Artificial Intelligence Apps for Medical Image Analysis using pyCERR and Cancer Genomics Cloud

Aditya P. Apte, Eve LoCastro, Aditi Iyer, Sharif Elguindi, Jue Jiang, Jung Hun Oh, Harini Veeraraghavan, Amita Shukla-Dave, Joseph O. Deasy

Department of Medical Physics, Memorial Sloan Kettering Cancer Center, New York, USA

Abstract This work introduces a user-friendly, cloud-based software framework for conducting Artificial Intelligence (AI) analyses of medical images. The framework allows users to deploy AI-based workflows by customizing software and hardware dependencies. The components of our software framework include the Python-native Computational Environment for Radiological Research (pyCERR) platform for radiological image processing, Cancer Genomics Cloud (CGC) for accessing hardware resources and user management utilities for accessing images from data repositories and installing AI models and their dependencies. GNU-GPL copyright pyCERR was ported to Python from MATLAB-based CERR to enable researchers to organize, access, and transform metadata from high dimensional, multi-modal datasets to build cloud-compatible workflows for AI modeling in radiation therapy and medical image analysis. pyCERR provides an extensible data structure to accommodate metadata from commonly used medical imaging file formats and a viewer to allow for multi-modal visualization. Analysis modules are provided to facilitate cloud-compatible AI-based workflows for image segmentation, radiomics, DCE MRI analysis, radiotherapy dose-volume histogram-based features, and normal tissue complication and tumor control models for radiotherapy. Image processing utilities are provided to help train and infer convolutional neural network-based models for image segmentation, registration and transformation. The framework allows for round-trip analysis of imaging data, enabling users to apply AI models to their images on CGC and retrieve and review results on their local machine without requiring local installation of specialized software or GPU hardware. The deployed AI models can be accessed using APIs provided by CGC, enabling their use in a variety of programming languages. In summary, the presented framework facilitates end-to-end radiological image analysis and reproducible research, including pulling data from sources, training or inferring from an AI model, utilities for data management, visualization, and simplified access to image metadata.

1 Introduction

This work addresses the need for a comprehensive cloud-based software framework for medical image analysis. Data and AI models are persistently hosted in linked cloud buckets from providers such as AWS and GCP or pulled in from external cloud repositories, including public sources such as TCIA[3] and IDC[4] as well as proprietary sources such as XNAT[7], BOX (https://www.box.com). Cancer Genomics Cloud (CGC)[8] provides integrated bioinformatics tools, automatic logging, versioning, and access-controlled sharing, facilitating reproducibility and collaboration. Figure 1 shows various components of the cloud-based framework for end-to-end AI workflows for medical image analysis.

The Computational Environment for Radiological Research (CERR) platform[12], developed in the MATLAB programming language has been widely used and adapted for various medical physics applications over the past decade due to its robust and user-friendly data structure and visualization capabilities. These include prototyping Intensity Modulated Radiation Therapy (IMRT)

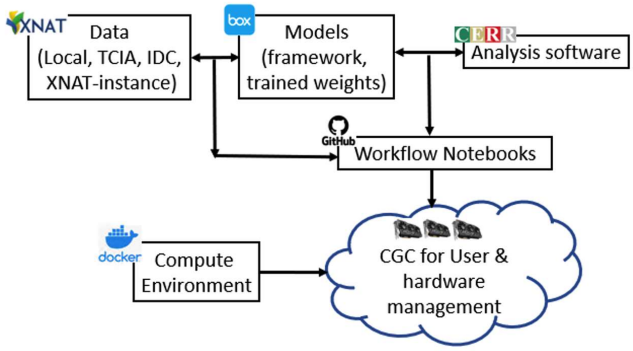


Figure 1: Components of cloud-based analyses for medical imges

optimization algorithms[13], outcomes modeling[14, 15], film dosimetry QA[16], quality assurance of RT[17] and imaging data by leading institutions, and CNN-based training and inference[14]. CERR's flexible data structure has been extended by several plugins such as segmentation benchmarking software developed by AAPM TG 211[15]. While CERR has been highly useful in clinical research, its adoption is limited by the requirement of MATLAB license for development. This is especially crucial in cloud-environments where open-source programming languages offer license-free use and readability of the underlying codebase. Python has emerged as the language of choice for data analysis and research, serving as a glue language for various software libraries. Python's scientific computing and visualization libraries now offer functionalities that are on par or surpass MATLAB and its toolboxes for medical image processing.

pyCERR maintains the flexible, readable data structure and graphical user interfaces of the CERR platform while providing relevant tools in the Python programming environment. This allows pyCERR to seamlessly integrate with the medical image analysis ecosystem and software such as XNAT, 3DSlicer, ANTPy, etc. enabling access to cutting-edge AI modeling and image processing libraries for local and cloud-friendly workflows. pyCERR is used in clinical deployment of AI segmentation models at our

institution which have been made publicly available for research use

We describe various AI-based apps developed using pyCERR and deployed on the CGC. This framework fosters collaborative research through persistent hosting of data, code, and analysis tools on the cloud. It enhances the medical image analysis ecosystem by using leveraging data archival platforms, libraries to train and infer AI models, cloud providers and Python APIs from popular image analysis libraries[18, 19]. The next sections describe the architecture and various modules of our cloud-based framework in detail.

2 Materials and Methods

(i) Data I/O:

Data analysis workflows require radiological inputs (images, doses, segmentations) and AI models. Radiological images are hosted in publicly available repositories such as TCIA, IDC, Zenodo (https://zenodo.org) which provide Python APIs to query/retrieve images. We have hosted an instance of XNAT ("PIXNAT") at our institution to archive images for research use, accessible at https://pixnat.mskcc.org. It facilitates the sharing of IRB-approved project datasets with external collaborators. XNAT provides APIs (pyXNAT, XNATpy) to query, retrieve, and upload images.

Trained AI models consist of the network architecture and associated weights. Depending on the network architecture, the weight files can be substantial in size. We archive the trained network along with its dependencies in the form of a packaged Anaconda environment. The network and weights are stored in our institution's BOX platform with publicly available static links for direct download.

Additionally, cloud providers such as GCP and AWS provide data storage which can be accessed programmatically. Users are charged for hosting as well as egress to move data to and from these buckets.

(ii) Analysis software:

The components of pyCERR can be broadly categorized as (1) Data container, (2) Viewer, and (3) Analysis modules. pyCERR builds on commonly used scientific libraries Numpy, scipy, pandas and image processing libraries such as pydicom, SimpleITK, and scikit-image. pyCERR uses Napari for visualization via graphical user interface objects within the Python environment.

(a) Metadata container: The central idea of pyCERR is to provide a consistent data structure for metadata from different file formats. The plan container class "planC" houses data classes containing attributes for various data types. These include (a) Scan for imaging data such as CT, MR, PET, (b) Structure for image segmentation, (c) Dose for RT dose, and (d) Beams for RT plans as well as derived data types such as for deformable registration. Figure 1 shows a schematic representation of data classes, including key attributes and routines for data access and transformation.

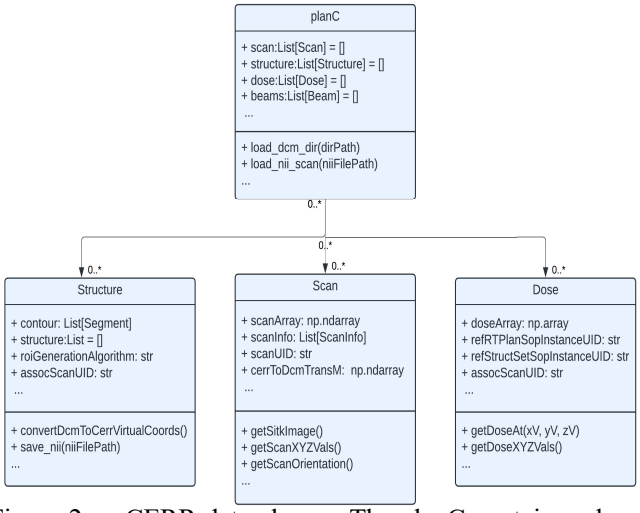


Figure 2: pyCERR data classes. The planC container class is extensible and composed of different radiological data types.

Modules are provided for importing and exporting imaging data between DICOM and NifTI formats as well as conversion to SimpleITK[20] Image objects. Pydicom[21] is used to read and write DICOM, Nibabel[22] to write to NifTI, and SimpleITK to read ITK-compatible formats such as NifTI, MHA, and NRRD. Data attributes in the planC container can be readily accessed for analyses, edited, and saved to DICOM and NifTI file formats.

pyCERR currently imports metadata from DICOM RTSTRUCT and SEG segmentation modalities, scan modalities CT, MR, PET and radiotherapy dose modalities RTDOSE and RTPLAN. Segmentations are stored as polygonal contours and can be converted to binary masks and vice-versa. pyCERR provides utilities to convert PET images to SUV[23] units using normalizations such as Body Weight, Body Surface Area, and Lean Body Mass approaches. pyCERR supports dynamic scan sequences such as DCE and diffusion weighted MRI, with image volumes are organized by acquisition time or b-value. MRI images are converted to real world units using vendor-specific private tags.

(b) Visualization: Multi-dimensional image viewer Napari[24] is perfectly suited for pyCERR's multi-modal data structure. As Napari is written in, it eliminates the need for users to transfer data between viewers written in other languages, such as ImageJ[25] and 3D Slicer[26]. The Napari ecosystem provides a rapidly-growing list of plugins for image analysis, such as formicroscopy datasets and the Segment Anything Model[27]. To our knowledge, pyCERR is the only software platform that enables visualization of radiotherapy data in Napari. The viewer module in pyCERR provides routines to visualize scan, segmentation, and dose data types. It provides widgets to window scans and to segment them by leveraging Napari's labeling tools.

(c) Analysis modules: Organization of imaging data in a plan container enables efficient access to the underlying data elements and their relationships for building analysis modules. Modules for data transformation, radiomics, dosimetric analysis, and segmentation evaluation are included with pyCERR. Data transformations include tools for post-processing segmentations, such as smoothing contour polygons, connected components filtering, resampling and transferring segmentations and doses from one scan grid to another. The radiomics module in pyCERR provides scalar features and texture filters consistent with their IBSI definitions. Feature and filter computations are entirely Python-based, using vectorized code via Numpy arrays for speed. Implementations were validated against IBSI benchmarks. IBSI2 Filter response maps are stored in planC, which facilitates visualization and analysis. pyCERR provides tools to extract dose-volume histogram (DVH)-based features such as Vx, Dx, MOHx, MOCx.

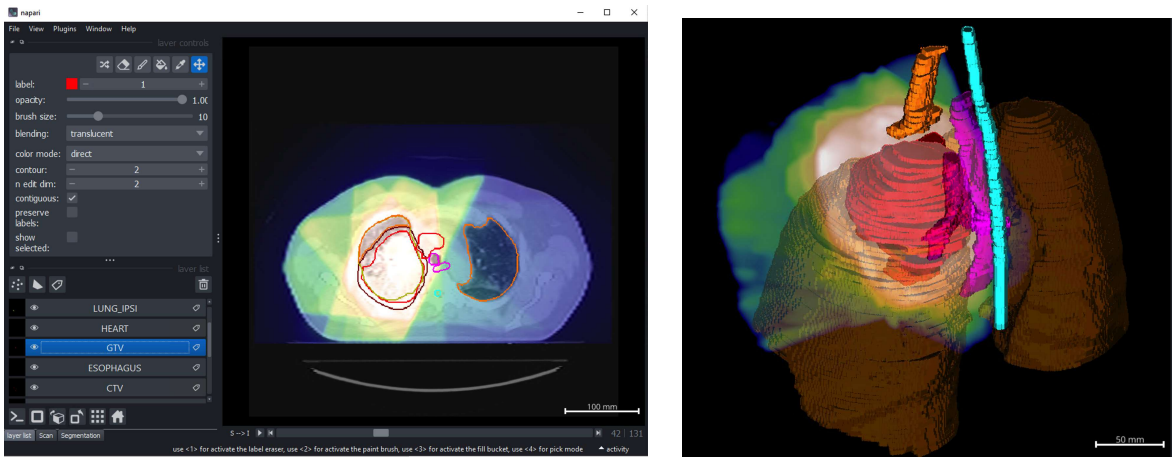
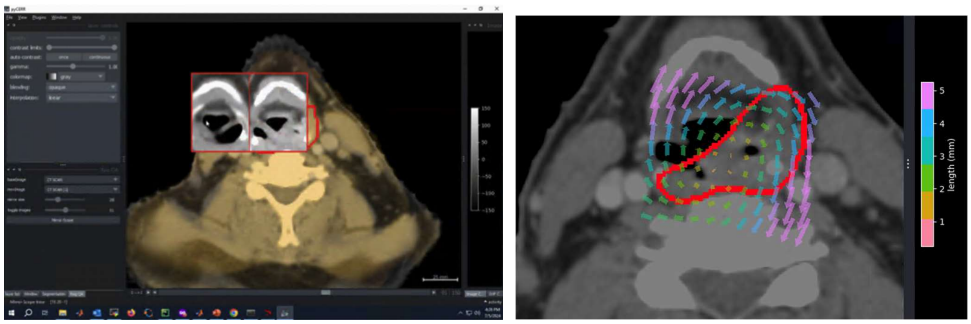


Figure 3: Radiotherapy does distributions can be visualized along with structures in 2D and 3D. Display settings such as Window, transparency, gamma can be adjusted from GUI or command, thus making it suitable for capturing screenshots in batch mode for large number of datasets.

Table 1: Pre-trained AI models

Site	**Modality**	**Architecture**	**Function**
Lung	CT	Multiple Resolution Residual Network[1]	OAR segmentation
		DeepLabV3+[2]	
Lung	CT	Self-distilled Masked Image Transformer[5]	Lung tumor segmentation
Head and neck	CT	DeepLabV3+[6]	OAR segmentation
		Self-attention[9]	
Prostate	T2w MRI	DeepLabV3+[10]	OAR segmentation
Pancreas	T2w MRI	ProRSeg[11]	Longitudinal deformable registration, OAR segmentation



(a) (b)

Figure 4: Mirrorscope and DVH display

(iii) User management and sharing:

User data and workflows are organized into projects on CGC; they can be kept private or made available for sharing with other users on the platform. Users create and share analysis workflows as "apps" that are launched in user-specified Docker environments. Apps are defined in Common Workflow Language (CWL); multiple CWL tasks can be chained together in a workflow definition. These workflows can be initiated as tasks in the cloud within the CGC-SB web portal or externally from users' local hardware using their API via Python or command line.

CGC-SB Interactive Data Studio environments include Jupyter, RStudio, SAS Studio and Galaxy. The OHIF [ref] viewer is also available for reviewing images in the project. CGC Workflow apps and Data Studio operate on AWS instances with specific CPU, RAM or GPU requirements configured by user.

Figure 5: Contouring tools in pyCERR. The segmentation widget provides the ability to create and edit structures. This is useful to edit contours resulting from auto-segmentation as well as to generate new structures. Napari provides Paintbrush, Eraser and Fill tools for editing segmentation labels.

3 Results

AI workflows are organized as a sharable project on CGC platform. Pretrained AI models are provided for segmentation of normal tissues across multiple sites. Table 1 lists models available on the platform. These can be accessed via CGC Seven Bridges API or via an interactive Jupyter notebook. A locally runnable Jupyter notebook demonstrating API call to heart sub-structure segmentation model is distributed at https://github.com/cerr/pyCERR-Notebooks/blob/main/SBG_autosegment_CT_Heart_OARs.ipynb.

The pyCERR package is distributed under an open-source, GNU GPL license at https://github.com/cerr/pyCERR. pyCERR can be installed from the GitHub repository by using Python package installer pip. Tests for various modules including IBSI-compliant radiomics, and conversion of data between various formats are distributed with pyCERR. These are run upon every code update to the GitHub repository. Notebooks demonstrating various use cases are provided at https://github.com/cerr/pyCERR-

```python
import os
from cerr import plan_container as pc
from cerr.radiomics import ibsi1

featList = []
idDict = {}
writeHeader = True
for pt_dir in all_pat_dirs[:5]:
    _, id = os.path.split(pt_dir)
    print("Data dir :" + id)
    planC = pc.load_dcm_dir(pt_dir)
    scanNum = 0
    structNum = 0
    featDict = ibsi1.computeScalarFeatures(scanNum, structNum, settingsFile, planC)
    idDict['id'] = id
    featDict = {**idDict, **featDict}
    featList.append(featDict)
    ibsi1.writeFeaturesToFile(featDict, csvFileName, writeHeader)
    writeHeader = False
```

```python
from cerr import dvh
structNum = 0
doseNum = 0
dosesV, volsV, isErr = dvh.getDVH(structNum, doseNum, planC)
binWidth = 0.025
doseBinsV,volHistV = dvh.doseHist(dosesV, volsV, binWidth)
percent = 70
dvh.MOHx(doseBinsV,volHistV,percent)
```

Figure 6: (a) Radiomics feature extraction for a batch of datasets: The ***radiomics*** module provides the flexibility of selecting scan and segmentation objects from planC and specifying calculation settings via JSON (b) DVH feature extraction: The ***dvh*** module provides routines to obtain RT dose within the segmentation, accumulation of dose values into histograms and calculation of scalar metrics.

Notebooks to provide a starting point for analyses. Following are some common use cases:

(1) Re-Radiation / Adaptive RT, dose accumulation:
pyCERR's data structure supports multi-modal and longitudinal imaging data, facilitating dose accumulation for re-radiation and adaptive RT by transferring segmentations and dose distributions between different time-points via image registration. Visualization tools are provided for quality assurance of image registration as well as deformation vector field (DVF). Figure 3 shows an example of mirrorscope and DVF display.

(2) Visualizing a batch of datasets:
pyCERR's viewer module provides routines to overlay imaging data from the planC container such as scans, doses and structures as demonstrated in figure 3. These routines can be used locally in an interactive mode or on a remote server in batch-mode; The notebook at https://github.com/cerr/pyCERR-Notebooks/blob/main/batch_visualize_scan_seg_ex1.ipynb demonstrates capturing screenshots for a batch of DICOM datasets on Google Colab platform.

(3) Image segmentation:
The cloud-based framework provides well-validated tools for normal tissue segmentation for various radiotherapy treatment sites as listed in Table 1. Additionally, pyCERR provides a manual segmentation widget supports multiple orthogonal views, overlaying image modalities such as PET and CT, as demonstrated in Figure 4. This can be used to generate ground-truth data for model training as well as editing the output of AI segmentation for downstream use.

(4) Radiomics and DVH-based metrics calculation:
pyCERR provides "*radiomics*" and "*dvh*" modules to support feature extraction from scan and dose datasets. The *radiomics* module provides routines for IBSI-1 and IBSI-2 compatible feature extraction. The dvh module provides an ability to compute commonly used metrics with user defined dose bin width.

4 Discussion

The presented framework simplifies the use of AI models. Data processing is pre-configured using pyCERR, and hardware dependencies are easily setup on the CGC, extending access to users with minimal programming experience. Moreover, users can readily combine imaging with genomics data available on CGC.
Napari is a relatively new project compared to more established viewers such as ITK-SNAP[28] and 3DSlicer. While the underlying Napari features are undergoing continuous development, it is the only Python-based viewer with a rich feature set. Napari, as a generic image viewer, supports visualization of various data types such as shapes, vectors, labels, images and allows for defining custom callback events. This enables users to customize pyCERR's visualization tools to suit their specific needs. Additionally, pyCERR will benefit from future development in Napari, including support for orthogonal view planes, grouping of display layers, and viewing remote datasets.

5 Conclusion

The framework for deploying AI apps presented in this work is accomplished without local setup or installation required in basic command line tools to remotely launch apps on the CGC platform. pyCERR provides comprehensive tools for radiological and radiotherapy data

processing, leveraging open-source formats and libraries. Users are provided with a shareable project on CGC to quickly get started on using the presented analyses. The tools and Jupyter notebooks are self-contained such that they can also be run on institutional HPCs or other cloud platforms.

Disclosure of Conflicts of Interest

The authors have no relevant conflicts of interest to disclose.

Acknowledgements

This study was supported in part by the MSK Cancer Center Support grant (NIH grant P30 CA008748), The Simons Foundation, the Breast Cancer Research Foundation (grant MATH-23-001), and the NIH ROBIN cooperative group (grant U54CA274291). We also acknowledge in-kind support for cloud computation from Seven Bridges Cancer Research Data Commons Cloud.

The Seven Bridges Cancer Research Data Commons Cloud Resource has been funded in whole or in part with Federal funds from the National Cancer Institute, National Institutes of Health, Contract No. HHSN261201400008C and ID/IQ Agreement No. 17X146 under Contract No. HHSN261201500003I and 75N91019D00024.

References

1. Jiang J, Hu YC, Liu CJ, Halpenny D, Hellmann MD, Deasy JO, Mageras G, Veeraraghavan H. Multiple Resolution Residually Connected Feature Streams for Automatic Lung Tumor Segmentation From CT Images. IEEE Trans Med Imaging. 2019;38(1):134-44. Epub 20180723. doi: 10.1109/TMI.2018.2857800. PubMed PMID: 30040632; PMCID: PMC6402577.
2. Haq R, Hotca A, Apte A, Rimner A, Deasy JO, Thor M. Cardio-pulmonary substructure segmentation of radiotherapy computed tomography images using convolutional neural networks for clinical outcomes analysis. Phys Imaging Radiat Oncol. 2020;14:61-6. Epub 20200610. doi: 10.1016/j.phro.2020.05.009. PubMed PMID: 33458316; PMCID: PMC7807536.
3. Clark K, Vendt B, Smith K, Freymann J, Kirby J, Koppel P, Moore S, Phillips S, Maffitt D, Pringle M, Tarbox L, Prior F. The Cancer Imaging Archive (TCIA): maintaining and operating a public information repository. J Digit Imaging. 2013;26(6):1045-57. doi: 10.1007/s10278-013-9622-7. PubMed PMID: 23884657; PMCID: PMC3824915.
4. Fedorov A, Longabaugh WJR, Pot D, Clunie DA, Pieper S, Aerts H, Homeyer A, Lewis R, Akbarzadeh A, Bontempi D, Clifford W, Herrmann MD, Hofener H, Octaviano I, Osborne C, Paquette S, Petts J, Punzo D, Reyes M, Schacherer DP, Tian M, White G, Ziegler E, Shmulevich I, Pihl T, Wagner U, Farahani K, Kikinis R. NCI Imaging Data Commons. Cancer Res. 2021;81(16):4188-93. Epub 20210615. doi: 10.1158/0008-5472.CAN-21-0950. PubMed PMID: 34185678; PMCID: PMC8373794.
5. Jiang J, Tyagi N, Tringale K, Crane C, Veeraraghavan H. Self-supervised 3D anatomy segmentation using self-distilled masked image transformer (SMIT). Med Image Comput Comput Assist Interv. 2022;13434:556-66. Epub 20220916. doi: 10.1007/978-3-031-16440-8_53. PubMed PMID: 36468915; PMCID: PMC9714226.
6. Iyer A, Thor M, Onochie I, Hesse J, Zakeri K, LoCastro E, Jiang J, Veeraraghavan H, Elguindi S, Lee NY, Deasy JO, Apte AP. Prospectively-validated deep learning model for segmenting swallowing and chewing structures in CT. Phys Med Biol. 2022;67(2). Epub 20220117. doi: 10.1088/1361-6560/ac4000. PubMed PMID: 34874302; PMCID: PMC8911366.
7. Marcus DS, Olsen TR, Ramaratnam M, Buckner RL. The Extensible Neuroimaging Archive Toolkit: an informatics platform for managing, exploring, and sharing neuroimaging data. Neuroinformatics. 2007;5(1):11-34. doi: 10.1385/ni:5:1:11. PubMed PMID: 17426351.
8. Pot D, Worman Z, Baumann A, Pathak S, Beck R, Beck E, Thayer K, Davidsen TM, Kim E, Davis-Dusenbery B, Otridge J, Pihl T, Program C, Barnholtz-Sloan JS, Kerlavage AR. NCI Cancer Research Data Commons: Cloud-Based Analytic Resources. Cancer Res. 2024;84(9):1396-403. doi: 10.1158/0008-5472.CAN-23-2657. PubMed PMID: 38488504; PMCID: PMC11063685.
9. Jiang J, Elguindi S, Berry SL, Onochie I, Cervino L, Deasy JO, Veeraraghavan H. Nested block self-attention multiple resolution residual network for multiorgan segmentation from CT. Med Phys. 2022;49(8):5244-57. Epub 20220608. doi: 10.1002/mp.15765. PubMed PMID: 35598077; PMCID: PMC9908007.
10. Elguindi S, Zelefsky MJ, Jiang J, Veeraraghavan H, Deasy JO, Hunt MA, Tyagi N. Deep learning-based auto-segmentation of targets and organs-at-risk for magnetic resonance imaging only planning of prostate radiotherapy. Phys Imaging Radiat Oncol. 2019;12:80-6. Epub 20191212. doi: 10.1016/j.phro.2019.11.006. PubMed PMID: 32355894; PMCID: PMC7192345.
11. Jiang J, Hong J, Tringale K, Reyngold M, Crane C, Tyagi N, Veeraraghavan H. Progressively refined deep joint registration segmentation (ProRSeg) of gastrointestinal organs at risk: Application to MRI and cone-beam CT. Medical Physics. 2023;50(8):4758-74. doi: 10.1002/mp.16527. PubMed PMID: WOS:000998838700001.
12. Deasy JO, Blanco AI, Clark VH. CERR: a computational environment for radiotherapy research. Med

Phys. 2003;30(5):979-85. doi: 10.1118/1.1568978. PubMed PMID: 12773007.
13. Deasy J, Lee EK, Bortfeld T, Langer M, Zakarian K, Alaly J, Zhang Y, Liu H, Mohan R, Ahuja R, Pollack A, Purdy J, Rardin R. A collaboratory for radiation therapy treatment planning optimization research. Ann Oper Res. 2006;148(1):55-63. doi: 10.1007/s10479-006-0079-7. PubMed PMID: WOS:000242603100004.
14. Apte AP, Iyer A, Thor M, Pandya R, Haq R, Jiang J, LoCastro E, Shukla-Dave A, Sasankan N, Xiao Y, Hu YC, Elguindi S, Veeraraghavan H, Oh JH, Jackson A, Deasy JO. Library of deep-learning image segmentation and outcomes model-implementations. Phys Med. 2020;73:190-6. Epub 20200501. doi: 10.1016/j.ejmp.2020.04.011. PubMed PMID: 32371142; PMCID: PMC8474066.
15. Shepherd T, Berthon B, Galavis P, Spezi E, Apte A, Lee J, Visvikis D, Hatt M, De Bernardi E, Das S. Design of a benchmark platform for evaluating PET-based contouring accuracy in oncology applications. European Journal of Nuclear Medicine and Molecular Imaging. 2012;39:S264-S.
16. Deasy J, Apte A. Data analysis tools for 3D dosimetry: the use of CERR as a platform to integrate and compare measurements and treatment planning information. Journal of Physics: Conference Series. 2010;250. doi: 10.1088/1742-6596/250/1/012085.
17. Ulin K, Yorty J, Hanusik R, Urie M, Bosch W, Apte A, Khullar D, Deasy J, Fitzgerald T. Use of CERR at the quality assurance review center to assess protocol compliance of radiation therapy treatment plans submitted in digital format. International Journal of Radiation Oncology, Biology, Physics. 2008;72(1):S673.
18. Lowekamp BC, Chen DT, Ibáñez L, Blezek D. The design of SimpleITK. Frontiers in neuroinformatics. 2013;7:45.
19. Tustison NJ, Cook PA, Holbrook AJ, Johnson HJ, Muschelli J, Devenyi GA, Duda JT, Das SR, Cullen NC, Gillen DL, Yassa MA, Stone JR, Gee JC, Avants BB. The ANTsX ecosystem for quantitative biological and medical imaging. Sci Rep. 2021;11(1):9068. Epub 20210427. doi: 10.1038/s41598-021-87564-6. PubMed PMID: 33907199; PMCID: PMC8079440.
20. B. C. Lowekamp DTC, L. Ibáñez, D. Blezek. The Design of SimpleITK. Front Neuroinform. 2013(7). doi: 10.3389/fninf.2013.00045.
21. Mason DL, et al, pydicom: An open source DICOM library. pydicom: An open source DICOM library.
22. Brett M, Markiewicz CJ, Hanke M, Côté M-A, Cipollini B, McCarthy P, Jarecka D, Cheng C, Halchenko Y, Cottaar M. nipy/nibabel: 3.2. 1. Zenodo. 2020.
23. Thie JA. Understanding the standardized uptake value, its methods, and implications for usage. J Nucl Med. 2004;45(9):1431-4. PubMed PMID: 15347707.
24. napari: a multi-dimensional image viewer for Python.
25. Abràmoff MD, Magalhães PJ, Ram SJ. Image processing with ImageJ. Biophotonics international. 2004;11(7):36-42.
26. Kikinis R PS, Vosburgh K. 3D Slicer: A Platform for Subject-Specific Image Analysis, Visualization, and Clinical Support. In: Jolesz F (eds) Intraoperative Imaging and Image-Guided Therapy. 2014. doi: https://doi.org/10.1007/978-1-4614-7657-3_19.
27. Ma J, He Y, Li F, Han L, You C, Wang B. Segment anything in medical images. Nature Communications. 2024;15(1):654.
28. Yushkevich PA, Gao Y, Gerig G, editors. ITK-SNAP: An interactive tool for semi-automatic segmentation of multi-modality biomedical images. 2016 38th annual international conference of the IEEE engineering in medicine and biology society (EMBC); 2016: IEEE.