Generating new synthesis PET images through out deep learning model (Topic -11)

Nguyen Thi Van,

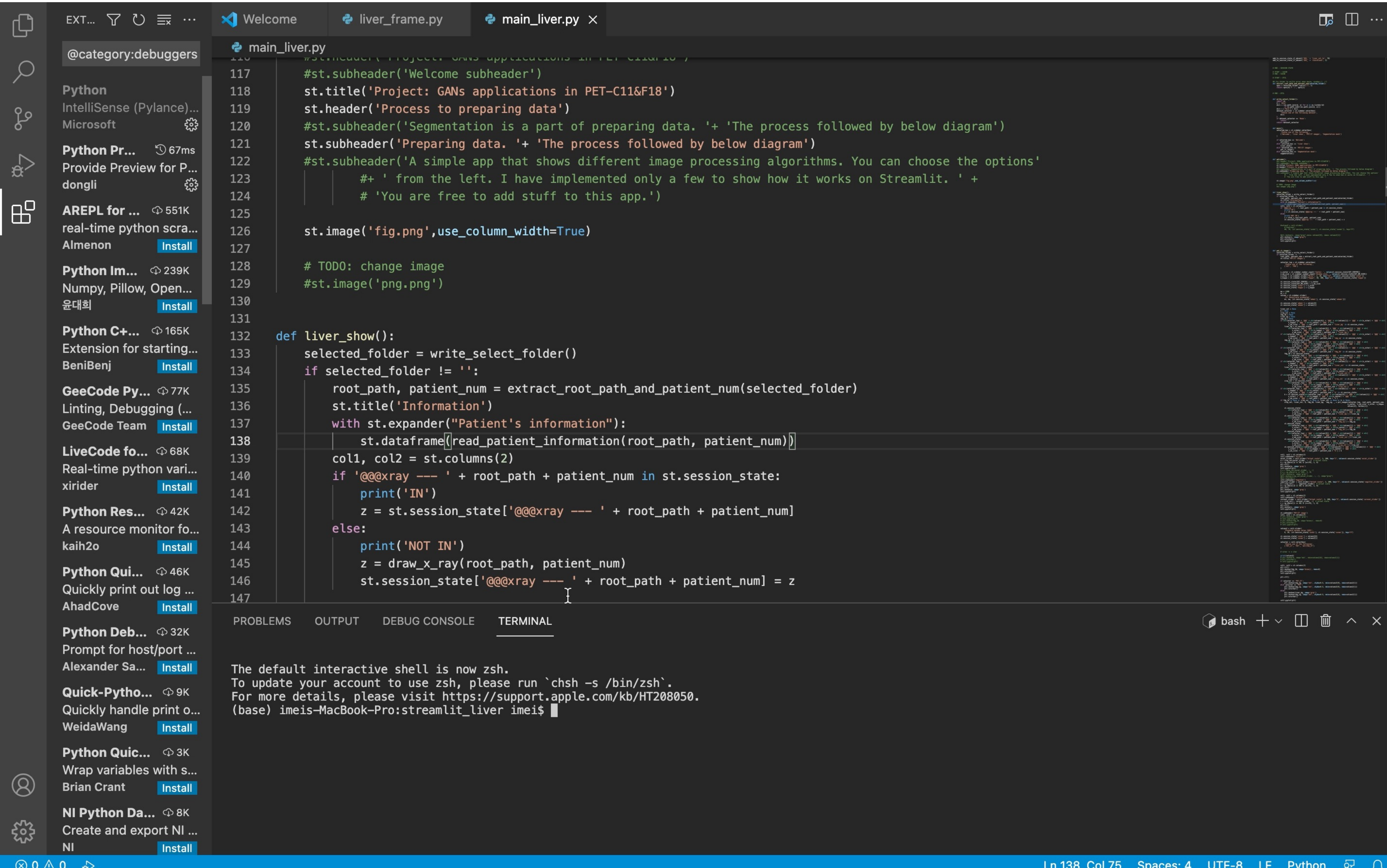
MSc candidate student

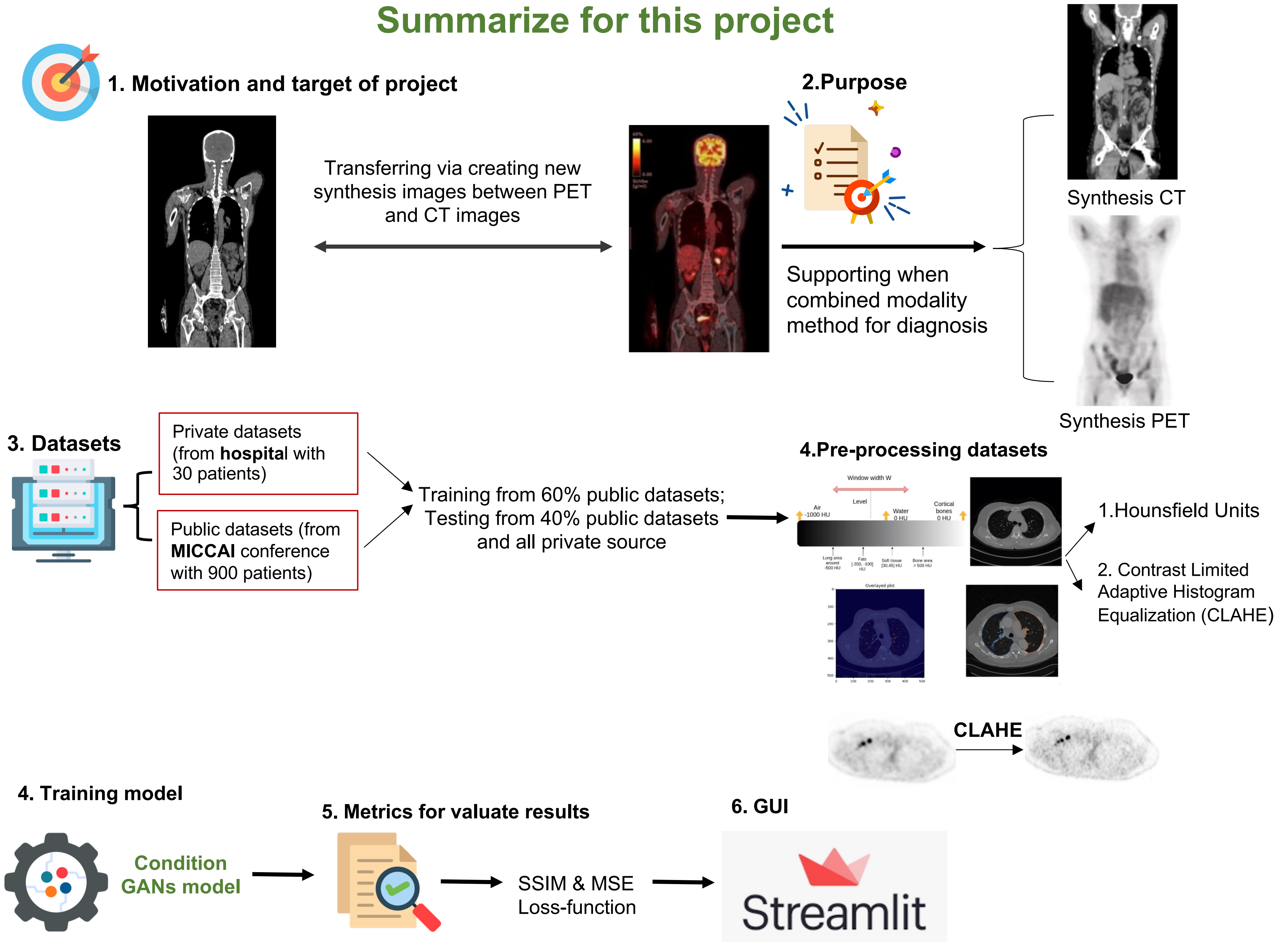
Department of Artificial Intelligence Convergence

Chonnam National University

Student ID: 217366

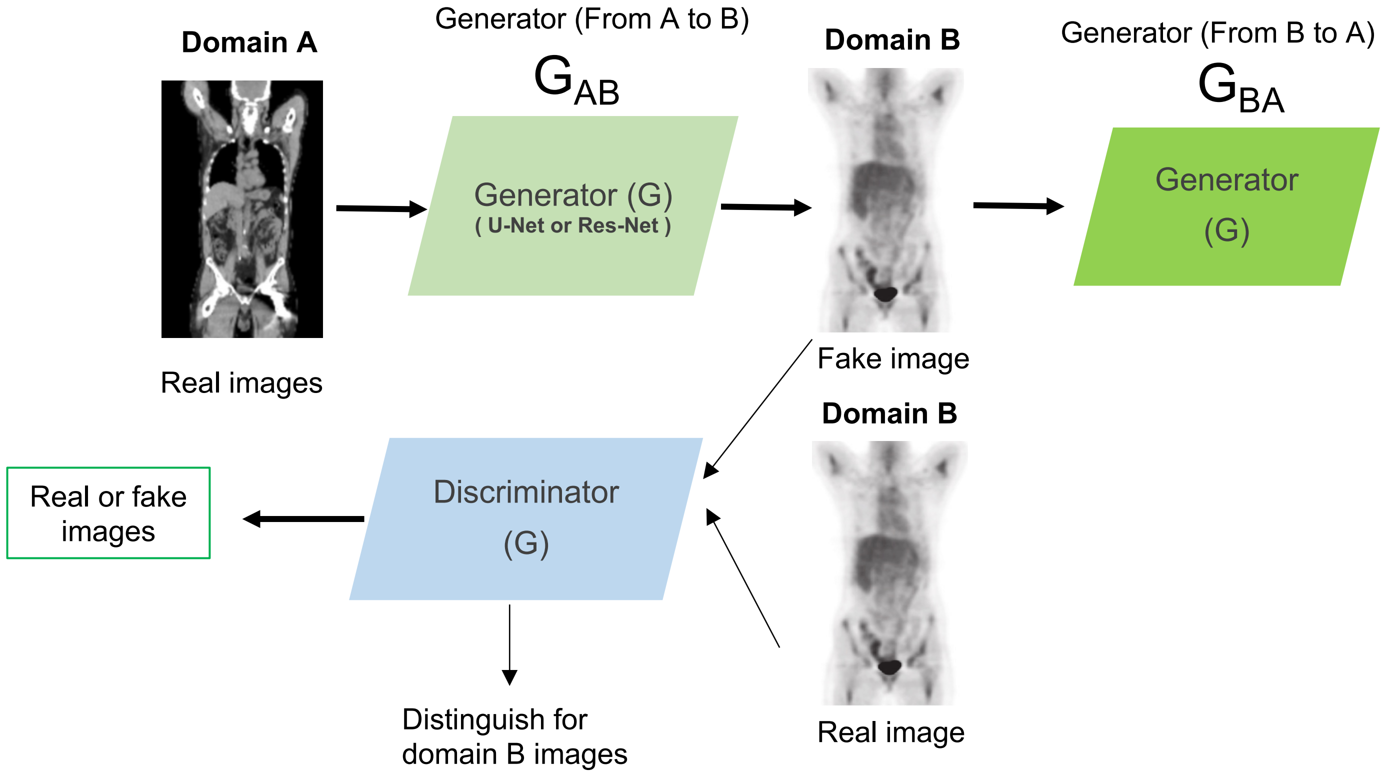
Video Demo

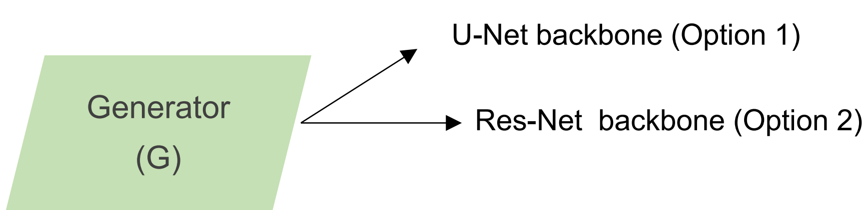

EXT...
@category:debuggers
Python
IntelliSense (Pylance)...
Microsoft
Python Pr... 67ms
Provide Preview for P...
dongli
AREPL for ... 551K
real-time python scra...
Almenon
Install
Python Im... 239K
Numpy, Pillow, Open...
윤대희
Install
Python C+... 165K
Extension for starting...
BeniBenj
Install
GeeCode Py... 77K
Linting, Debugging (...
GeeCode Team
Install
LiveCode fo... 68K
Real-time python vari...
xirider
Install
Python Res... 42K
A resource monitor fo...
kaih2o
Install
Python Qui... 46K
Quickly print out log ...
AhadCove
Install
Python Deb... 32K
Prompt for host/port ...
Alexander Sa...
Install
Quick-Pytho... 9K
Quickly handle print o...
WeidaWang
Install
Python Quic... 3K
Wrap variables with s...
Brian Crant
Install
NI Python Da... 8K
Create and export NI ...
NI
Install
Welcome
liver_frame.py
main_liver.py
main_liver.py
#st.subheader('Welcome subheader')
st.title('Project: GANs applications in PET-C11&F18')
st.header('Process to preparing data')
#st.subheader('Segmentation is a part of preparing data. '+ 'The process followed by below diagram')
st.subheader('Preparing data. '+ 'The process followed by below diagram')
#st.subheader('A simple app that shows different image processing algorithms. You can choose the options'
#+ ' from the left. I have implemented only a few to show how it works on Streamlit. ' +
# 'You are free to add stuff to this app.')
st.image('fig.png',use_column_width=True)
# TODO: change image
#st.image('png.png')
def liver_show():
selected_folder = write_select_folder()
if selected_folder != '':
root_path, patient_num = extract_root_path_and_patient_num(selected_folder)
st.title('Information')
with st.expander("Patient's information"):
st.dataframe(read_patient_information(root_path, patient_num))
col1, col2 = st.columns(2)
if '@@@xray --- ' + root_path + patient_num in st.session_state:
print('IN')
z = st.session_state['@@@xray --- ' + root_path + patient_num]
else:
print('NOT IN')
z = draw_x_ray(root_path, patient_num)
st.session_state['@@@xray --- ' + root_path + patient_num] = z
PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
bash
The default interactive shell is now zsh.
To update your account to use zsh, please run `chsh -s /bin/zsh`.
For more details, please visit https://support.apple.com/kb/HT208050.
(base) imeis-MacBook-Pro:streamlit_liver imei$
0 0
Ln 138, Col 75
Spaces: 4
UTF-8
LF
Python

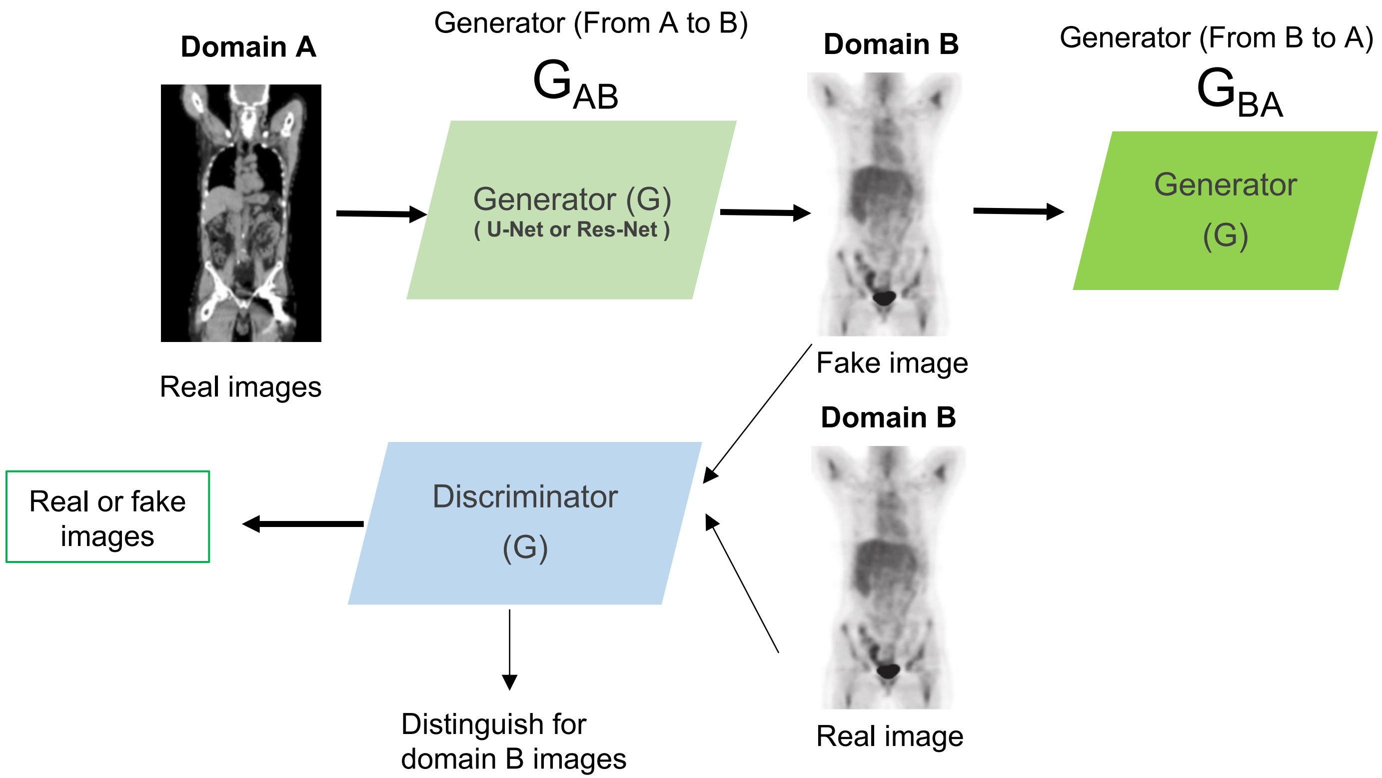


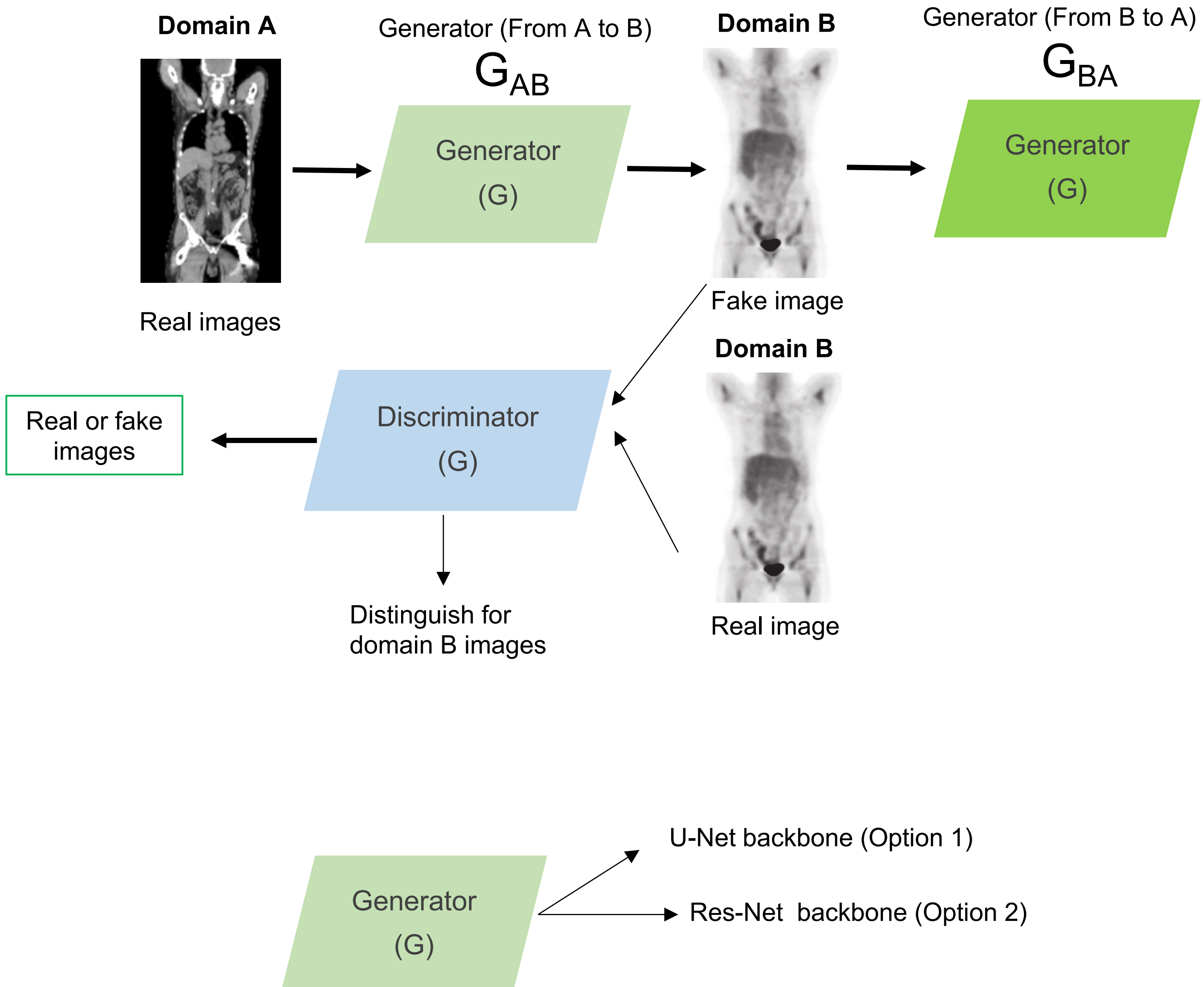
Summarize for this project
1. Motivation and target of project
Transferring via creating new synthesis images between PET and CT images
2.Purpose
Supporting when combined modality method for diagnosis
Synthesis CT
Synthesis PET
3. Datasets
Private datasets (from hospital with 30 patients)
Public datasets (from MICCAI conference with 900 patients)
Training from 60% public datasets; Testing from 40% public datasets and all private source
4.Pre-processing datasets
Window width W
Level
Air -1000 HU
Water 0 HU
Cortical bones 0 HU
Overlayed plot
1.Hounsfield Units
2. Contrast Limited Adaptive Histogram Equalization (CLAHE)
CLAHE
4. Training model
Condition GANs model
5. Metrics for valuate results
SSIM & MSE Loss-function
6. GUI
Streamlit

4. Training model



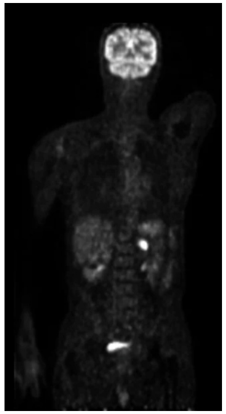



Domain A
Generator (From A to B)
G_{AB}
Domain B
Generator (From B to A)
G_{BA}
Generator (G)
(U-Net or Res-Net)
Generator
(G)
Real images
Fake image
Domain B
Real or fake
images
Discriminator
(G)
Distinguish for
domain B images
Real image

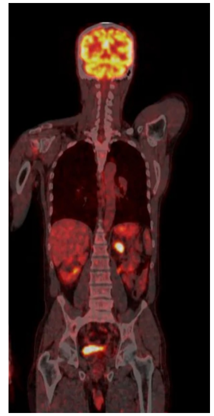

Domain A
Generator (From A to B)
G_{AB}
Domain B
Generator (From B to A)
G_{BA}
Generator
(G)
Generator
(G)
Real images
Fake image
Domain B
Real or fake images
Discriminator
(G)
Distinguish for domain B images
Real image
U-Net backbone (Option 1)
Generator
(G)
Res-Net backbone (Option 2)

5. Analyzing datasets

CT whole body



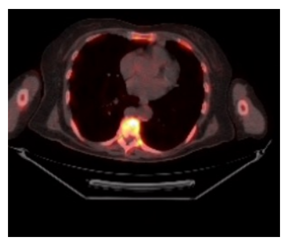
PET whole body



PET whole body showing via SUV values

CT axial

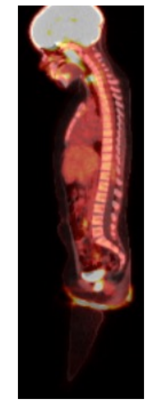
PET axial



PET axial via SUV values

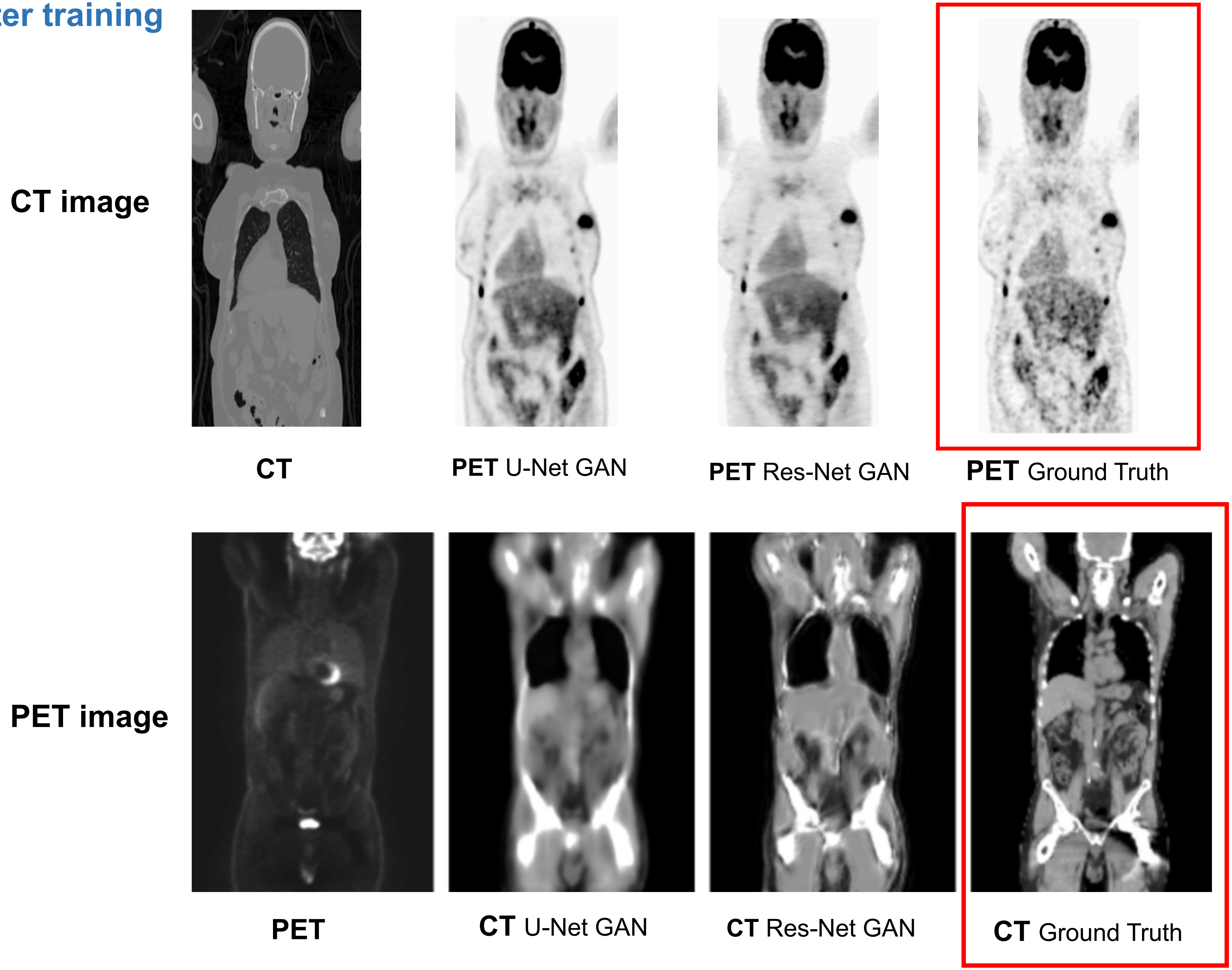
CT sagittal

PET sagittal



PET sagittal via SUV values

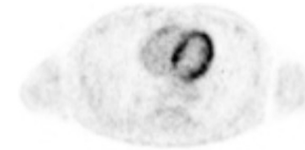
After training



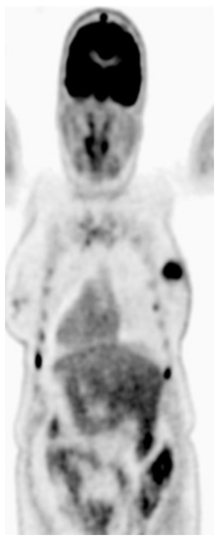
Axial

PET U-Net GAN

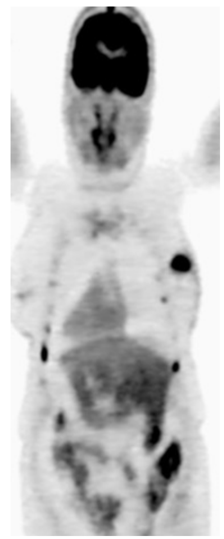
PET Res-Net GAN



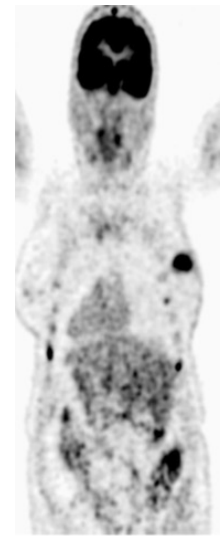
PET Ground Truth



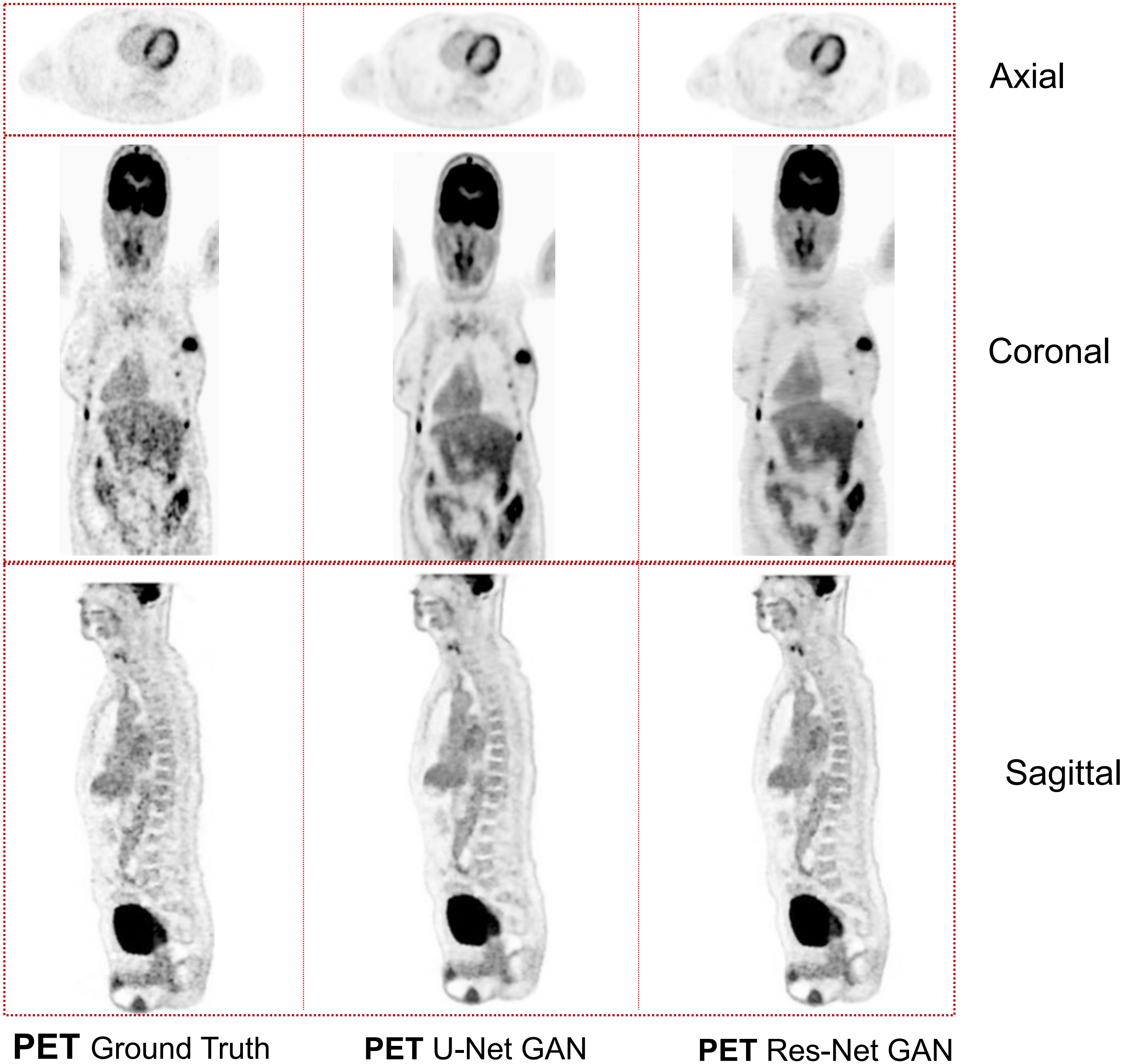
PET U-Net GAN

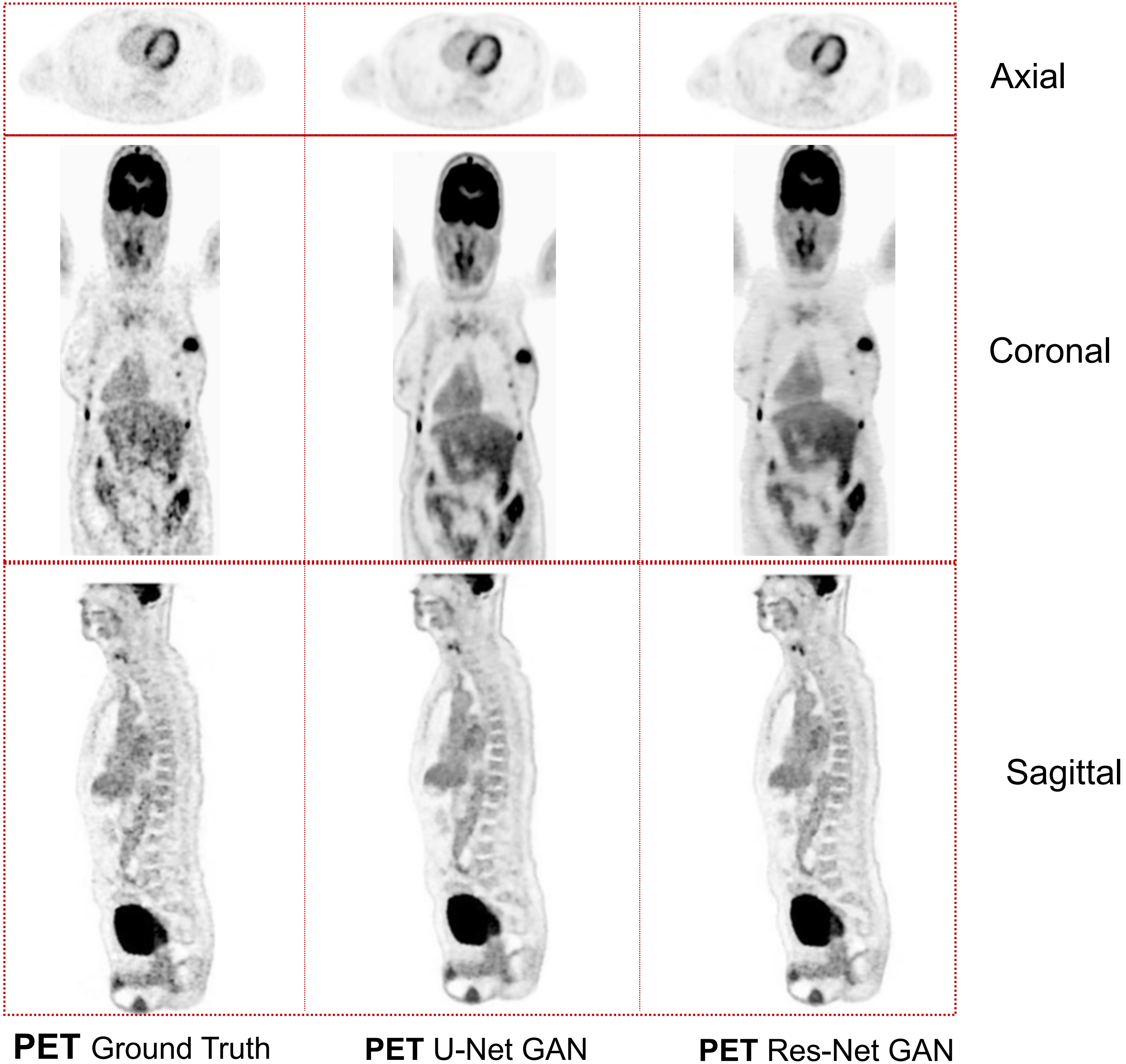


PET Res-Net GAN

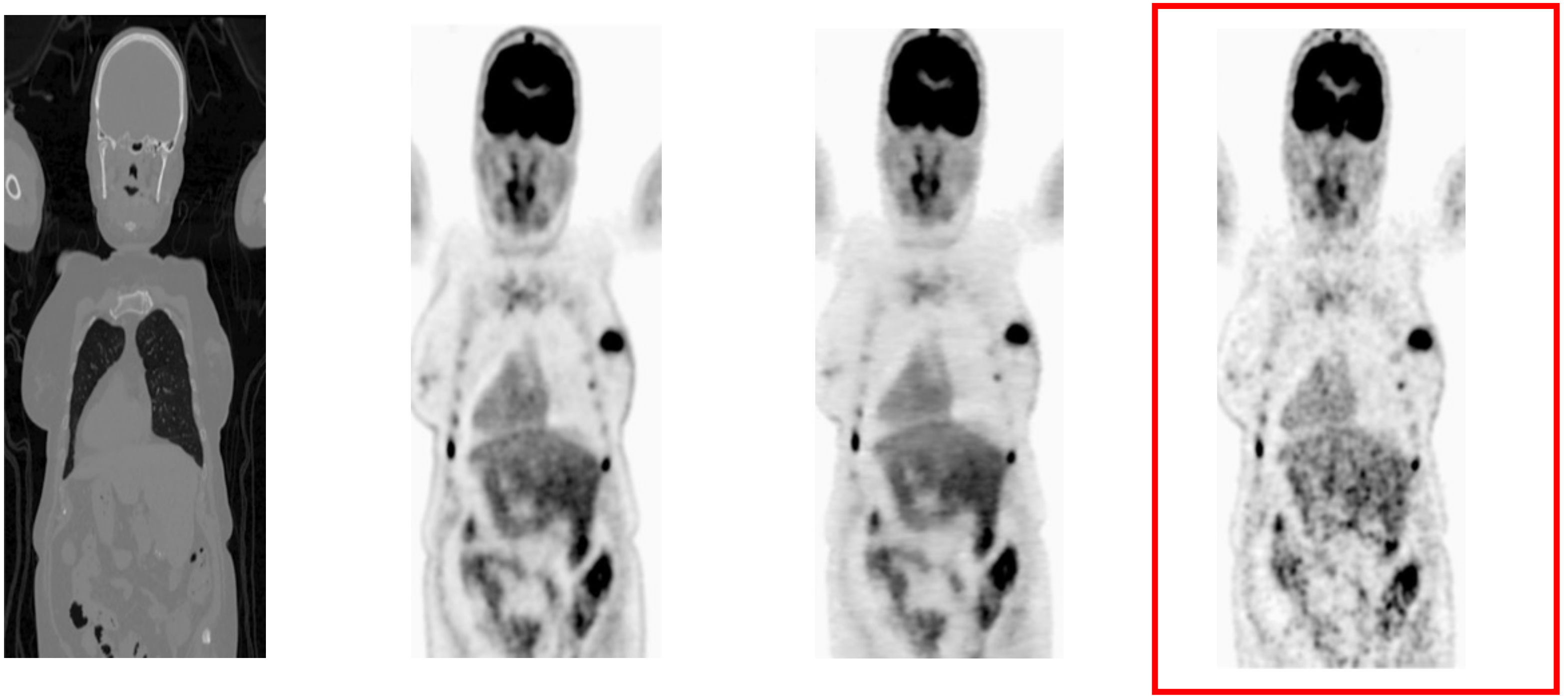


PET Ground Truth


Axial
Coronal
Sagittal
PET Ground Truth
PET U-Net GAN
PET Res-Net GAN

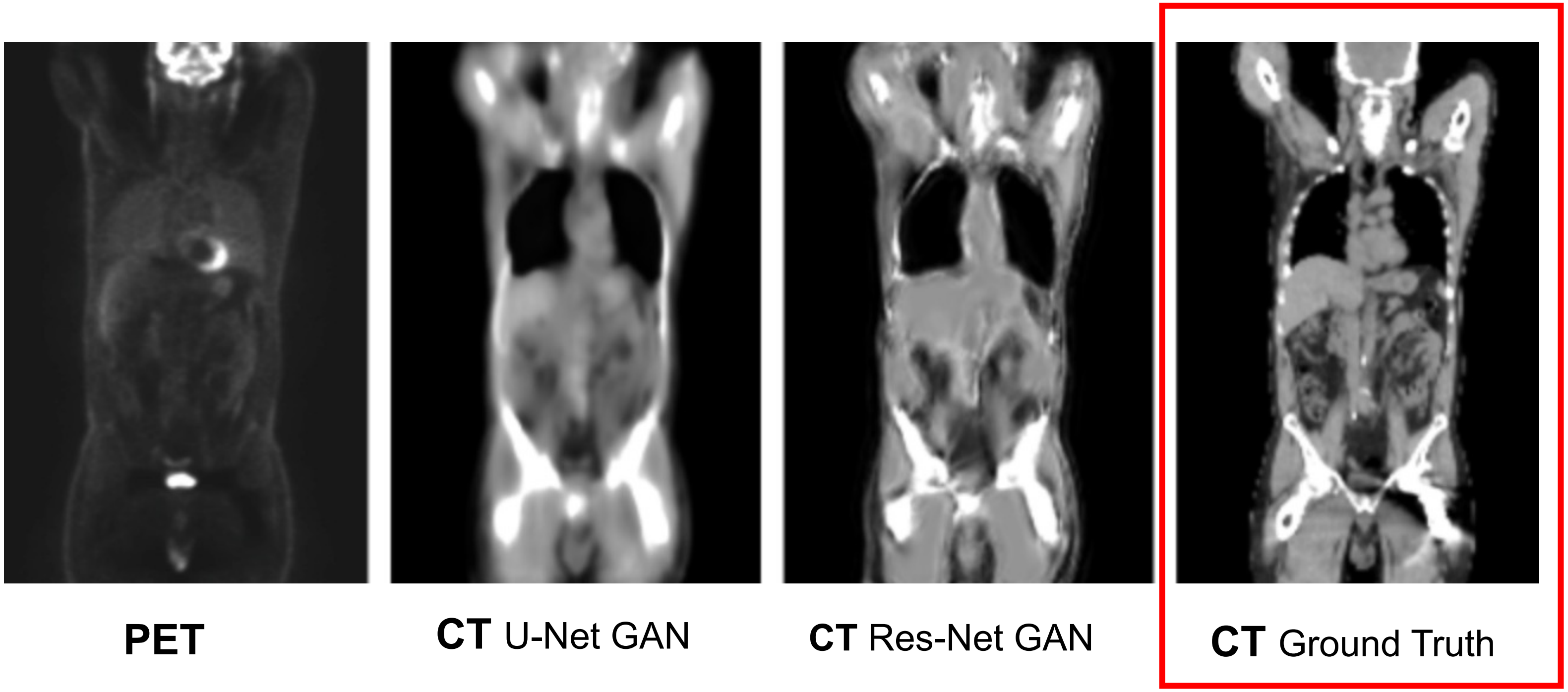

Axial
Coronal
Sagittal
PET Ground Truth
PET U-Net GAN
PET Res-Net GAN

6. Results



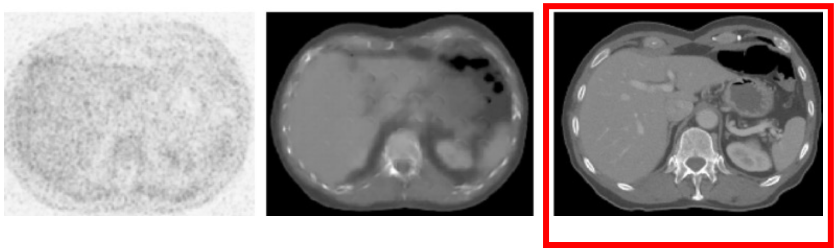
CT **PET** U-Net GAN **PET** Res-Net GAN **PET** Ground Truth

CT images synthesis with **whole body** images



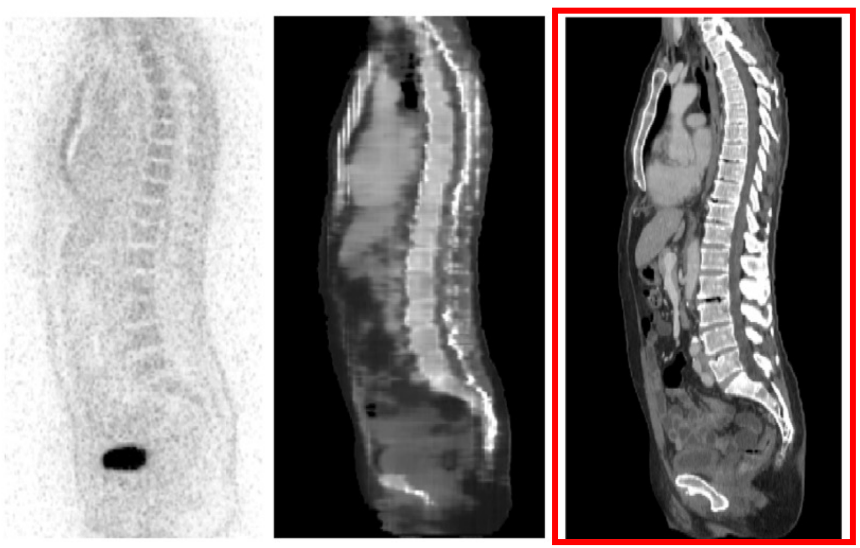
CT images synthesis with **each slice for each position**

1. Axial



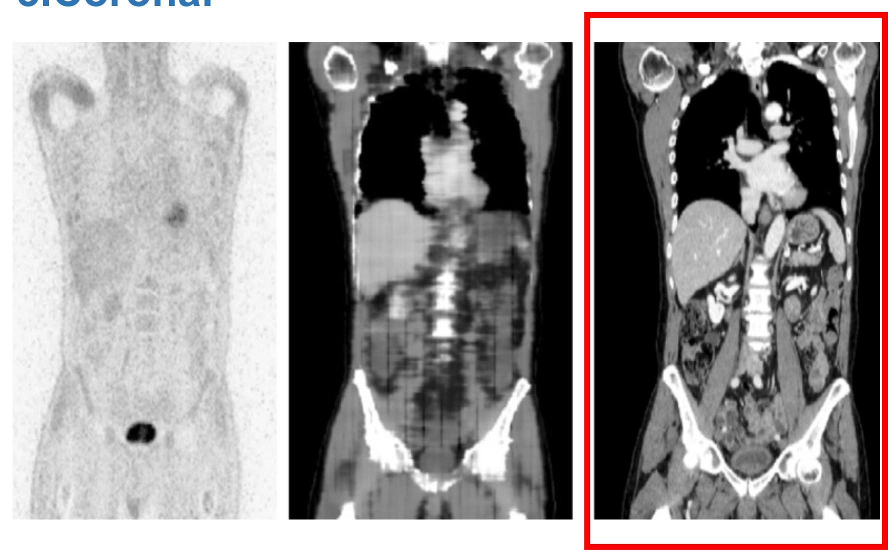
PET | CT GAN | CT ground truth

2. Sagittal



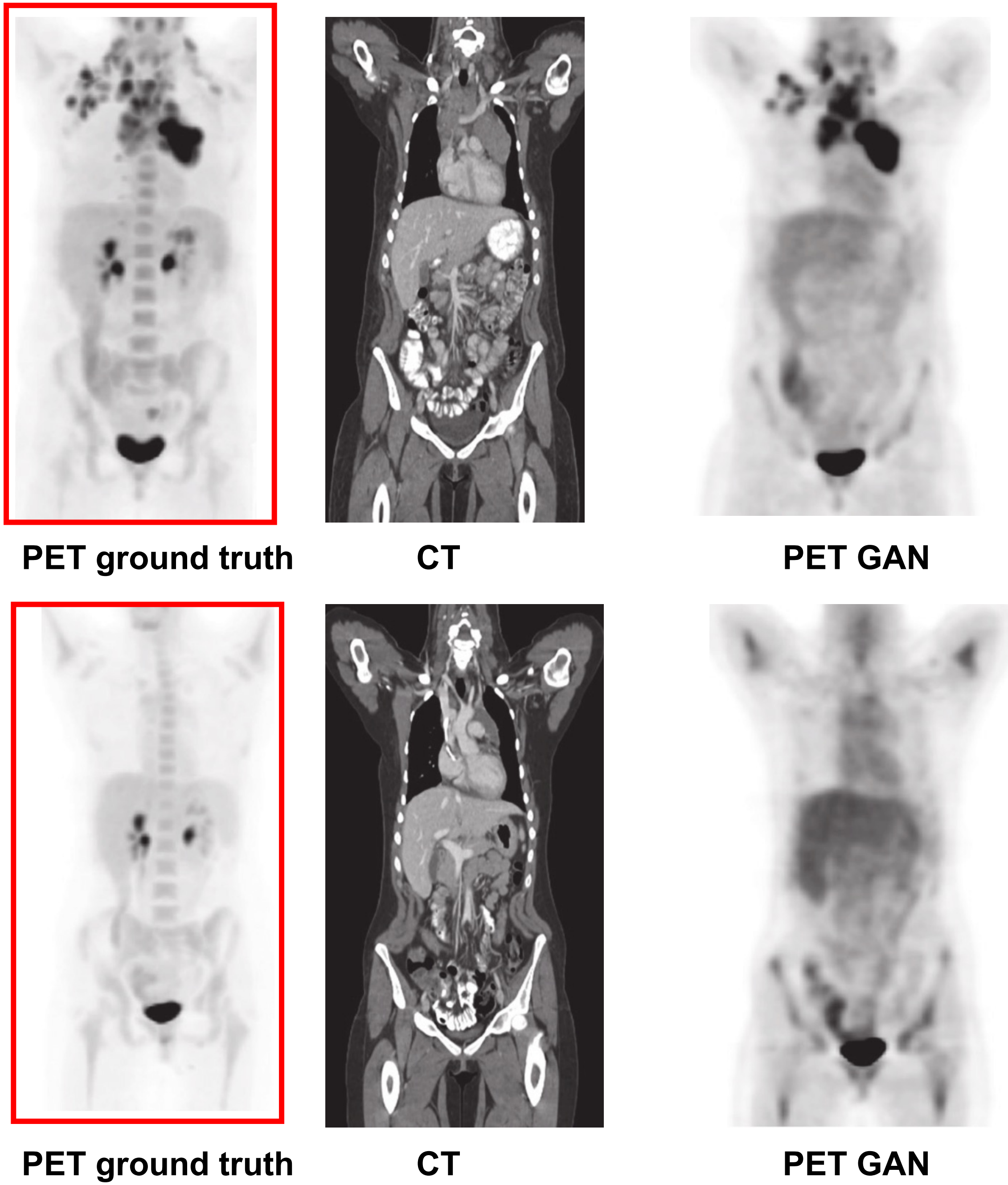
PET | CT GAN | CT ground truth

3.Coronal

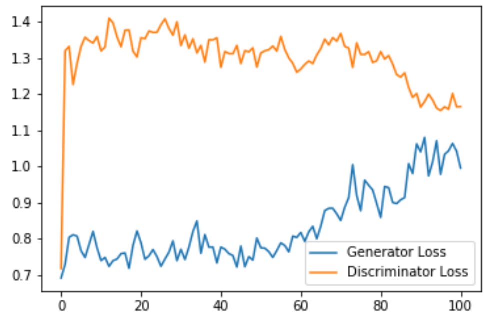


PET | CT GAN | CT ground truth

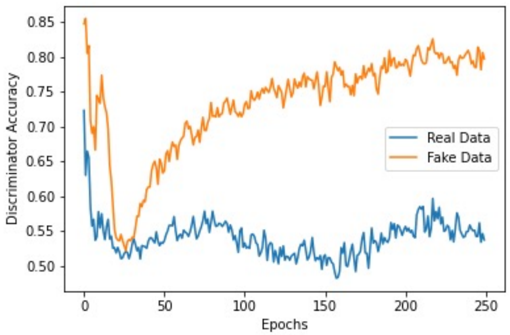
PET images synthesis with **whole body** images



Loss function with U-Net backbone



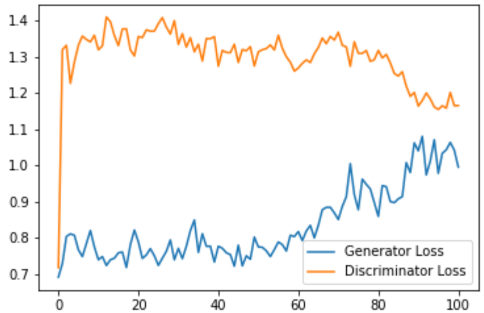
Loss function with Res-Net backbone



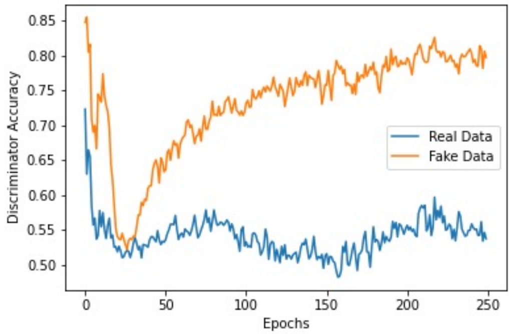
<u>SSIM & MSE</u>

	Data augmentation		Data non-augmentation	
	U-Net	Res-Net	U-Net	Res-Net
SSIM	0.65	0.72	0.52	0.31
MSE	0.16	0.09	0.25	0.17

Loss function with U-Net backbone



Loss function with Res-Net backbone



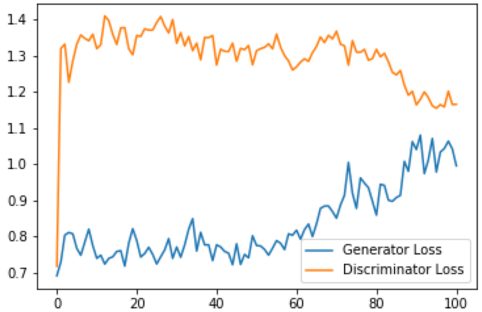
<u>SSIM & MSE</u>

	Data augmentation		**Data non-augmentation**	
	U-Net	Res-Net	U-Net	Res-Net
SSIM	0.65	0.72	0.52	0.31
MSE	0.16	0.09	0.25	0.17

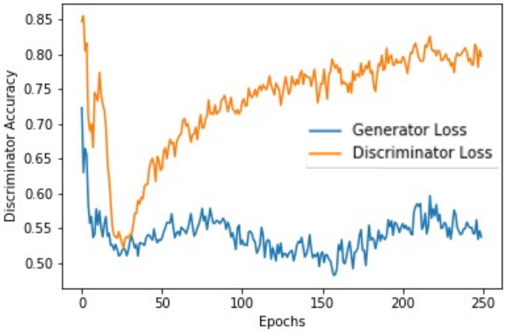
7. Evaluation

7.1 Loss function

Loss function with U-Net backbone



Loss function with Res-Net backbone



7.2 SSIM & MSE

The Structural Similarity Index (SSIM) metric extracts 3 key *features* from an image:

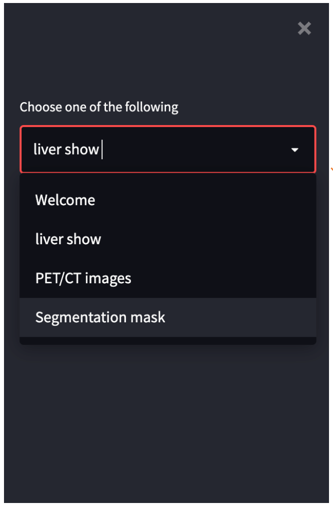
- **Luminance**
- **Contrast**
- **Structure**

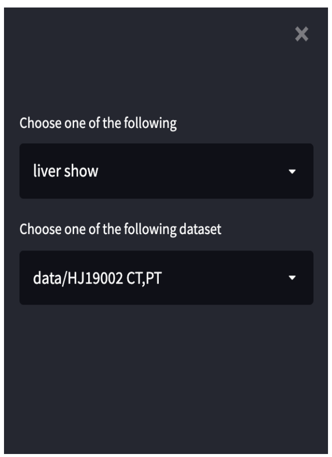
The comparison the similarity between the two images is performed on the basis of these 3 features.

$$\mathrm{SSIM}(\mathbf{x},\mathbf{y}) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{\left(\mu_x^2 + \mu_y^2 + C_1\right)\left(\sigma_x^2 + \sigma_y^2 + C_2\right)}.$$

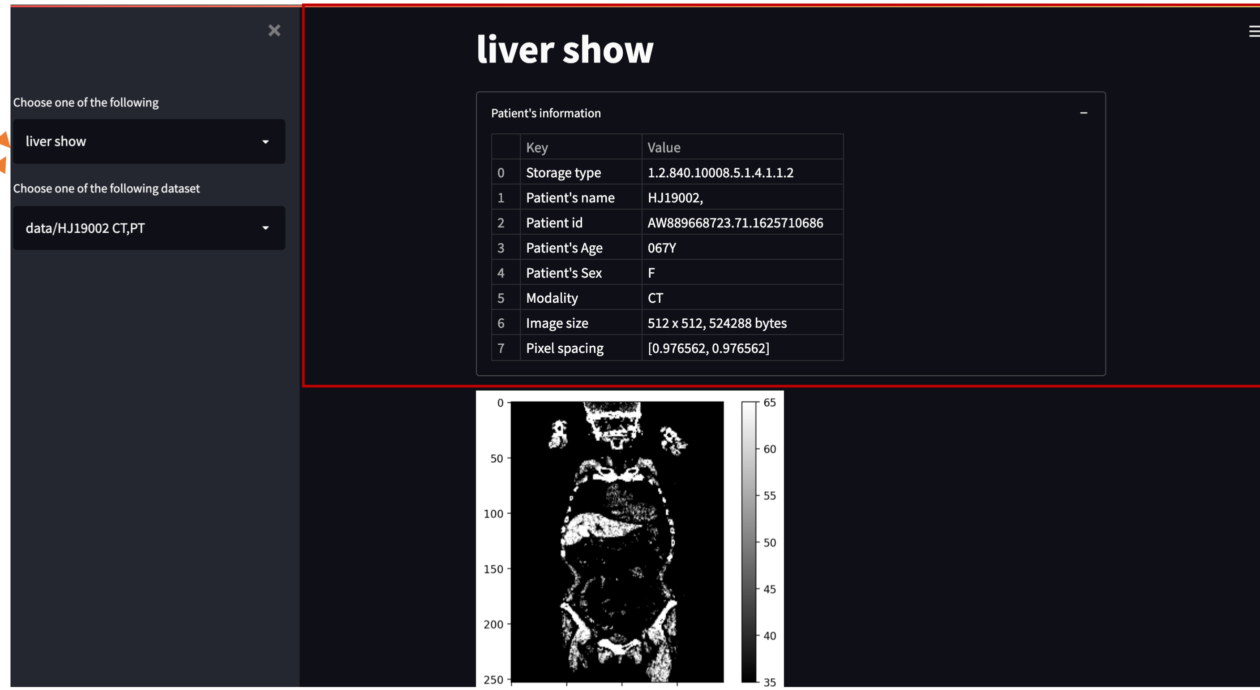
	Data augmentation		**Data non-augmentation**	
	U-Net	Res-Net	U-Net	Res-Net
SSIM	0.65	0.72	0.52	0.31
MSE	0.16	0.09	0.25	0.17

8. Interface to show

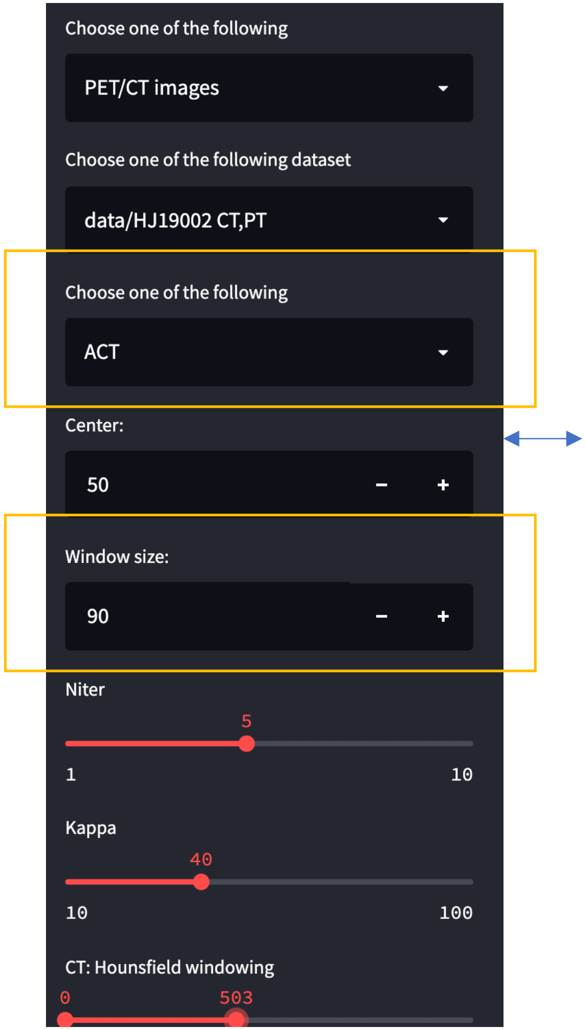


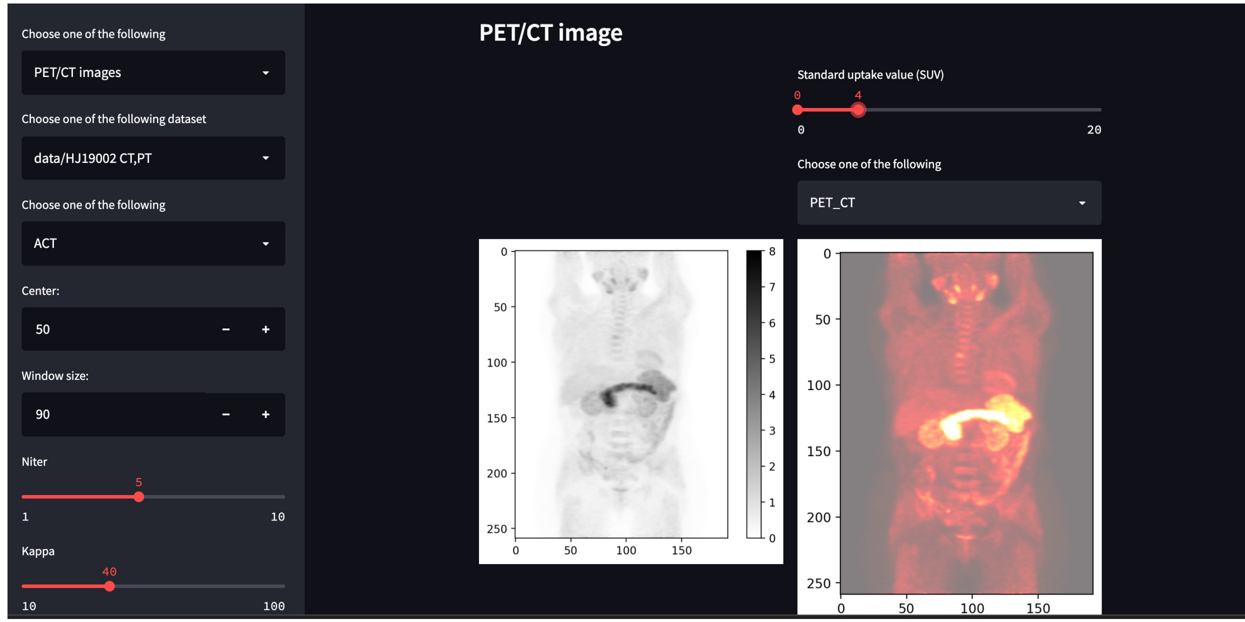


Showing the information of patients



1.This function allow to choose **PET-ACT, PET-FDG** or **CT** images
2. Adding window size to adjust **Hounsfield Units**





Supplementation

Adding data public (MICCAI)

Pages / Wiki / Collections

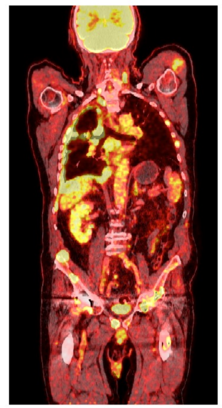
A whole-body FDG-PET/CT dataset with manually annotated tumor lesions (FDG-PET-CT-Lesions)

Created by Brittney Camp, last modified by Tracy Nolan on Oct 04, 2022

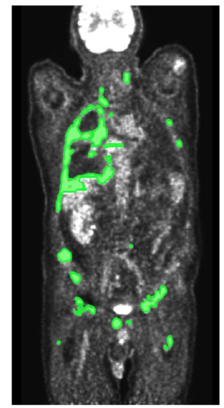
Summary

Purpose: To provide an annotated data set of oncologic PET/CT studies for the development and training of machine learning methods and to help address the limited availability of publicly available high-quality training data for PET/CT image analysis projects. This data can also be used for machine learning challenges, which is exemplified in the autoPET MICCAI 2022 competition: https://autopet.grand-challenge.org/.

Data: The anonymized publication of data was approved by the local ethics committee and data protection officer. 501 consecutive whole body FDG-PET/CT data sets of patients with malignant lymphoma, melanoma and non small cell lung cancer (NSCLC) as well as 513 data sets without PET-positive malignant lesions (negative controls) examined between 2014 and 2018 at the University Hospital Tübingen were included. All examinations were acquired on a single, state-of-the-art PET/CT scanner (Siemens Biograph mCT). The imaging protocol consists of a diagnostic CT scan (mainly from skull base to mid-thigh level) with intravenous contrast enhancement in most cases, except for patients with contraindications. The following CT parameters were used: reference dose of 200 mAs, tube voltage of 120 kV, iterative reconstruction with a slice thickness of 2 - 3 mm. In addition, a whole-body FDG-PET scan was acquired 60 minutes after I.V. injection of 300-350 MBq 18F-FDG. PET data were reconstructed using an ordered-subset expectation maximization (OSEM) algorithm with 21 subsets and 2 iterations and a gaussian kernel of 2 mm and a matrix size of 400 x 400.



whole-body
FDG-PET/CT



manual segmentation
of FDG-avid

⌛ Training and test cohort

Training cases: 1,014 studies (900 patients)

Test cases (final evaluation): 200 studies

Test cases (preliminary evaluation): 5 studies

A case (training or test case) consists of one 3D whole body FDG-PET volume, one corresponding 3D whole body CT volume and one 3D binary mask of manually segmented tumor lesions on FDG-PET of the size of the PET volume. CT and PET were acquired simultaneously on a single PET/CT scanner in one session; thus PET and CT are anatomically aligned up to minor shifts due to physiological motion.

Training set

Training data consists of 1,014 studies acquired at the University Hospital Tübingen and is made publicly available on TCIA (as DICOM, NiFTI and HDF5 files).

Data is available on TCIA: DOI 10.7937/gkr0-xv29

Data pre-processing and structure

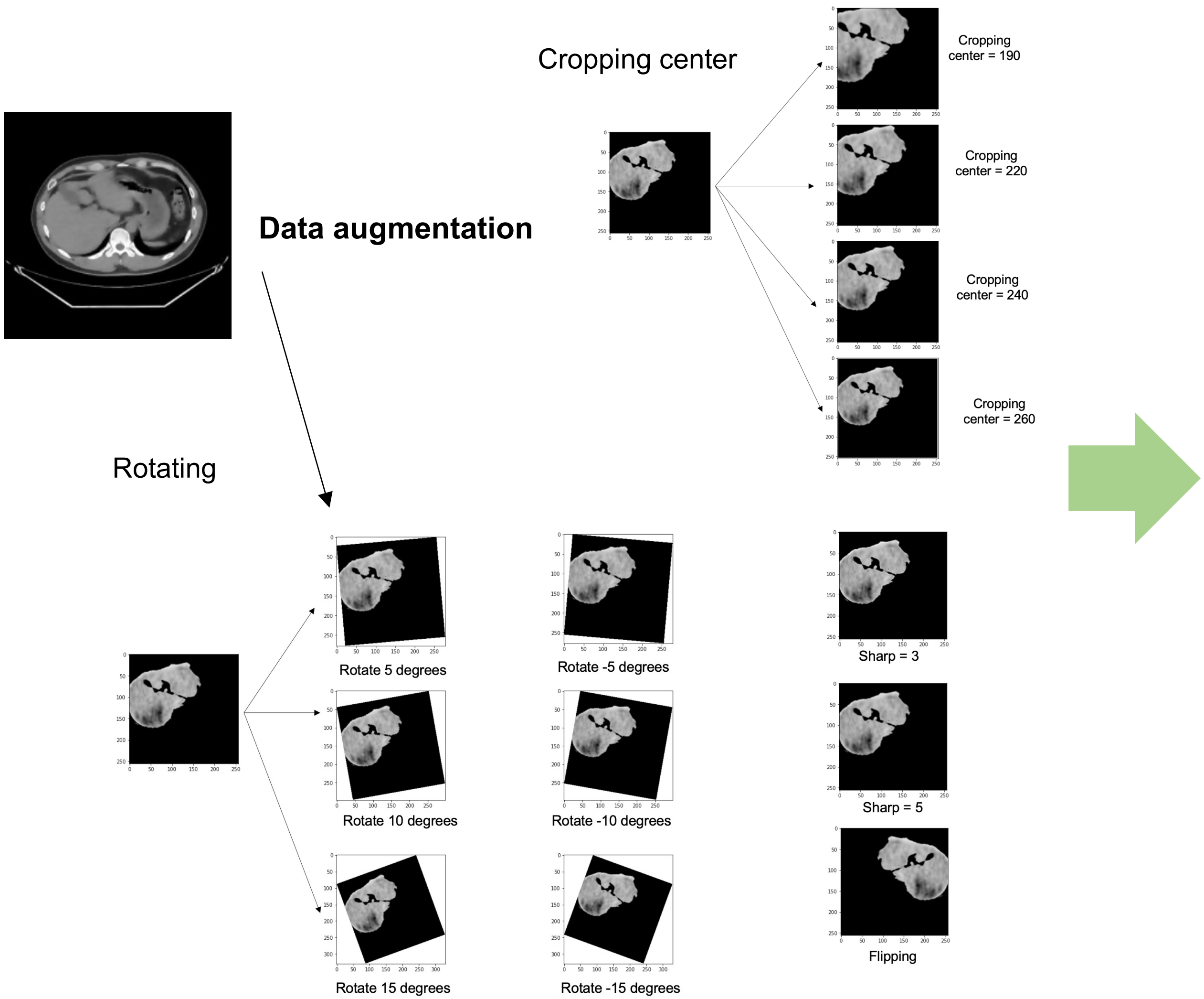
In a pre-processing step, the TCIA DICOM files are resampled (CT to PET imaging resolution, i.e. same matrix size) and normalized (PET converted to standardized update values; SUV).

For the challenge, the pre-processed data will be provided in NifTI format. PET data is standardized by converting image units from activity counts to standardized uptake values (SUV). We recommend to use the resampled CT (`CTres.nii.gz`) and the PET in SUV (`SUV.nii.gz`). The mask (`SEG.nii.gz`) is binary with `1` indicating the lesion. The training and test database have the following structure.

NiFTI

```
|--- Patient 1
     |--- Study 1
          |--- SUV.nii.gz    (PET image in SUV)
          |--- CTres.nii.gz  (CT image resampled to PET)
          |--- CT.nii.gz     (Original CT image)
          |--- SEG.nii.gz    (Manual annotations of tumor lesions)
          |--- PET.nii.gz    (Original PET image as actictivity counts)
     |--- Study 2            (Potential 2nd visit of same patient)
          |--- ...
|--- Patient 2
     |--- ...
```

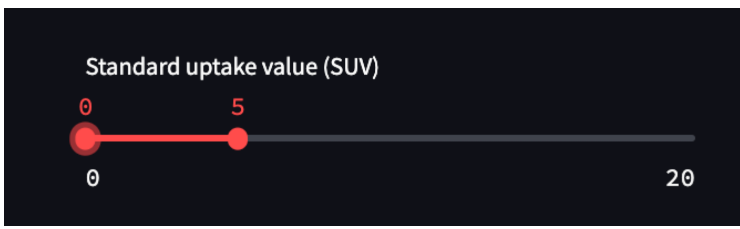
Data Augmentation

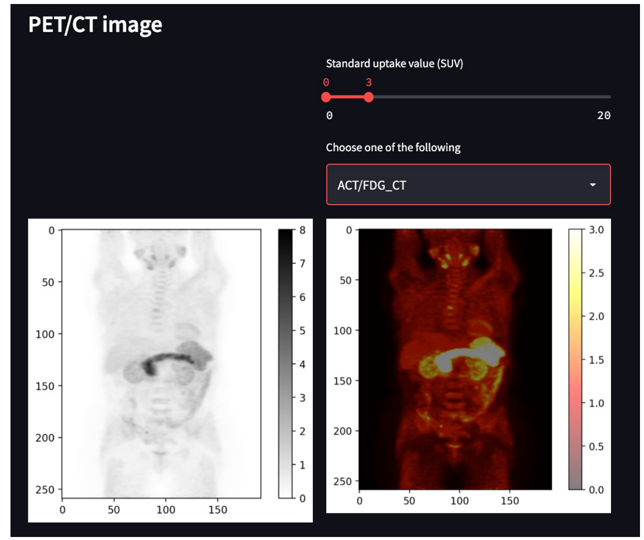


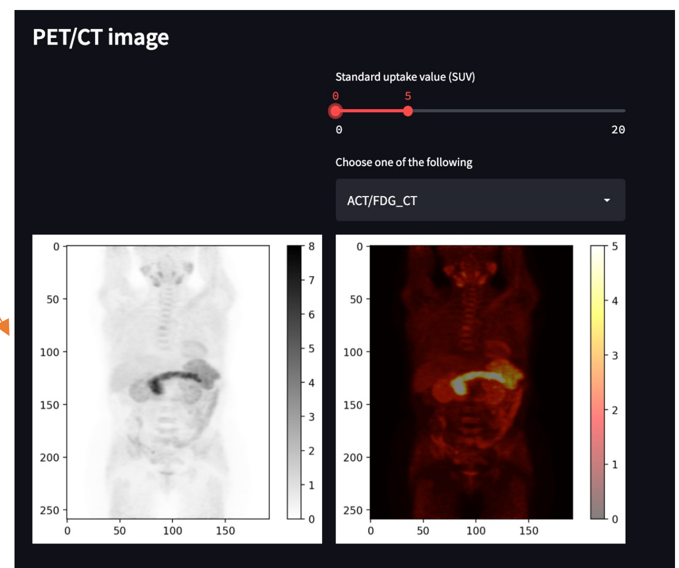
Up-creasing this volume datasets to 3-5 times. This process avoid bias in training stage

GUI tool document

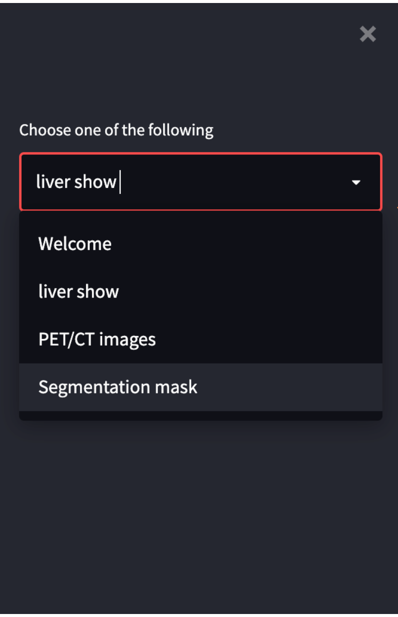
Bar width to adjust Standard uptake values (SUV)



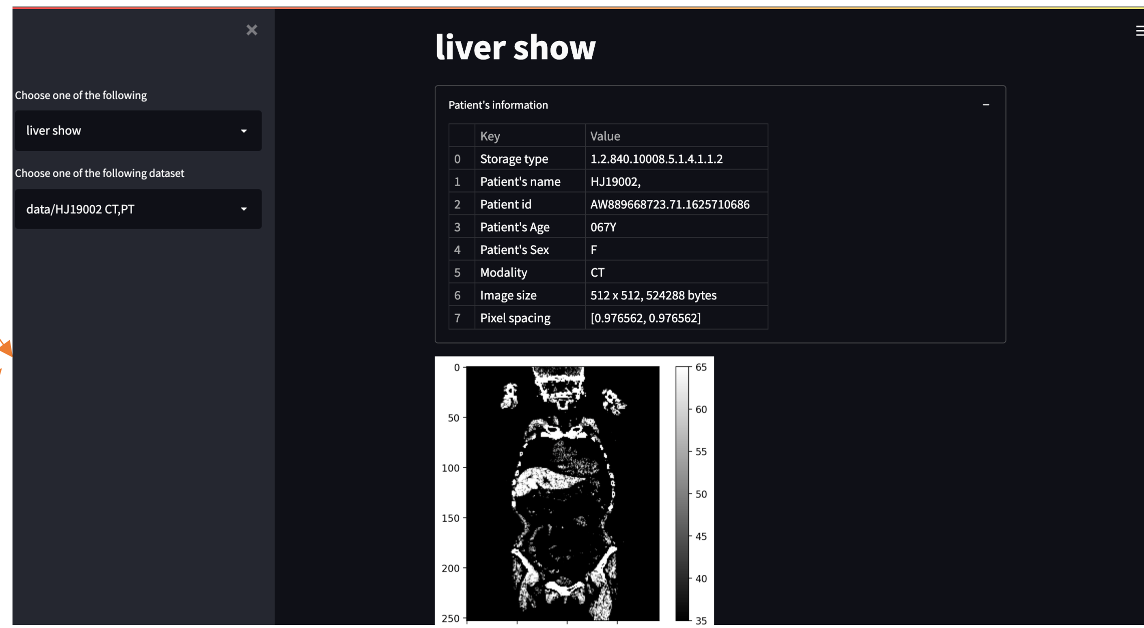


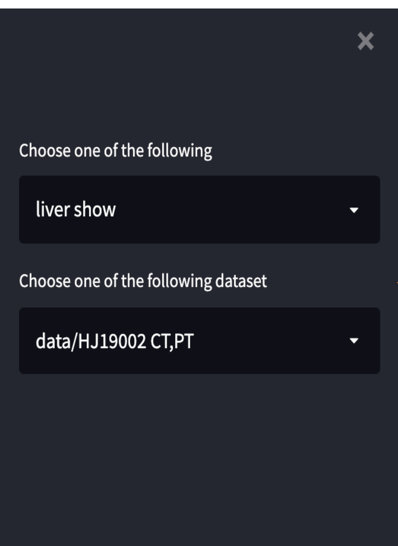


GUI tool document

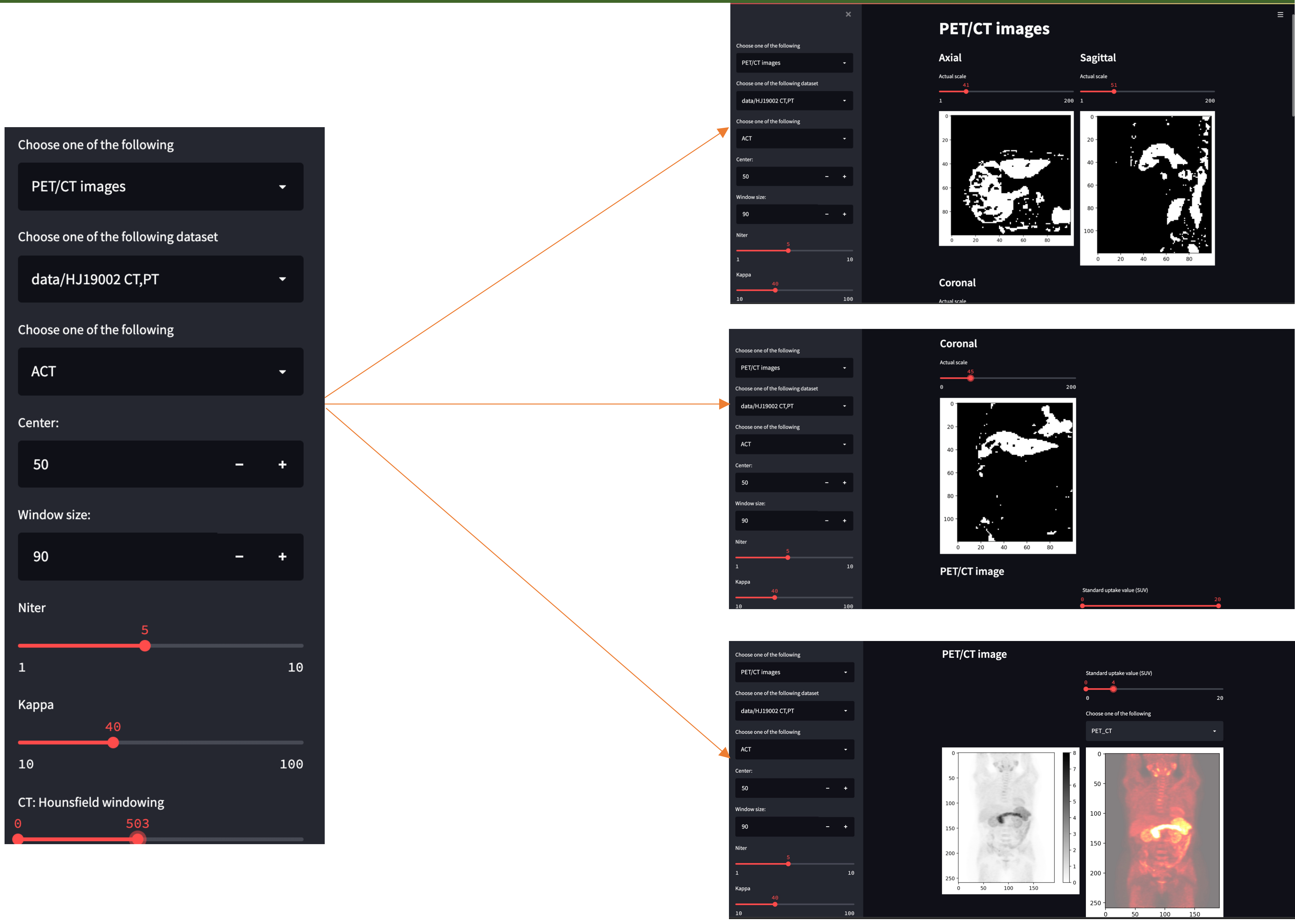


The same with GUI tool for Bone. The interface when show liver segmentation also similar

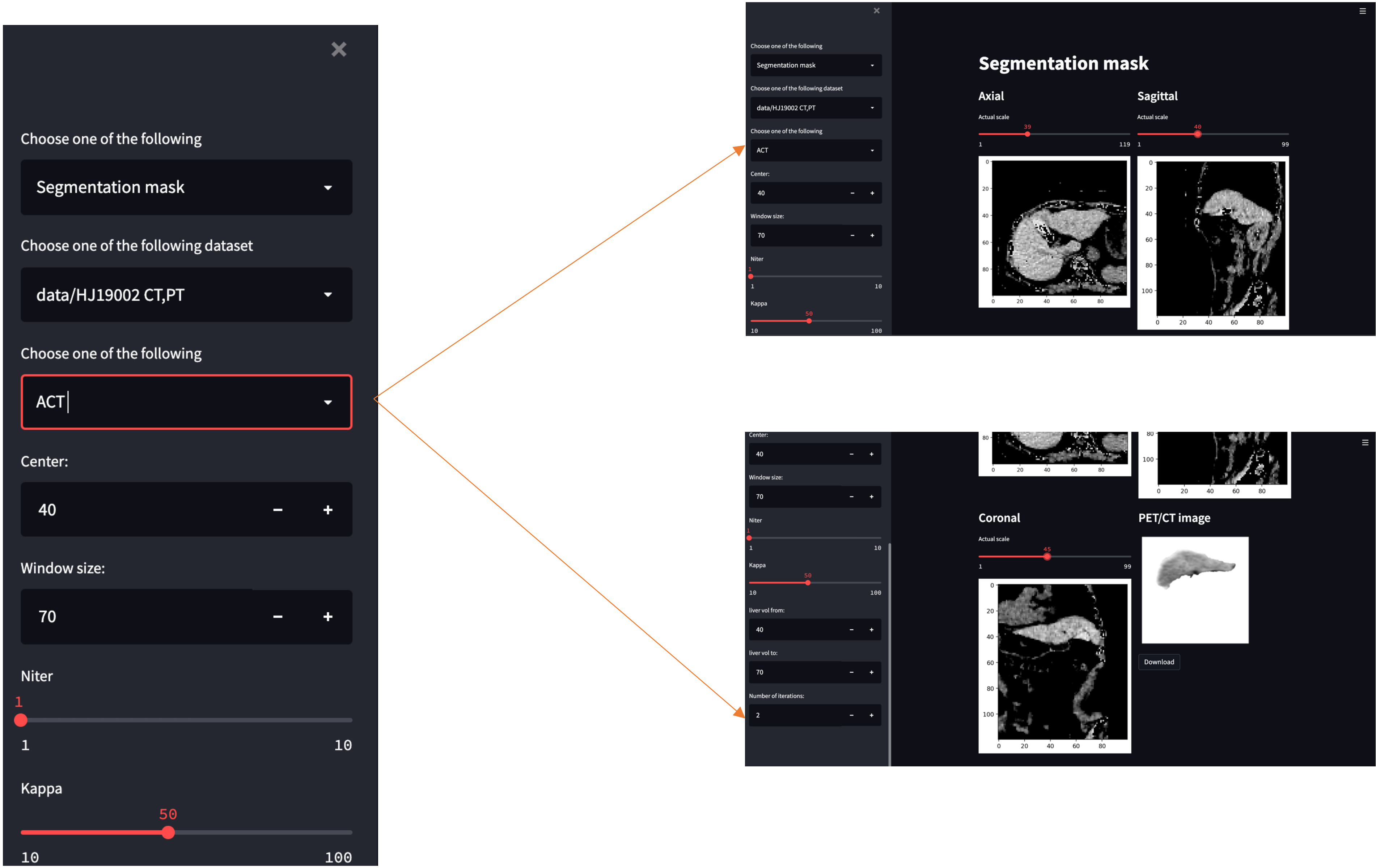




GUI tool document



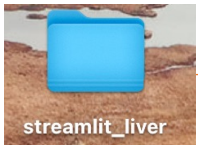
GUI tool document



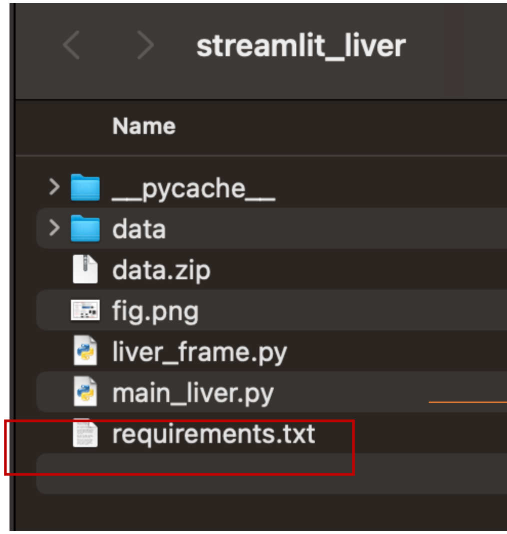
File and environment to set up for GUI

Requirement

(1)



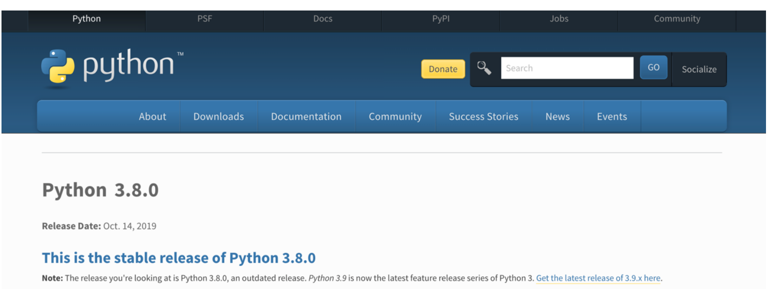
Detail package code



Library (Pip install these library in Python)

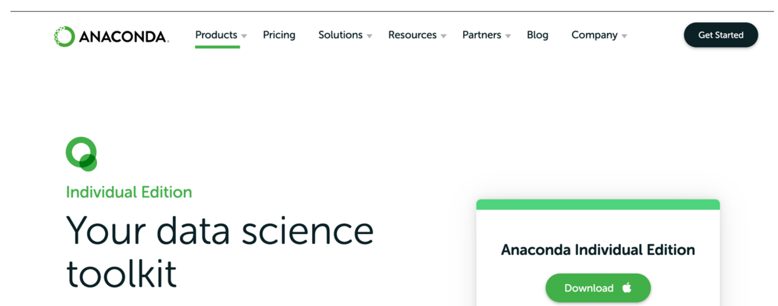
```
streamlit~=1.5.0
pydicom~=2.2.2
numpy~=1.22.1
ipywidgets~=7.6.5
Pillow~=9.0.0
ipython~=8.0.1
matplotlib~=3.5.1
ipyvolume~=0.5.2
sklearn~=0.0
scipy~=1.7.3
scikit-image~=0.19.1
plotly~=5.5.0
SimpleITK~=2.1.1
scikit-learn~=1.0.2
pandas
```

(2) Python version >= 3.



Link download:
https://www.python.org/downloads/release/python-380/

(3) Anaconda



https://www.anaconda.com/products/individual

File and environment to set up for GUI

To detail for step to install and setup Python, Streamlit and Anaconda
Run package code on terminal by command line

1.Set up and install anaconda detail in this link: https://docs.anaconda.com/anaconda/install/windows/

2.Set up python detail in this link: https://realpython.com/installing-python/

3. Create enviroment ”Streamlit”: https://docs.streamlit.io/library/get-started/installation

3.1 Navigate to “streamlit_liver” folder and type command line on terminal as below

cd streamlit_liver

3.2. Create a new Pipenv environment in that folder and activate that environment by command line:

```
pipenv shell
```

3.3. Install Streamlit in your environment:

```
pip install streamlit
```

3.4. Test that the installation worked:

```
streamlit hello
```

The end!