Kinahan, W. van Marken Lichtenbelt, F.I. Lin, J.J. Sunderland, K.A. Virtanen, and R.L. Wahl, “Brown Adipose Reporting Criteria in Imaging STudies (BARCIST 1.0): Recommendations for Standardized FDG-PET/CT Experiments in Humans,” *Cell Metab.*, vol.24, no.2, pp. 210–222, 2016. Available: <https://doi.org/10.1016/j.cmet.2016.07.014>

- [9] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, 2016, pp. 770–778. Available: <https://doi.org/10.1109/CVPR.2016.90>
- [10] W.H. Kim, C.G. Kim, and D.-W. Kim, “Optimal CT Number Range for Adipose Tissue When Determining Lean Body Mass in Whole-Body F-18 FDG PET/CT Studies,” *Nucl. Med. Mol. Imaging*, vol.46, no.4, pp. 294–299, 2012. Available: <https://doi.org/10.1007/s13139-012-0175-3>
- [11] J. Wasserthal, H.-C. Breit, M.T. Meyer, M. Pradella, D. Hinck, A.W. Sauter, T. Heye, D.T. Boll, J. Cyriac, S. Yang, *et al.*, “TotalSegmentator: robust segmentation of 104 anatomical structures in CT images,” *Radiol. Artif. Intell.*, vol.5, no.5, e230024, 2023. Available: <https://doi.org/10.1148/ryai.230024>
- [12] S. Hussein, A. Green, A. Watane, D. Reiter, X. Chen, G.Z. Papadakis, B. Wood, A. Cypess, M. Osman, and U. Bagci, “Automatic Segmentation and Quantification of White and Brown Adipose Tissues from PET/CT Scans,” *IEEE Trans. Med. Imaging*, vol.36, no.3, pp. 734–744, 2017. Available: <https://doi.org/10.1109/TMI.2016.2636188>
- [13] K. Jorgensen, F.E. Hoi-Hansen, R.J.F. Loos, C. Hinge, and F.L. Andersen, “Automated supraclavicular brown adipose tissue segmentation in computed tomography using nnU-Net: integration with TotalSegmentator,” *Diagnostics*, vol.14, no.24, Art. no. 2786, 2024. Available: <https://doi.org/10.3390/diagnostics14242786>
- [14] E. Erdil, A.S. Becker, M. Schwyzer, B. Martinez-Tellez, J.R. Ruiz, T. Sartoretti, H.A. Vargas, A.I. Burger, A. Chirindel, D. Wild, N. Zamboni, B. Deplancke, V. Gardeux, C.I. Maushart, M.J. Betz, C. Wolfrum, and E. Konukoglu, “Predicting standardized uptake value of brown adipose tissue from CT scans using convolutional neural networks,” *Nat. Commun.*, vol.15, no.1, Art. no. 8402, 2024. Available: <https://doi.org/10.1038/s41467-024-52622-w>
- [15] G. Litjens, T. Kooi, B.E. Bejnordi, A.A. Setio, F. Ciompi, M. Ghafoorian, J.A. van der Laak, B. van Ginneken, and C.I. Sanchez, “A survey on deep learning in medical image analysis,” *Med. Image Anal.*, vol.42, pp. 60–88, 2017. Available: <https://doi.org/10.1016/j.media.2017.07.005>
- [16] H.-C. Shin, H.R. Roth, M. Gao, L. Lu, Z. Xu, I. Nogues, J.Y. Choy, and R.M. Summers, “Deep convolutional neural networks for computer-aided detection: CNN architectures, dataset characteristics and transfer learning,” *IEEE Trans. Med. Imaging*, vol.35, no.5, pp. 1285–1298, 2016. Available: <https://doi.org/10.1109/TMI.2016.2528162>
- [17] O. Ronneberger, P. Fischer, and T. Brox, “U-Net: Convolutional networks for biomedical image segmentation,” in *Proc. Int. Conf. Med. Image Comput. Comput. Assist. Interv. (MICCAI)*, 2015, pp. 234–241. Available: https://doi.org/10.1007/978-3-319-24574-4_28 (arXiv: <https://arxiv.org/abs/1505.04597>)
- [18] F. Isensee, P.F. Jager, S.A.A. Kohl, J. Petersen, and K.H. Maier-Hein, “nnU-Net: a self-configuring method for deep learning-based biomedical image segmentation,” *Nat. Methods*, vol.18, no.2, pp. 203–211, 2021. Available: <https://doi.org/10.1038/s41592-020-01008-z>

- [19] D.P. Kingma and J. Ba, “Adam: A Method for Stochastic Optimization,” *arXiv preprint arXiv:1412.6980*, 2014. Available: <https://arxiv.org/abs/1412.6980>
- [20] M. Soret, S.L. Bacharach, and I. Buvat, “Partial-volume effect in PET tumor imaging,” *J. Nucl. Med.*, vol.48, no.6, pp. 932–945, 2007. Available: <https://jnm.snmjournals.org/content/48/6/932>

Chapter 8

Appendices

.1 Appendix 1: Project plan

Name: Abdurrahmaan Ali

Supervisor's name: Dr. Ghita Kouadri Mostefaoui (not confirmed)

Project title: AI/Deep Learning for Brown Fat Detection and Removal in PET-CT Scans

Aims and Objectives

Aims:

First phase: Develop a deep learning model to automatically detect brown adipose tissue (brown fat) uptake in PET-CT scans using 2D images. This phase aims to formulate the problem as a classification task (e.g. present/absent or a 0-3 scale for level of brown fat activity) and demonstrate feasibility/proof of concept with a reasonably accurate convolutional neural network (CNN).

Second phase: Extend the approach to 3D PET-CT data to create a method that not only identifies brown fat but also removes or suppresses its signal in the scans. The goal is to generate brown fat-free PET-CT images, improving image clarity for diagnostic purposes. This is the harder phase and the bulk of the work.

Objectives:

1. Review the literature on brown fat in medical imaging and deep learning techniques (CNNs, transfer learning, 3D image networks) relevant to detection and segmentation in PET-CT scans.
2. Prepare and understand the dataset of 458 annotated 2D PET-CT scan images, and perform any preprocessing/augmentation needed for training the model.
3. Design and implement a CNN classifier (potentially using transfer learning/LoRa's since small dataset) to categorize PET-CT images by brown fat content (could either classify images as present vs. absent, or by grade 0-3). Keep the architecture and training approach flexible
4. Acquire or organize the 3D PET-CT scan data for Phase 2. Since annotations may not be available try unsupervised methods to handle brown fat in 3D.
5. Develop a brown fat removal method for 3D scans. May involve building a model (such as a 3D CNN or U-Net segmentation network) that can identify brown fat regions in the PET volume and then remove or replace them. Explore and implement the most feasible approach. Refine and develop the model in stages.
6. Evaluate the outcomes of both phases thoroughly. For Phase 1, evaluate classification performance against the annotated ground truth. For Phase 2, evaluate the brown fat removal by comparing the original and modified images and potentially by asking an expert or using quantitative measures (like reduction in standardized uptake values in brown fat regions); depends on data set. Document potential improvements/issues.

Deliverables

By the end of the project, the following deliverables are expected:

- ❖ **Literature Review:** This will summarize key findings from literature on PET-CT brown fat imaging and deep learning methods (especially CNN-based classification and segmentation in medical imaging) that inform the project's approach.

- ❖ **Trained Phase 1 Model:** A working CNN model (with saved weights and architecture details) for 2D image classification of brown fat presence/level. This includes the training code and a description of the model's performance (e.g. achieved accuracy or other metrics on the annotated dataset).
- ❖ **Phase 2 Brown Fat Removal System:** A prototype solution for processing 3D PET-CT scans to remove brown fat, for example in the form of a trained 3D neural network or a pipeline that identifies brown fat regions and outputs a modified scan with those regions suppressed. Should include the code for this system, trained model weights, and example output images demonstrating brown fat removal.
- ❖ **Documented Software/User Guides:** All code developed in the project will be organised and commented. There will be a README explaining how to run the models (Phase 1 and Phase 2), and how to reproduce the results, to ensure the project can be understood and used by others in the future.
- ❖ **Evaluation Strategy and Results:** A documented evaluation plan outlining how each phase's results were measured. The actual results (tables of metrics, example images before/after removal) will be provided to demonstrate the project's outcomes.

Work Plan

The project will be executed in a structured manner across two main phases:

- ❖ **Initial Background Study:** Begin with researching relevant background material. Read about brown adipose tissue in PET-CT imaging (to understand why brown fat appears on scans and how it is currently identified or mitigated) and study prior applications of deep learning in medical image classification/segmentation. This stage also includes clarifying project requirements and formulating a detailed approach for Phase 1; but will continue throughout the project as needed.

Phase 1:

- ❖ **Data Analysis and 2D Model Development:** Using the obtained dataset of 458 annotated 2D PET-CT images, perform data exploration and preprocessing. Decide on the classification scheme (binary vs. 4-class) based on data distribution. Then design the CNN model architecture for brown fat detection. Will need to set up the training environment, perhaps using transfer learning or LoRAs (e.g. using a pre-trained CNN like ResNet or VGG as a starting point) due to the small dataset size, and defining the output (sigmoid for binary or softmax for multi-class).
- ❖ **Training and Evaluation:** Train the CNN with the annotated images. Monitor training performance, adjust hyperparameters to avoid overfitting and improve accuracy. Validate the model on held-out data to measure how well it generalises. If performance is not satisfactory, go back to adjust the model, trying different architectures or more data augmentation. Once a reasonably capable model is achieved, record the results and insights. Ideally, conclude Phase 1 by verifying that deep learning can indeed detect brown fat uptake.

Phase 2:

- ❖ **3D Data Preparation and Approach Design:** Move on to the 3D PET-CT scan data. Ensure the 3D scans are anonymized and suitable for use. Since they might lack annotations for brown fat regions, plan how to generate training targets or otherwise develop the removal method. Could utilise the Phase 1 classifier on 2D slices of the 3D volume to roughly locate slices with brown fat, or to use known characteristics (brown fat often shows certain patterns on PET and corresponding CT regions are fat tissue) to heuristically label areas. Develop a strategy for the removal task - for example, decide whether to frame it as a segmentation problem

(identify voxels of brown fat) or a direct image-to-image translation problem. At this stage, also consider network architecture and the practicality of training it with the data available.

- ❖ **Iterative Model Development (Brown Fat Removal):** Implement the chosen approach in an iterative fashion. For instance, start with a simpler solution: use the classifier from phase 1 to identify slices with brown fat and manually remove or mask those regions (this provides a baseline for comparison). Next, attempt a learning-based solution: train a model to predict either a mask of brown fat or an image with those areas suppressed. Could generate synthetic training pairs (original vs. “brown fat removed” images) if ground truth cleaned images are not directly available. Iterate through cycles of development, training, testing, and refining and being flexible about solutions to try to achieve acceptable results.
- ❖ **Testing and Integration:** After developing the models, perform end-to-end testing. For example, take a full 3D scan that has brown fat, run it through the system (which might first detect if significant brown fat is present using the classifier, then apply the removal model) and inspect the results. Verify that the brown fat-free image still aligns with the original CT (other structures not corrupted). If possible, compare the PET scan before and after removal. Address remaining issues or final tweaks can be added here.
- ❖ **Reporting and Documentation:** Throughout the project, maintain clear documentation. The key milestones include preparing an interim report (after first phase complete) that details the progress (literature findings, methodology and results of the 2D CNN, and the final plan for Phase 2). As the project nears completion, need to dedicate time for final thesis writing, which would involve documenting the methodology of Phase 2, compiling results (with figures such as example images and performance graphs), and writing analysis/discussion of what the results mean.

Ethics review

I believe no formal ethics review is required for this project. The data used (PET-CT scans) are anonymized medical images, and are being reused for technical research purposes only. The project does not involve intervention in clinical care, and no new data is being collected. As such, it should pose minimal risk in terms of privacy/ethical considerations.

.2 Appendix 2: Interim report

Interim Project Report

Name: Abdurrahmaan Ali

Project Title (from Plan): AI/Deep Learning for Brown Fat Detection and Removal in PET-CT Scans

Current Project Title: AI/Deep Learning for Brown Fat Detection and Removal in PET/CT Scans

Supervisor: Dr. Ghita Kouadri Mostefaoui

External Supervisor: N/A

Progress Made to Date:

Work on the project has progressed, with the first phase now complete. This initial phase focused on developing a deep learning model to automatically detect brown adipose tissue (BAT) uptake using 2D Maximum Intensity Projection (MIP) images from PET-CT scans. The goal was to establish the feasibility of using a CNN for this classification task.

A literature review was performed at the start, looking into brown fat's appearance in medical imaging and exploring deep learning techniques suited for this problem, like Convolutional Neural Networks (CNNs), transfer learning approaches for smaller datasets, and methods used for 3D image analysis in PET-CT.

Following the literature review and preparation of the initial 2D dataset, a CNN classifier was successfully developed and trained. The model achieves good results in classifying the 2D MIP images based on BAT uptake, showing promising performance in terms of accuracy, F1 score, and other relevant metrics [.907 f1-score]. The model performed better when trained for binary classification (presence/absence of significant BAT) compared to a 4-class approach (grading levels 0-3). This is likely because lower levels of brown fat activity (i.e. grade 1) are harder to distinguish reliably. Based on this, considering focusing the classification aspect on a binary determination going forward.

Remaining Work to be Done:

With Phase 1 concluded, the project is now moving into Phase 2, which tackles the more challenging goal of extending the approach to full 3D PET-CT volumes and developing a method to remove or suppress the BAT signal. The aim here is to produce PET-CT images that are essentially "brown fat-free," which could help improve the clarity of scans for other diagnostic purposes.

To support this phase, a dataset of approximately 30 full 3D PET/CT volumes has been acquired. These volumes correspond to a subset of the cases from which the original 2D MIP images were derived. A key challenge is that these 3D volumes are not annotated for BAT regions, and the dataset size is likely too small to train a complex 3D segmentation model from scratch.

Therefore, two main strategies are being considered for developing the brown fat removal method:

1. Existing Segmentation Tools: Use a specialized segmentation tool, specifically the TotalSegmentator BAT fork, which is designed to identify various tissues including brown fat on CT scans. The idea is to use this tool to segment the BAT regions on the CT component of the scans. Once identified, the corresponding signal in the PET volume within these regions could be set to match the signal of background fat tissue, effectively removing the high uptake signature of BAT.
2. Rule-Based Classical Approach: Develop a more traditional image processing system. This would involve defining rules based on the known characteristics of BAT in PET/CT. For example, we could identify regions that show fat density on the CT scan but exhibit unusually high radiotracer uptake on the corresponding PET scan. These identified voxels, presumed to be BAT, would then have their PET signal adjusted downwards to typical background fat levels.

The next steps involve implementing and testing these two approaches. Development will likely be iterative, potentially starting with simpler versions or applying methods to subsets of the data before scaling up.

Evaluation of the Phase 2 outcome will focus on how effectively the brown fat signal is suppressed without negatively impacting the rest of the image. This will involve inspection of the original versus the processed scans and potentially quantitative analysis (like measuring the reduction in SUV in targeted areas). Expert radiological review would also be sought for qualitative assessment.

The final deliverables will include the developed brown fat removal system (code/potentially trained models), along with thorough documentation explaining how to use it, and a detailed evaluation of its performance.

.3 Appendix 3: User guide

For detailed setup, refer to README.md on the repository

User guide: 2D classifier

Setup and Dependencies:

- **Required Libraries:** The software relies on the following Python libraries: torch, torchvision, pydicom, imageio, numpy, and scikit-learn.
- **Installation:** These dependencies can be installed by running the command: `!pip install torch torchvision pydicom imageio numpy scikit-learn` within the notebook environment.

3. Data Input and Configuration

- **Input Data:** The classifier processes 2D PET/CT DICOM files.
- **Label Files (CSV format):**
 - A CSV file is required for labels and must include "filename" and "brown_fat" columns.
 - The "filename" should be relative to the dcm_root (DICOM root directory).
 - "brown_fat" is an integer score from 0 to 3, indicating the level of brown fat uptake.
- **Configuration Parameters (Args class):** Key operational parameters are defined within the Args class in the notebook. Users may need to adjust these paths and settings:
 - `data_csv`: Path to the CSV file containing labels for training data.
 - `test_csv`: Path to the CSV file containing labels for test data.
 - `dcm_root`: The root directory where the DICOM image files are stored.
 - `out_dir`: The directory where outputs from model runs (like saved models) will be stored.
 - `model`: Path to a pre-trained model file (for inference or testing) or the location where the best-performing trained model will be saved.
 - `img_size`: The image size to which DICOM images will be resized (e.g., 224x224 pixels).

4. Running the Software

Training a New Model: To train a new model, users can uncomment and execute the `train(arguments)` cell in the notebook.

However, A pre-trained model is already provided, so retraining which takes time may not be necessary for initial use.

Evaluating on a Test Set:

The `test_set(arguments)` function can be used to run the trained model on a test dataset and evaluate its performance.

Inference on a Single DICOM File:

To predict the brown fat score for an individual DICOM file, use the `infer_single(model_path, dicom_file, img_size=224)` function.

Ensure the correct directories and folder structure is used

User guide: 3D tool

- Required Libraries:** SimpleITK, pydicom, TotalSegmentator, scipy, numpy, glob, os, datetime.

- Installation:** Install necessary packages using the command: `!pip install SimpleITK pydicom TotalSegmentator scipy`.

- Google Drive:** The notebook is set up to mount Google Drive (`drive.mount('/content/drive')`) for accessing datasets, so should ensure their data is accessible via Drive or adjust paths accordingly. Ensure correct path structure for OS.

Core Functionality

1. Liver and Fat Segmentation (`segment_liver_ct` function):

BAT Suppression (`suppress_bat` function):

Input Requirements

- CT DICOM Series:** A directory (`ct_dir`) containing the CT scan slices in DICOM format.

- PET DICOM Series:** A directory (`pet_dir`) containing the corresponding PET scan slices in DICOM format.

Running the Software: Example Workflow

2. Set Paths: Define the following paths and parameters in the notebook:

- `ct_dir`: e.g., `/content/drive/MyDrive/dataset/3D/xxx/CT`
- `pet_dir`: e.g., `/content/drive/MyDrive/dataset/3D/xxx/PET`
- `seg_output_dir`: Directory for TotalSegmentator outputs (e.g., `/content/segm_out/`)
- `output_dir`: Directory to save the modified PET DICOM files (e.g., `/content/out`)
- `liver_mult`: Multiplier for BAT threshold calculation (e.g., 1.2)
- `min_3d_cluster_voxels`: Minimum size for BAT clusters (e.g., 10)

3. Output

- Primary Output:** A new series of DICOM PET files with BAT activity suppressed. These are saved in the specified `output_dir`.

- Intermediate Files:** TotalSegmentator creates segmentation files (e.g., `liver_segments.nii`) in the `seg_output_dir`.

- Console Logs:** The script prints progress updates, calculated thresholds, number of voxels/slices processed, and any warnings or errors encountered.

.4 Appendix 4: Code

All code and data can be found on the github:

<https://github.com/abduali02/c0138>

Listing 1: 2D Brown fat classifier - Phase 1

45

```
1 !pip install torch torchvision pydicom imageio numpy scikit-learn
2
3 """# Brown Fat Uptake Classifier (PET/CT)"""
4
5 #!/usr/bin/env python3
6 """
7 Brown Fat Classifier (PET/CT DICOM)
8
9 trained a ResNet50 on ~500 PET/CT DICOM slices to predict:
10     Brown fat uptake score (0 / 1 / 2 / 3 )
11
12 DATA:
13 CSV must contain columns:
14     filename,brown_fat
15 Where filename is relative to dcm_root, brown_fat {0,1,2,3},
16
17 Dependencies:
18     pip install torch torchvision pydicom imageio numpy scikit-learn
19 """
20
21 import os
22 import csv
23 import argparse
```



```
24 from pathlib import Path
25 from typing import Optional
26 from sklearn.metrics import confusion_matrix, classification_report
27
28 import numpy as np
29 import pydicom
30 import torch
31 import torch.nn as nn
32 import torch.optim as optim
33 from torch.utils.data import Dataset, DataLoader, random_split
34 from torchvision import transforms, models
35 from IPython.display import Image
36 #from google.colab import drive
37
38 root_dir="content" #
39
40 #drive.mount('/content/drive')
41 #root_dir = "/content/drive/MyDrive/"
42
43 Image(os.path.join(root_dir, "dataset/bfexample.png"))
44
45 """Brown fat is a harmless metabolically active fat that shows up more in colder weather. Uptake can be seen in the neck, lymph
    nodes, and upper spine on the right image (outlined in red). This can interfere with reading a scan as it can obfuscate actual
    malignancies."""
46
47 class Args:
48     data_csv = os.path.join(root_dir, "dataset/labels.csv")
```

```

49     test_csv = os.path.join(root_dir, "dataset/labels_test.csv")
50     dcm_root = os.path.join(root_dir, "dataset/dicoms")
51     epochs = 30
52     batch_size = 8
53     out_dir = os.path.join(root_dir, "runs/exp1/")
54     model = os.path.join(root_dir, "runs/exp1/best.pt")
55     img_size = 224
56     lr = 3e-4
57     weight_decay = 1e-4
58     num_workers = 0
59     freeze_backbone = 5
60
61 arguments = Args()
62
63 """## Dataset
64
65 Load 2D PET/CT DICOM files, assign labels, and expose them via a custom `torch.utils.data.Dataset` subclass ready for the
66     DataLoader
67 """
68
69 class DicomDataset(Dataset):
70     """Load DICOM files and associated labels"""
71
72     def __init__(self, csv_file: str, dcm_root: str, transform=None):
73         self.items = [] # list[(Path, int, ] (file, brown_fat,)
74         with open(csv_file) as f:
75             reader = csv.DictReader(f)

```

```

75         for row in reader:
76             path = Path(dcm_root) / (row["filename"] + ".dcm")
77             bf = int(row["brown_fat"])
78             self.items.append((path, bf,))
79     self.transform = transform
80
81     # ----- helpers -----
82     def _load_dicom(self, fname: Path) -> np.ndarray:
83         ds = pydicom.dcmread(str(fname))
84         arr = ds.pixel_array.astype(np.float32).squeeze()
85         #rescale -> 0 255 (8 bit )
86         arr -= arr.min()
87         arr /= (arr.max() + 1e-5)
88         arr *= 255.0
89         return arr.clip(0, 255).astype(np.uint8)
90
91     # ----- torch.Dataset interface -----
92     def __len__(self):
93         return len(self.items)
94
95     def __getitem__(self, idx):
96         path, bf = self.items[idx]
97
98         img = self._load_dicom(path)
99
100         # if single-channel, stack into RGB; otherwise leave
101         if img.ndim == 2:

```

```

102         img = np.stack([img, img, img], axis=-1)
103     elif not (img.ndim == 3 and img.shape[2] == 3):
104         raise ValueError(f"Unexpected DICOM shape {img.shape}")
105
106     if self.transform:
107         img = self.transform(img)
108         target_bf = torch.tensor(bf, dtype=torch.long)
109         return img, {"bf": target_bf}
110
111     """## Transforms & Augmentation
112
113     Define training & validation transform pipelines. These include resizing to the network input size, PET/CT standardisation, and
114     data augmentation techniques such as random rotation, horizontal flip, and normalisation.
115
116     """
117
118     def build_transforms(img_size: int = 224, train: bool = True):
119         ops = [
120             transforms.ToPILImage(),
121             transforms.Resize((img_size, img_size)),
122         ]
123
124         if train:
125             ops += [transforms.RandomHorizontalFlip(), transforms.RandomRotation(10)]
126
127         ops += [
128             transforms.ToTensor(),
129             transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225]),
130         ]
131
132         return transforms.Compose(ops)

```

```

128
129 """## Model: A ResNet50 backbone
130
131 Implementation of the neural network architecture: a ResNet50 encoder pretrained on ImageNet
132 """
133
134 class ResNetBF(nn.Module):
135     def __init__(self, num_bf: int = 4):
136         super().__init__()
137         self.backbone = models.resnet50(weights=models.ResNet50_Weights.IMAGENET1K_V2)
138         in_features = self.backbone.fc.in_features
139         self.backbone.fc = nn.Identity() # remove singletask head
140         self.dropout = nn.Dropout(0.5)
141         self.fc_bf = nn.Linear(in_features, num_bf)
142
143     def forward(self, x):
144         feat = self.backbone(x)
145         feat = self.dropout(feat)
146         return self.fc_bf(feat)
147
148 """## Training / Evaluation routines
149
150 Set up the training loop (optimizer, scheduler, loss functions) and evaluation helpers for metrics like AUC and F1 score.
151 Includes checkpointing.
152 """
153 def train(args):

```

```
154 device = "cuda" if torch.cuda.is_available() else "cpu"
155 transform = build_transforms(args.img_size, train=True)
156 dataset = DicomDataset(args.data_csv, args.dcm_root, transform)
157 val_size = int(len(dataset) * 0.2)
158 train_size = len(dataset) - val_size
159 train_ds, val_ds = random_split(dataset, [train_size, val_size])
160 train_loader = DataLoader(train_ds, batch_size=args.batch_size, shuffle=True, num_workers=args.num_workers)
161 val_loader = DataLoader(val_ds, batch_size=args.batch_size, shuffle=False, num_workers=args.num_workers)
162
163 model = ResNetBF().to(device)
164 crit_bf = nn.CrossEntropyLoss()
165
166 optimizer = optim.AdamW(model.parameters(), lr=args.lr, weight_decay=args.weight_decay)
167 scheduler = optim.lr_scheduler.CosineAnnealingLR(optimizer, T_max=args.epochs)
168
169 freeze_epochs = max(args.freeze_backbone, 0)
170 if freeze_epochs:
171     for p in model.backbone.parameters():
172         p.requires_grad = False
173
174 best_val = float('inf')
175 os.makedirs(args.out_dir, exist_ok=True)
176
177 for epoch in range(1, args.epochs + 1):
178     if epoch == freeze_epochs + 1:
179         for p in model.backbone.parameters():
180             p.requires_grad = True
```

```
181     print("Backbone unfrozen from epoch", epoch)
182
183 model.train()
184 running_loss = 0.0
185 for img, targets in train_loader:
186     img = img.to(device); bf_gt = targets["bf"].to(device)
187     bf_logits= model(img)
188     loss = crit_bf(bf_logits, bf_gt)
189     optimizer.zero_grad(); loss.backward(); optimizer.step()
190     running_loss += loss.item() * img.size(0)
191
192 train_loss = running_loss / len(train_loader.dataset)
193 val_loss = evaluate(model, val_loader, crit_bf, device)
194 print(f"Epoch {epoch:02d}/{args.epochs} | train {train_loss:.4f} | val {val_loss:.4f}")
195 scheduler.step()
196
197 tries=6
198 if val_loss < best_val:
199     best_val = val_loss
200     torch.save(model.state_dict(), Path(args.out_dir) / "best.pt")
201     trigger= 0 # reset patience counter
202 else:
203     trigger += 1
204     if trigger >= tries:
205         print(f"No improvement for {trigger} epoch(s)")
206         print("Early stopping triggered.")
207         break
```

```

208
209     print("Training complete. Best val loss:", best_val)
210
211 @torch.inference_mode()
212 def evaluate(model, loader, crit_bf, device):
213     model.eval(); running_loss = 0.0
214     for img, targets in loader:
215         img = img.to(device); bf_gt = targets["bf"].to(device)
216         bf_logits = model(img)
217         running_loss += (crit_bf(bf_logits, bf_gt)).item() * img.size(0)
218     return running_loss / len(loader.dataset)
219
220 """## Inference & Test set functions
221
222 Utility functions to run inference on a single scan or an entire test directory
223 """
224
225 @torch.inference_mode()
226 def infer_single(model_path: str, dcm_file: str, img_size: int = 224):
227     device = "cuda" if torch.cuda.is_available() else "cpu"
228     model = ResNetBF(); model.load_state_dict(torch.load(model_path, map_location=device))
229     model.to(device).eval()
230     transform = build_transforms(img_size, train=False)
231     ds = pydicom.dcmread(dcm_file); img = ds.pixel_array.astype(np.float32)
232     img = (img - img.min()) / (img.max() + 1e-5) * 255; img = img.clip(0, 255).astype(np.uint8)
233     img = np.stack([img, img, img], axis=-1)
234     img = transform(img).unsqueeze(0).to(device)

```



```
235     bf_logits= model(img)
236     bf_pred = torch.argmax(bf_logits, 1).item()
237     return bf_pred
238
239 def test_set(args):
240     device = "cuda" if torch.cuda.is_available() else "cpu"
241     transform = build_transforms(args.img_size, train=False)
242     dataset = DicomDataset(args.test_csv, args.dcm_root, transform)
243     loader = DataLoader(dataset, batch_size=args.batch_size, shuffle=False, num_workers=args.num_workers)
244     model = ResNetBF(); model.load_state_dict(torch.load(args.model, map_location=device))
245     model.to(device).eval()
246     crit_bf = nn.CrossEntropyLoss()
247     test_loss = evaluate(model, loader, crit_bf, device)
248     print(f"Test loss: {test_loss:.4f}")
249
250     total = 0
251     bf_bin_correct = 0
252     bf_4class_correct = 0
253     path_correct = 0
254
255     # helper for binary grouping
256     def bf_bin(x): return 0 if x in (0,1) else 1
257
258     with torch.inference_mode():
259         for imgs, targets in loader:
260             imgs = imgs.to(device)
261             bf_true = targets["bf"]
```

```
262
263     logits_bf = model(imgs)
264     preds_bf = torch.argmax(logits_bf, 1)
265
266
267     for t, p in zip(bf_true, preds_bf):
268         total += 1
269         # binary
270         if bf_bin(t) == bf_bin(p):
271             bf_bin_correct += 1
272         # 4-class
273         if t == p:
274             bf_4class_correct += 1
275
276
277     print(f"Brown fat binary: {bf_bin_correct}/{total}")
278     print(f"Brown fat 4-class: {bf_4class_correct}/{total}")
279
280
281     all_preds = []
282     all_targets = []
283     for imgs, targets in loader:
284         imgs = imgs.to(device)
285         bf_true = targets["bf"] # extract brown-fat labels only
286         logits = model(imgs)
287         preds = torch.argmax(logits, dim=1)
288
289         all_preds.extend(preds.cpu().numpy())
```

```

289     all_targets.extend(bf_true.cpu().numpy())
290
291     # Confusion matrix & metrics
292     cm = confusion_matrix(all_targets, all_preds, labels=[0, 1, 2, 3])
293     print("\nConfusion Matrix (4-class):")
294     print(cm)
295
296     print("\nClassification Report:")
297     print(classification_report(all_targets, all_preds, labels=[0, 1, 2, 3], digits=3))
298
299     """## Run
300
301     Training takes approximately 10-30min on CPU and 1-5min on GPU. A trained model is already provided.
302     """
303
304     #train(arguments)
305     # ^ uncomment to train new model
306
307     """A loss of  $<0.6$  is considered good. Random chance is  $\sim 1.3$ .
308     Binary considers only the presence of absence of brown fat uptake. 4-class considers four levels (no,low,med,high / 0,1,2,3).
309     """
310
311     test_set(arguments)
312
313     """All poor scores for class 1 suggest difficulty distinguishing minimal levels of brown (i.e only in the neck, from 0). Perhaps
        binary classification would be more suitable (0,1 = 0; 2,3 = 3). However, weighted averages of  $> .9$  show high overall
        performance.

```

```
314
315 Single file inference (for manual checking):
316 """
317
318 model_path = os.path.join(root_dir, "runs/exp1/best.pt")
319 dicom_file = os.path.join(root_dir, "dataset/dicoms/388.dcm")
320
321 bf_pred = infer_single(model_path, dicom_file, img_size=224)
322
323 print(f"Brown fat prediction: {bf_pred}")
324 #correct value for this file is 2
```

Listing 2: 3D Brown fat removal - Phase 2

58

```
1 !pip install SimpleITK pydicom TotalSegmentator scipy
2
3 # Cell 1: Setup
4 import os
5 import glob
6 import numpy as np
7 import pydicom
8 import datetime
9 from pydicom.uid import generate_uid
10 #from google.colab import drive
11 from totalsegmentator.python_api import totalsegmentator
12 from scipy.ndimage import label as ndimage_label
13 import SimpleITK as sitk
14 import shutil
15
16 #drive.mount('/content/drive')
17
18 # Cell 1.1
19
20 root_dir = "/content/"
21
22 shutil.unpack_archive("c0138-main.zip", root_dir)
23
24 root_dir = "/content/c0138-main/"
25
26 base_dicom_path = os.path.join(root_dir, "dataset/3D")
```

```
27 # User input:
28 scan_number = "44" # < Enter scan number (Provided: 44, 108, 171)
29 #
30
31 # Cell 1.2: path configuration
32 ct_dir = os.path.join(base_dicom_path, scan_number, "CT/")
33 pet_dir = os.path.join(base_dicom_path, scan_number, "PET/")
34
35 output_dir = os.path.join(base_dicom_path, scan_number, "PET_scrubbed/")
36
37 segm_dir = os.path.join(base_dicom_path, scan_number, "segm_dir/")
38 multilabel_liver_nii_filename = "liver_segments.nii"
39 multilabel_liver_nii_path = os.path.join(segm_dir, multilabel_liver_nii_filename)
40
41
42 os.makedirs(output_dir, exist_ok=True)
43 os.makedirs(segm_dir, exist_ok=True)
44
45 # Cell 2
46 def load_series(directory):
47     """Return a list of (dataset, pixel_array) sorted by instance number."""
48     paths = sorted(
49         glob.glob(os.path.join(directory, "**", "*"), recursive=True),
50         key=lambda p: int(pydicom.dcmread(p, stop_before_pixels=True).InstanceNumber)
51     )
52     series = []
53     for p in paths:
```

```

54     ds = pydicom.dcmread(p)
55     arr = ds.pixel_array          # keep native dtype
56     series.append((ds, arr))
57     return series
58
59 # Cell 3          liver segmentation on CT series
60 def segment_liver_ct(ct_dir_local, pet_dir_local, multilabel_liver_nii_path_local):
61     """
62     Generates liver and fat masks in PET coordinate space.
63     Handles multi-label liver segmentation output from TotalSegmentator.
64
65     Args:
66         ct_dir_local (str): Path to the directory containing CT DICOM series.
67         pet_dir_local (str): Path to the directory containing PET DICOM series.
68         multilabel_liver_nii_path_local (str): Full path for TotalSegmentator multi-label output NIfTI file.
69
70     Returns:
71         tuple[np.ndarray, np.ndarray]: liver_mask_pet, fat_mask_pet (boolean arrays)
72     """
73
74     # 1) run TotalSegmentator (if needed)
75
76     if not os.path.exists(multilabel_liver_nii_path_local):
77         print(f"Running TotalSegmentator for liver segments...")
78
79         os.makedirs(os.path.dirname(multilabel_liver_nii_path_local), exist_ok=True)
80         try:

```

```

81     totalsegmentator(
82         input=ct_dir_local,
83         output=multilabel_liver_nii_path_local,
84         task="liver_segments",
85         ml=True,
86         device="gpu"
87     )
88     print(f"TotalSegmentator finished. Expecting multi-label file: {multilabel_liver_nii_path_local}")
89     if not os.path.exists(multilabel_liver_nii_path_local):
90         raise FileNotFoundError(f"TotalSegmentator did not produce expected file: {multilabel_liver_nii_path_local}")
91 except Exception as e:
92     print(f"Error running TotalSegmentator: {e}")
93     try:
94         reader_pet_check = sitk.ImageSeriesReader()
95         uid_pet_check = reader_pet_check.GetGDCMSeriesIDs(pet_dir_local)[0]
96         reader_pet_check.SetFileNames(reader_pet_check.GetGDCMSeriesFileNames(pet_dir_local, uid_pet_check))
97         pet_ref_check = reader_pet_check.Execute()
98         pet_shape = sitk.GetArrayFromImage(pet_ref_check).shape
99         return np.zeros(pet_shape, dtype=bool), np.zeros(pet_shape, dtype=bool)
100 except Exception as pet_e:
101     print(f"Could not read PET dimensions to create empty masks: {pet_e}")
102     return np.zeros((1,1,1), dtype=bool), np.zeros((1,1,1), dtype=bool)
103
104 print(f"Loading multi-label liver segmentation from: {multilabel_liver_nii_path_local}")
105 liver_multilabel_nii = sitk.ReadImage(multilabel_liver_nii_path_local)
106
107 liver_binary_mask_nii = (liver_multilabel_nii >= 1) & (liver_multilabel_nii <= 8)

```



```
liver_binary_mask_nii = sitk.Cast(liver_binary_mask_nii, sitk.sitkUInt8)
print("Combined liver segments 1-8 into a single binary mask.")

print(f"Loading CT series from: {ct_dir_local}")
reader_ct = sitk.ImageSeriesReader()
uid_ct = reader_ct.GetGDCMSeriesIDs(ct_dir_local)[0]
reader_ct.SetFileNames(reader_ct.GetGDCMSeriesFileNames(ct_dir_local, uid_ct))
ct_img = reader_ct.Execute()
slope = float(ct_img.GetMetaData('0028|1053') if ct_img.HasMetaDataKey('0028|1053') else 1.0)
intercept = float(ct_img.GetMetaData('0028|1052') if ct_img.HasMetaDataKey('0028|1052') else 0.0)
ct_arr = sitk.GetArrayFromImage(ct_img)
hu_arr = ct_arr * slope + intercept
fat_ct_arr = ((hu_arr >= -190) & (hu_arr <= -20)).astype(np.uint8)
fat_img = sitk.GetImageFromArray(fat_ct_arr)
fat_img.CopyInformation(ct_img)
print(f"Generated CT fat mask (shape: {fat_ct_arr.shape})")

print(f"Loading PET series reference geometry from: {pet_dir_local}")
reader_pet = sitk.ImageSeriesReader()
uid_pet = reader_pet.GetGDCMSeriesIDs(pet_dir_local)[0]
reader_pet.SetFileNames(reader_pet.GetGDCMSeriesFileNames(pet_dir_local, uid_pet))
pet_ref = reader_pet.Execute()
print(f"PET reference image loaded (shape: {sitk.GetArrayFromImage(pet_ref).shape})")

print("Resampling masks to PET grid...")
def resample_to_pet(img_to_resample, interpolation=sitk.sitkNearestNeighbor):
    ref_size = pet_ref.GetSize()
```

```
135     ref_spacing = pet_ref.GetSpacing()
136     ref_origin = pet_ref.GetOrigin()
137     ref_direction = pet_ref.GetDirection()
138     transform = sitk.Transform()
139     resampled_img = sitk.Resample(
140         img_to_resample, ref_size, transform, interpolation, ref_origin,
141         ref_spacing, ref_direction, 0, img_to_resample.GetPixelID()
142     )
143     return resampled_img
144
145 liver_rs = resample_to_pet(liver_binary_mask_nii, sitk.sitkNearestNeighbor)
146 fat_rs = resample_to_pet(fat_img, sitk.sitkNearestNeighbor)
147 print("Resampling complete.")
148
149 liver_mask_pet = sitk.GetArrayFromImage(liver_rs).astype(bool)
150 fat_mask_pet = sitk.GetArrayFromImage(fat_rs).astype(bool)
151 print(f"Final PET-space liver mask shape: {liver_mask_pet.shape}, "
152       f"Fat mask shape: {fat_mask_pet.shape}")
153
154 return liver_mask_pet, fat_mask_pet
155
156 # Cell 3.1 - Helper functions
157
158 def remove_small_3d_clusters(mask_3d, min_voxels_3d):
159     if not mask_3d.any() or min_voxels_3d <= 0:
160         return mask_3d
161
```

```
162 labeled_mask, num_labels = ndimage_label(mask_3d)
163
164 if num_labels == 0:
165     return mask_3d
166
167 label_sizes = np.bincount(labeled_mask.ravel())
168
169
170 too_small_labels = np.where((label_sizes[1:] < min_voxels_3d))[0] + 1
171
172 small_cluster_mask = np.isin(labeled_mask, too_small_labels)
173
174 output_mask = mask_3d.copy()
175 output_mask[small_cluster_mask] = False
176
177 print(f"  Removed {len(too_small_labels)} 3D clusters smaller than {min_voxels_3d} voxels.")
178 return output_mask
179
180 # Cell 4          BAT detection & suppression
181
182 def suppress_bat(pet_series,
183                 liver_mask_pet, fat_mask_pet,
184                 liver_mult=1.2,
185                 min_3d_cluster_voxels=50):
186
187     print(f"\n--- Starting suppress_bat ---")
```

```

188 print(f"Total slices: {len(pet_series)}")
189 print(f"Input liver mask shape: {liver_mask_pet.shape}, Any True: {liver_mask_pet.any()}")
190 print(f"Input fat mask shape: {fat_mask_pet.shape}, Any True: {fat_mask_pet.any()}")
191 print(f"Liver multiplier: {liver_mult}, Min 3D Cluster Voxels: {min_3d_cluster_voxels}")
192
193 if not liver_mask_pet.any() or not fat_mask_pet.any():
194     print("ERROR: Input liver or fat mask is empty!")
195     return [arr for ds, arr in pet_series]
196
197 # --- Step 1: Stack PET volume ---
198 try:
199     pet_volume_f32 = np.stack([arr.astype(np.float32, copy=False) for ds, arr in pet_series], axis=0)
200     if pet_volume_f32.shape != liver_mask_pet.shape:
201         raise ValueError(f"Shape mismatch: PET Volume {pet_volume_f32.shape} vs Masks {liver_mask_pet.shape}")
202 except Exception as e:
203     print(f"ERROR: Could not stack PET slices or shape mismatch: {e}")
204     return [arr for ds, arr in pet_series]
205
206 # --- Step 2: Calculate global medians ---
207 pet_liver_values_3d = pet_volume_f32[liver_mask_pet]
208 pet_fat_values_3d = pet_volume_f32[fat_mask_pet]
209 if pet_liver_values_3d.size == 0 or pet_fat_values_3d.size == 0:
210     print("ERROR: Masks cover zero voxels in PET volume!")
211     return [arr for ds, arr in pet_series]
212
213 global_liver_median = np.median(pet_liver_values_3d)
214 global_adipose_median = np.median(pet_fat_values_3d)

```

```

215 global_bat_threshold = liver_mult * global_liver_median
216
217 print(f"Global Liver Median: {global_liver_median:.2f}")
218 print(f"Global Adipose Median: {global_adipose_median:.2f}")
219 print(f"Global BAT Threshold (>= {liver_mult} * liver_med): {global_bat_threshold:.2f}")
220 target_patch_value = np.round(global_adipose_median).astype(pet_series[0][1].dtype)
221 print(f"Target Patch Value: {target_patch_value}") # Verify patch value
222
223 # --- Step 3: Determine Z-range based on liver mask ---
224 liver_z_indices = np.where(np.any(liver_mask_pet, axis=(1, 2)))[0]
225 if liver_z_indices.size == 0:
226     print("WARNING: No liver mask found. Processing all slices.")
227     min_process_z = 0
228     max_process_z_ignored = pet_volume_f32.shape[0] - 1
229 else:
230     # process from top slice down to the minimum index containing liver
231     min_process_z = np.min(liver_z_indices)
232     max_process_z_ignored = pet_volume_f32.shape[0] - 1
233     print(f"Processing slices from index {max_process_z_ignored} down to {min_process_z} (inclusive, based on liver mask
234           extent).")
235
236 # --- Step 4: Build initial mask ---
237 initial_bat_mask_3d = np.zeros_like(liver_mask_pet, dtype=bool)
238 slices_considered_for_mask = 0
239 # iterate through all slices, but only apply threshold if within range
240 for idx in range(pet_volume_f32.shape[0]):
241     if idx < min_process_z: # skip slices below the liver's minimum extent

```

```

241         continue
242
243     pet_f32_slice = pet_volume_f32[idx]
244     fat_mask_slice = fat_mask_pet[idx]
245
246     if not fat_mask_slice.any(): continue
247
248     bat_slice = fat_mask_slice & (pet_f32_slice >= global_bat_threshold)
249     initial_bat_mask_3d[idx] = bat_slice
250     if bat_slice.any():
251         slices_considered_for_mask += 1
252
253 total_initial_candidates = np.sum(initial_bat_mask_3d)
254 print(f"Found {total_initial_candidates} initial BAT candidate voxels across relevant slices (index >= {min_process_z}).")
255
256 # --- Step 5: 3D cluster filtering ---
257 final_bat_mask_3d = remove_small_3d_clusters(initial_bat_mask_3d, min_3d_cluster_voxels)
258 total_final_bat_voxels = np.sum(final_bat_mask_3d)
259 print(f"Total BAT voxels remaining after 3D cluster filter: {total_final_bat_voxels}")
260
261 # --- Step 6: Patch slices ---
262 new_pet_arrays = []
263 slices_modified = 0
264 for idx, (ds, pet_arr_orig) in enumerate(pet_series):
265
266     bat_mask_slice = final_bat_mask_3d[idx]
267

```

```

268         if bat_mask_slice.any():
269             patched = pet_arr_orig.copy()
270             patched[bat_mask_slice] = target_patch_value
271             new_pet_arrays.append(patched)
272             slices_modified += 1
273         else:
274             # No BAT on this slice after filtering, or slice outside Z range
275             new_pet_arrays.append(pet_arr_orig)
276
277     print(f"\n--- suppress_bat Finished ---")
278     print(f"Slices actually modified (within index range >= {min_process_z}): {slices_modified}")
279     print(f"Total BAT voxels patched (3D count): {total_final_bat_voxels}")
280     print(f"-----\n")
281     if slices_modified == 0:
282         print("WARNING: No slices were modified.")
283     if total_final_bat_voxels == 0 and total_initial_candidates > 0:
284         print("WARNING: Initial BAT candidates found, but zero voxels remained after 3D cluster filtering.")
285
286     return new_pet_arrays
287
288 # Cell 5
289
290 def save_modified_pet(pet_series, new_arrays, out_dir):
291     """
292     pet_series: list of (ds, _) tuples from load_series(pet_dir)
293     new_arrays: list of numpy arrays, same length and shape as pet_series
294     """

```

```
295 os.makedirs(out_dir, exist_ok=True)
296
297 new_series_uid = generate_uid()
298
299 for idx, ((ds, _), arr) in enumerate(zip(pet_series, new_arrays), start=1):
300
301     rows, cols = int(ds.Rows), int(ds.Columns)
302     if arr.shape != (rows, cols):
303         raise ValueError(
304             f"Slice {idx:03d} shape mismatch: arr is {arr.shape}, "
305             f"but DICOM says {(rows, cols)}"
306         )
307
308     ds_mod = ds.copy()
309     ds_mod.SeriesInstanceUID = new_series_uid
310     ds_mod.SOPInstanceUID = generate_uid()
311
312     # enforce correct dimensions
313     ds_mod.Rows = rows
314     ds_mod.Columns = cols
315
316
317     ds_mod.PixelData = arr.tobytes()
318
319     # 0001.dcm, 0002.dcm,
320     out_path = os.path.join(out_dir, f"{idx:04d}.dcm")
321     ds_mod.save_as(out_path)
```



```
322
323     print(f"Wrote {len(new_arrays)} full-resolution PET slices to {out_dir}")
324
325 # Utility: sort a (dataset, array) series by absolute z-position
326 def sort_series_by_z(series):
327     return sorted(series,
328                   key=lambda pair: float(pair[0].ImagePositionPatient[2]))
329
330 # Cell 7: Main execution
331
332 if not os.path.isdir(ct_dir):
333     print(f"ERROR: CT directory not found: {ct_dir}")
334 elif not os.path.isdir(pet_dir):
335     print(f"ERROR: PET directory not found: {pet_dir}")
336 else:
337     print("Input directories confirmed.")
338
339 # load DICOM PET series (sorted)
340 pet_series = sort_series_by_z(load_series(pet_dir))
341
342 # get liver & fat masks
343 liver_mask_pet, fat_mask_pet = segment_liver_ct(ct_dir, pet_dir, multilabel_liver_nii_path)
344
345 # scrub
346 scrubbed = suppress_bat(
347     pet_series,
348     liver_mask_pet,
```

```
349         fat_mask_pet,
350         liver_mult=1.2,
351         min_3d_cluster_voxels=10 # hyper parameters, found these a good compromise
352     )
353
354     save_modified_pet(pet_series, scrubbed, output_dir)
355     print(f"BAT suppressed with TS-hybrid      check: {output_dir}")
356
357     archive_base_name = os.path.join(base_dicom_path, scan_number, f"PET_scrubbed_{scan_number}")
358     shutil.make_archive(archive_base_name, 'zip', output_dir)
359     print(f"Zipped output to: {archive_base_name}.zip")
```