applied sciences

MDPI

Article

SegMed: An Open-Source Desktop Tool for Deploying Pretrained Deep Learning Models in 3D Medical Image Segmentation

Mhd Jafar Mortada [1], Agnese Sbrollini [1], Klaudia Proniewska-van Dam [2], Peter M. Van Dam [3] and Laura Burattini [1,*]

[1] Department of Information Engineering, Università Politecnica delle Marche, 60131 Ancona, Italy; m.j.mortada@pm.univpm.it (M.J.M.); a.sbrollini@staff.univpm.it (A.S.)
[2] Center for Digital Medicine and Robotics, Jagiellonian University Medical College, 31-034 Krakow, Poland; klaudia.proniewska@uj.edu.pl
[3] Department of Radiology, Jagiellonian University Medical College, 31-501 Krakow, Poland; peter.dam@uj.edu.pl
* Correspondence: l.burattini@univpm.it

Featured Application

The proposed application provides a standalone desktop solution for deploying pretrained deep learning segmentation models on 3D medical images. By abstracting model configuration and inference details, SegMed simplifies the adoption of AI-based segmentation in medical imaging research and educational environments.

Abstract

Deep learning has become central to semantic segmentation of three-dimensional medical images. However—despite many published models—their adoption in practice remains limited, as deployment often requires advanced programming skills and familiarity with specific machine learning frameworks. Thus, technical barriers restrict its use to specialized users. To address this, we present SegMed (version 1.0), an open-source, standalone desktop application that provides an end-to-end workflow for deep learning-based medical image segmentation. SegMed supports the loading and inspection of common medical image formats, as well as array-based formats. The application integrates standard preprocessing operations often used in the field and directly supports loading of pretrained segmentation models implemented in both PyTorch (version 2.X) and Keras (version 2.X) and those created using the Medical Open Network for AI framework (version 1.X). Models are automatically inspected to infer required configurations, such as input size and post-processing steps, enabling segmentation with minimal user intervention. Results can be exported as volumetric images or 3D surface meshes for downstream analysis, visualization, or special applications such as virtual reality. SegMed was tested using multiple publicly available pretrained models, demonstrating robustness and flexibility across diverse segmentation tasks. By abstracting low-level implementation details, SegMed lowers technical barriers, promotes reproducibility, and facilitates the integration of AI-assisted segmentation into medical imaging workflows.

Keywords: medical image segmentation; deep learning; pretrained models; 3D imaging; desktop application; open-source software

Check for updates

Academic Editor: Alexander Zhbanov

Received: 10 February 2026
Revised: 22 March 2026
Accepted: 30 March 2026
Published: 3 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

1. Introduction

Medical image semantic segmentation is a crucial yet challenging problem in healthcare. Manual segmentation often requires time and expertise [1], while the automation of the task faces many obstacles [2] such as limited data availability for testing and validation, as well as technical issues (such as low contrast and poor signal-to-noise ratio). Traditional approaches, like thresholding and spatial filtering, have been proposed and developed over the years [3]. Recently, the field has shifted towards deep learning solutions, especially based on convolutional neural networks, which are a specialized type of artificial neural network that have achieved high performance [4,5]. One breakthrough was the development of U-Net [6], a neural network architecture initially designed specifically for 2D biomedical image segmentation and then extended to 3D data [7] and modified to get numerous variants such as nnU-Net [8], SwinUNETR [9] and others.

The rapid development of these networks led to the publication of a considerable number of deep learning models each year, often accompanied by source code (commonly in Python (Python Software Foundation, Beaverton, OR, USA)) and sometimes even with pretrained model weights. These models are usually developed to perform various semantic segmentation tasks, aimed to extract specific structures, such as organs, tumors or lesions. In support of developing such models, many annotated datasets have been made public such as: Medical Segmentation Decathlon [10], which includes many annotated body parts from computerized tomography (CT) images; Multimodal Brain Tumor Image Segmentation Benchmark (BraTs) [11,12], which provides multi-institutional and multi-modal magnetic resonance imaging (MRI) scans (typically including T1, T1c, T2, and FLAIR) with detailed annotations of glioma sub-regions; CEREBRUM-7T [13], which includes volumetric brain segmentations derived from ultra-high-field 7 Tesla MRI; and ToothFairy2 [14–16], which provides expert-annotated cone-beam CT scans for fine-grained tooth and dental structure segmentation, released as part of an international segmentation challenge. These datasets typically consist of paired files comprising the medical image and its corresponding segmentation mask, with identical spatial dimensions and voxel-wise label annotations. In addition, the files contain metadata describing voxel spacing and image orientation, which are essential not only for accurate visualization but also for downstream tasks such as allowing quantitative measurements to be expressed in physical units (millimeters) rather than pixel or voxel units, and surface mesh generation from segmentation results. Therefore, deep learning models have become more widely adopted in medical imaging workflows, which increased the importance of visualization and interpretability tools for understanding model predictions and supporting clinical decision-making [17].

In practice, medical deep learning models generally follow a common workflow that consists of image loading, preprocessing, model inference and postprocessing. Image loading involves dealing with specific formats used in clinics such as Digital Imaging and Communications in Medicine (DICOM), Neuroimaging Informatics Technology Initiative (NIfTI), and Nearly Raw Raster Data (NRRD). Preprocessing typically involves controlling image size, normalizing intensity ranges, adjusting spatial resolution, and/or applying smoothing or blurring filters. Inference consists of passing the preprocessed image into a segmentation model, which usually requires specific input settings, including spatial resolution and image dimensions. The model subsequently processes the image either as a full three-dimensional volume, using 3D convolutional kernels, or in a slice-wise manner, where 2D images are processed independently and later reconstructed into a 3D volume. The output of the model is usually voxel-wise probabilities representing the likelihood of the voxel to belong to a specific class representing a structure. This model inference can either be executed using the central processing unit (CPU) or using an accelerator via the

graphical processing unit (GPU). Finally, postprocessing typically includes the conversion of the model output into discrete labels using mathematical operations.

Practical use of pretrained segmentation models remains limited, as applying them typically requires significant programming expertise. Most published models lack user-friendly interfaces and are often published accompanied by a code tailored to the datasets on which they were trained, making adaptation to new datasets or workflows challenging. Although most models are implemented in PyTorch (PyTorch Foundation, San Francisco, CA, USA) [18] or Keras/TensorFlow (Google LLC, Mountain View, CA, USA) [19], loading and using them requires familiarity with framework-specific procedures. Moreover, visualizing and exporting segmentation results as 3D meshes (which has experienced a growing need in research and visualization, especially in augmented, mixed, or virtual reality applications, is not commonly supported by existing research code.

The Medical Open Network for Artificial Intelligence (MONAI) (NVIDIA Corporation, Santa Clara, CA, USA) [20] addresses several challenges related to reproducibility and standardization by providing a domain-specific framework for medical imaging research built on top of PyTorch. MONAI offers reusable components for data loading, preprocessing, model design, training and postprocessing, and their website allows for many pretrained models to be hosted and downloaded by users. However, MONAI primarily targets developers or researchers with programming experience and does not provide a standalone graphical user interface (GUI) for deploying arbitrary pretrained models in an end-user setting. Other existing open-source applications, such as 3D Slicer (Surgical Planning Laboratory, Brigham and Women's Hospital, Harvard Medical School, Boston, MA, USA) [21], provide GUI for medical image visualization and segmentation; nevertheless, they do not natively support arbitrary deep learning models. Users must either perform manual or semi-automated segmentation or rely on limited add-ons. For instance, TotalSegmentator [22] which offers pretrained models for 3D Slicer.

2. Methods

2.1. Software Architecture

SegMed is implemented as a standalone desktop application with modular architecture to allow independent management of data input, model inference, preprocessing, and postprocessing. The software was developed in Python (version 3.11) and leverages the PySide6 package (version 6.10, The Qt Company, Espoo, Finland) for GUI frameworks. The main modules are summarized as follows:

- Images Module: This module manages medical image input and associated metadata. It stores image array, segmentation arrays (both predicted by the model and ground truth that are loaded independently), spatial information (such as voxel spacing and orientation) and maintains the association between images and segmentation masks.
- DicomSeriesSelectionDialog Module: This module provides a window for the user to choose image series from the available series in case of loading from DICOMDIR files.
- PreProcessingWindow Module: This module shows preprocessing options to the user (Figure 1A) to define which operations should be applied before inference.
- PreprocessingExecutor Module: This module executes the selected preprocessing operations and enforces compatibility constraints between preprocessing steps.
- Networks Module: This module provides a unified interface for loading, configuring and executing deep learning models implemented in either PyTorch (version 2.X) or Keras (version 2.X), and created in MONAI (version 1.X) and nnU-Net (both versions V1 and V2) frameworks.

- NetworkDimensionInputWindow Module: This module provides a panel (Figure 1B) to manually input the model configuration such as dimensions, processing type (3D or 2D) and resizing options.
- nnunetHandler and nnunetHandlerV1 Modules: These modules are designed to specifically handle the nnU-Net pipeline and are used by the Networks module only.
- Mesh Module: This module converts segmentation masks (produced by the module or loaded by the user) into surface meshes and stores them as matrices of vertices and triangles for each class.

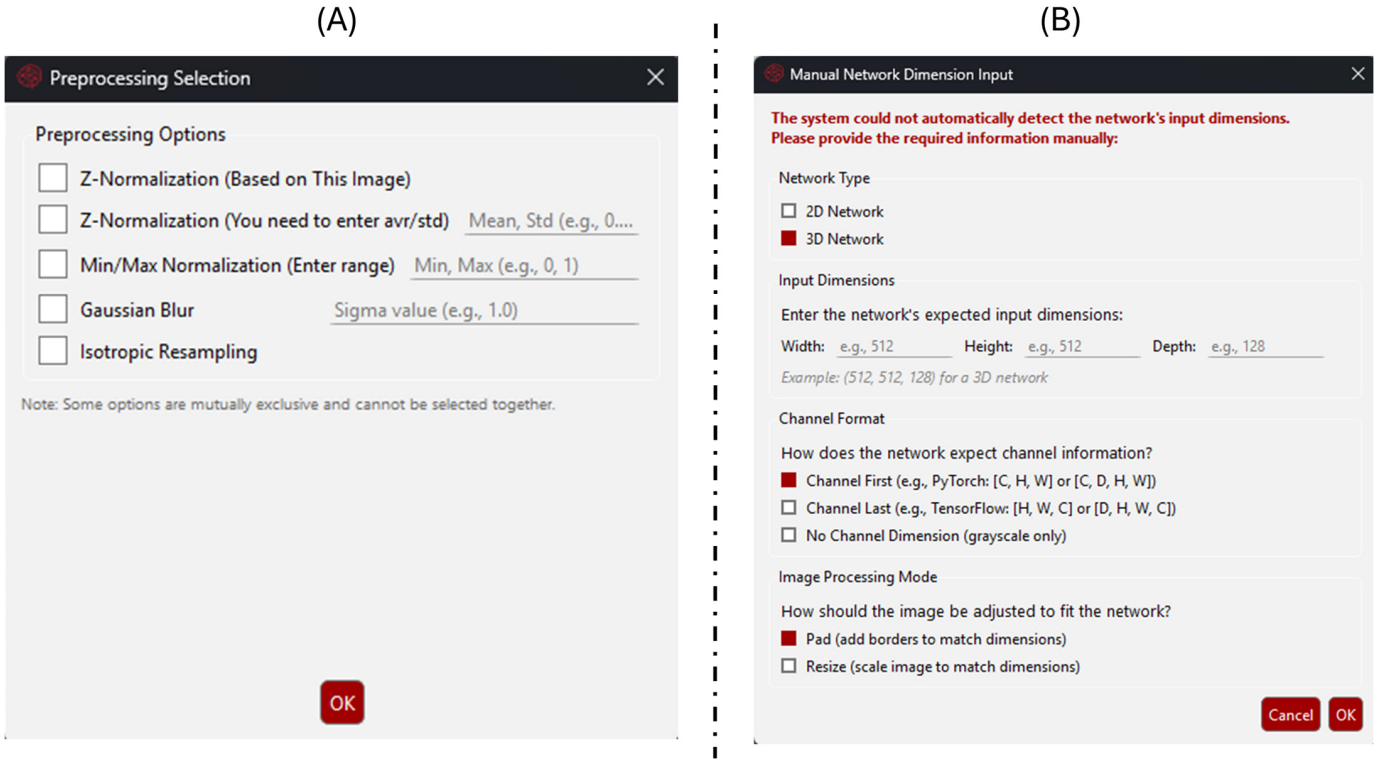


Figure 1. Windows from SegMed: (**A**) PreProcessingWindow showing the available preprocessing option users can set; (**B**) NetworkDimensionInputWindow which allows the user to manually configure the model.

- ColorSelectionWindow Module: This module controls visualization parameters, including class-specific colors for both 2D and 3D views, and allows users to enable or disable individual segmentation classes interactively, as well as assigning a name to each class.
- MainWindow Module: This module orchestrates the overall application workflow and user interaction (Figure 2 [23–26]). It connects all other modules and manages both 2D visualization (via matplotlib version 3.10.7) and 3D visualization (via OpenGL (version 3.1.10, Khronos Group, Beaverton, OR, USA)).

2.2. Software Functionality

SegMed operates according to the general workflow explained earlier, which consists of data loading, preprocessing, model inference, postprocessing, visualization and export of the results.

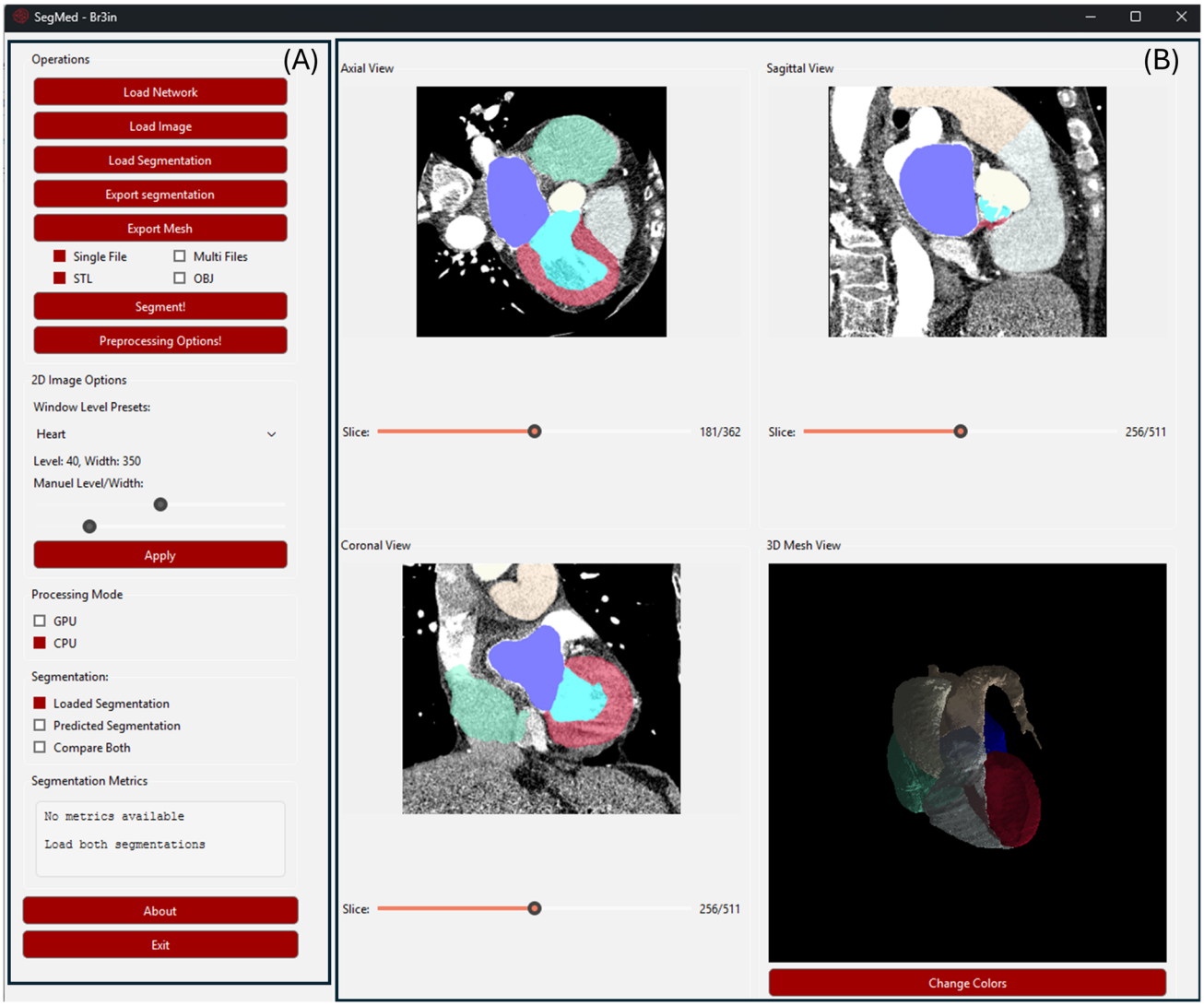


Figure 2. Main graphical user interface of SegMed with a cardiac example loaded [23–26]. The left control panel (**A**) provides access to core operations, including image and network loading, preprocessing configuration, segmentation execution, processing mode selection (CPU/GPU), and export of segmentation masks and 3D meshes. The central and right panels (**B**) display multi-planar visualization of the medical image with segmentation overlays in axial, sagittal, and coronal views, together with an interactive 3D mesh rendering of the segmented structures. Colors indicate the segmented anatomical classes as follows: left ventricle: cyan; myocardium: red; right ventricle: gray; left atrium: purple; right atrium: green; ascending aorta: cream; pulmonary artery: orange.

2.2.1. Data Loading and Visualization

SegMed supports medical images in DICOM, including both single DICOM files and complete DICOM series, NIfTI and NRRD formats. In all these cases, SegMed extracts the affine matrix from the metadata and related voxel spacing information and stores them. In addition, SegMed supports loading NumPy files (npy), though in that case SegMed will ask for JavaScript Object Notation (JSON) file with the related metadata; if not provided, SegMed assumes that the affine matrix is the identity transformation and the voxels are isotropic voxels. During visualization, the user may adjust window level and width for CT images either using predefined presets, which include preset values for multiple structures (Table 1), or setting arbitrary values. Also, the user may choose to skip the windowing and leveling and keep the raw original image settings; in that case a linear rescaling is applied.

Segmentation masks can also be imported independently. Thus, the user can compare the generated and ground truth segmentations.

Table 1. Presets defined in SegMed for window level and width for CT images.

Structure	Window Level (HU)	Window Width (HU)
Brain	40	80
Subdural	50	130
Stroke	35	25
Lung	−600	1500
Mediastinum	40	400
Liver	50	150
Abdomen	60	400
Bone	400	1800
Angiography	300	600
Heart	40	350

HU: Hounsfield Unit.

To perform a visual qualitative comparison the user can set SegMed in order to show the ground truth as a colored filled structure, while the generated segmentation is shown as a contour in 2D, and with a lower transparency in 3D. Quantitative comparison is also possible by computing the Dice similarity coefficients (D) and Intersection over union (I) for each class. D and I metrics are expressed in Equations (1) and (2) respectively, where TP, TN, FP and FN are the number of true positive, true negative, false positive and false negative voxels, respectively.

$$D = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{1}$$

$$I = \frac{TP}{TP + FP + FN} \tag{2}$$

2.2.2. Image Preprocessing

SegMed provides a set of commonly used preprocessing operations that can be selected by the user, with logical constraints applied to prevent incompatible preprocessing combinations. These preprocessing operations include z-normalization (Equation (3)), min/max normalization (Equation (4)), Gaussian blur (Equation (5)) with a Gaussian kernel (Equation (6))

$$I'(i,j,k) = \frac{I(i,j,k) - \mu}{\sigma} \tag{3}$$

$$I'(i,j,k) = a + \frac{I(i,j,k) - I_{min}}{I_{max} - I_{min}} \times (b - a) \tag{4}$$

$$I'(i.j.k) = \sum_{m=-c}^{c} \sum_{n=-c}^{c} \sum_{l=-c}^{c} I(i-m.j-n, k-l) G_\sigma(m,n,l) \tag{5}$$

$$G_\sigma(m,n,l) = \frac{1}{Z} e^{\left(-\frac{m^2+n^2+l^2}{2\sigma^2}\right)} \tag{6}$$

In Equations (3)–(6), I and I′ are the old and new intensity of a voxel respectively; i, j and k are voxel coordinates, μ and σ are the mean and standard deviation of the original image; a and b are intensity limits defined by the user; $c = 3\sigma$; Z is normalization constant and m, n and l are integers ranging from −c to +c.

2.2.3. Model Loading

The system implements a unified model loading architecture that supports multiple deep learning frameworks and specialized medical imaging models where the Network

module serves as the central interface, providing framework-agnostic model management. Five distinct model types are supported through automatic format detection:

- Keras/TensorFlow Models: These models can be loaded as a JSON architecture file (.json) or YAML architecture files (.yaml, .yml) with separate weights files (.h5) or a complete saved models (.h5, .keras) containing both architecture and weights in case the system employs custom object handling to manage Keras models with custom loss functions, metrics, or layers. When custom objects are detected in the model configuration, dummy functions are automatically created to enable successful loading without requiring the original custom implementations.
- PyTorch Models: These models can be TorchScript (.ts) with optional external weights files, or Checkpoint files (pt, pth,pth.tar) containing complete model state or model dictionary.
- MONAI Bundles: These bundles can be loaded after confirming the presence of required directory structure, including configs directory containing "inference.json" or "inference.yaml"; and models directory with trained model weights. The bundle loader implements a sophisticated multi-step initialization process, that uses the official Application Programming Interface (API) of MONAI.
- nnU-Net (V1 and V2) Models: Detection of these models employs automated version-specific validation, depending on the detected files. V2 detection requires: "plans.json" or "nnUNetPlans.json" and at least one "fold_X" directory containing "checkpoint_final.pth" or "checkpoint_best.pth". Instead, V1 detection requires at least one ".pkl" file (plans, postprocessing configuration), and at least one ".model" or "model_*.pth" checkpoint file.

2.2.4. Model Inference

SegMed implements a unified inference interface that abstracts framework-specific execution. Based on the detected model type, inference is routed to one of five specialized pipelines:

- Standard Keras/TensorFlow Model: Since these models store input shape properties, SegMed first applies the selected preprocessing operations and then determines whether the input image must be resized or padded with zeros to match the required model dimensions. SegMed explicitly informs the user of any structural modifications applied to the input image to match the model expectations. The preprocessed image is then passed through the network as a volume if the model expects a 3D input, or processed slice-by-slice otherwise. Finally, postprocessing involves applying rounding (in the case of binary segmentation) or the argmax operation (in the case of multiclass segmentation) to convert probability maps into label-wise segmentation masks.
- Standard PyTorch 3D Model: Image preprocessing and postprocessing match the case of the Keras/TensorFlow Model Inference. However, PyTorch models do not have the tag that describes the dimensions of the input. Thus, SegMed tries to determine the required input by passing a dummy image through the network, if it fails SegMed tries to pad the image, where each dimension is equal to the nearest greater power of 2. In cases when input dimensions cannot be reliably inferred through dynamic probing, such as in cases involving custom forward implementations or non-standard architectural constraints, the automatic configuration procedure may fail. In such instances, SegMed prompts the user to manually define the required input dimensions and processing mode.
- MONAI Bundle: For MONAI bundles, SegMed executes a full end-to-end clinical inference workflow derived from the bundle configuration. Input images are first converted to temporary NIfTI files with correctly constructed affine matrices to pre-

serve spacing, orientation, and origin. Preprocessing follows the extracted MONAI transform; thus all pre- and post- processing are determined by the bundle itself, and user choice is ignored. Inference is performed using MONAI's sliding window strategy using the window size described in the configuration.

- nnU-Net (v1 and v2): nnU-Net models are handled through dedicated pipelines that replicate the official inference behavior. For nnU-Net v2, SegMed uses the official nnU-Net predictor, automatically selecting the most appropriate configuration (3D full-resolution, 3D low-resolution, or 2D) based on the detected plans. Preprocessing, resampling, normalization, sliding-window inference, and postprocessing are executed exactly as defined during training. Ensemble prediction across multiple folds is supported by averaging probability maps prior to label fusion. For legacy nnU-Net v1 models, SegMed reconstructs the inference pipeline from stored pickle files, manually handling preprocessing, patch-based inference, aggregation of overlapping predictions, inverse resampling, and postprocessing, while maintaining compatibility with modern PyTorch versions and operating systems.

In all cases, SegMed does not modify model weights or the transformation pipelines defined by the original implementations. All preprocessing, resampling, and postprocessing operations are executed according to the inference configuration files provided with each model.

2.2.5. Result Visualization

Results can be visualized in both 2D and 3D. In 2D mode, users can navigate through axial, sagittal, and coronal slices with segmentation overlays, while in 3D mode, meshes are generated using the marching cubes algorithm [27], and can be interactively explored through rotation, zooming, and panning, with user-defined color mapping for each class, and the possibility to activate or deactivate classes.

2.2.6. Hardware Acceleration and Export

SegMed supports GPU acceleration via CUDA (NVIDIA Corporation, Santa Clara, CA, USA) when available; otherwise, computation defaults to the CPU. Segmentation results can be exported as NIfTI volumes or as 3D meshes in STL or OBJ format. Meshes may be saved per class or combined into a single object, enabling downstream applications such as surgical planning, educational visualization, and augmented, mixed, or virtual reality workflows.

3. Illustrative Examples

To verify compatibility and correct execution within SegMed, six publicly available pretrained segmentation models (Table 2) were selected from the literature. Each model was applied to a representative example image from a public dataset to confirm proper loading, preprocessing, and inference within the SegMed workflow. To provide quantitative context, Table 3 summarizes the training settings and reported D scores as published in the original studies or official model cards. All selected models were successfully executed in SegMed without modification of the original model code, producing volumetric segmentation masks with correct spatial alignment relative to the input images. Figure 3 presents representative segmentation outputs as visualized in SegMed, Figure 4 shows qualitative comparisons between predicted and ground-truth segmentations, and Table 4 reports the D and I values obtained from these illustrative runs [8,13,28–31]. These examples are intended to validate correct integration and execution of heterogeneous pretrained models and are not designed as a statistically rigorous performance evaluation.

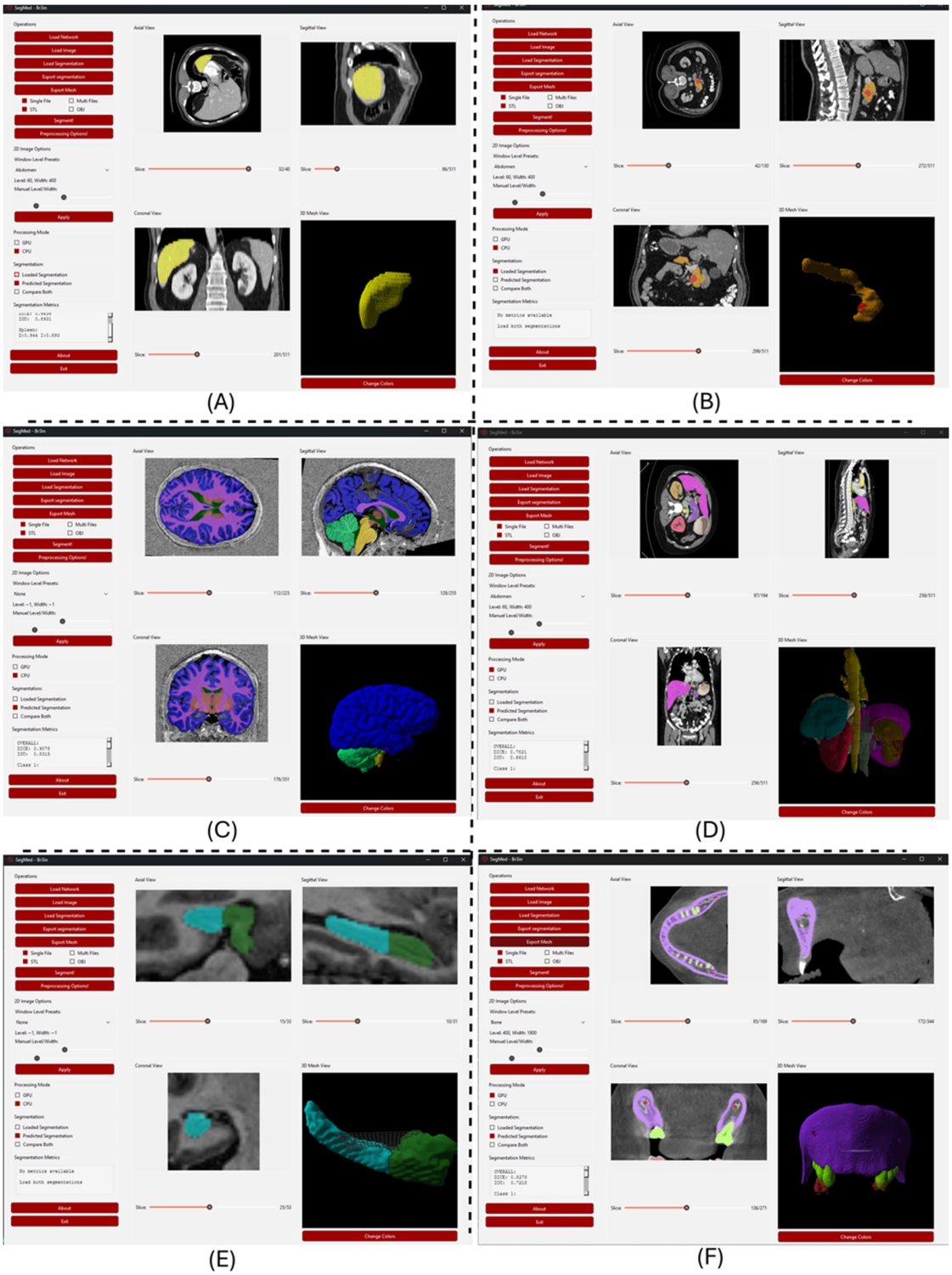


Figure 3. Representative segmentation results produced by SegMed using pretrained models, shown in multi-planar views with corresponding 3D mesh renderings: (**A**) spleen: yellow [28]; (**B**) pancreas: orange; tumor: red [29]; (**C**) brain tissues [13]: gray matter: blue; white matter: dark violet; cerebrospinal fluid: green; cerebellum: spring green; brain stem: gold; basal ganglia: brown; (**D**) abdominal organs [30]: spleen: cream; right kidney: red; left kidney: cyan; gallbladder: olive; esophagus: gray-green; liver: fuchsia; stomach: tan; aorta: yellow; inferior vena cava: light green-gray; portal vein and splenic vein: cyan; pancreas: purple; right adrenal gland: dark purple; left adrenal gland: bluish cyan; (**E**) hippocampus [8]: anterior hippocampus: forest green; posterior hippocampus: cyan; (**F**) dental structures [32]: upper skull: dark red; mandible: violet; upper teeth: light red; lower teeth: green; mandibular canal: red.

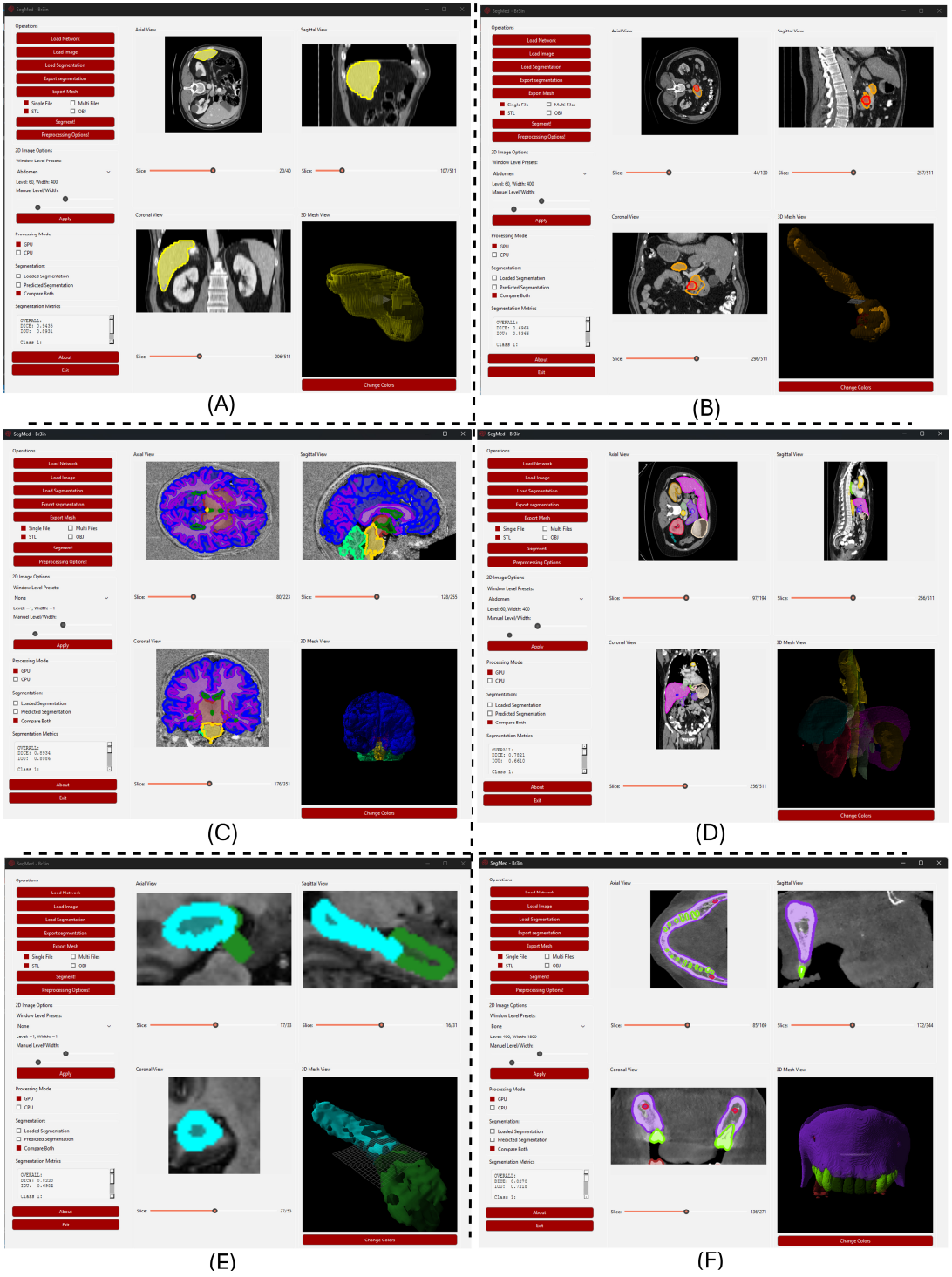


Figure 4. Qualitative comparison between ground-truth and predicted segmentations in SegMed for the illustrative examples: (**A**) spleen: yellow [28]; (**B**) pancreas: orange; tumor: red [29]; (**C**) brain tissues [13]: gray matter: blue; white matter: dark violet; cerebrospinal fluid: green; cerebellum: spring green; brain stem: gold; basal ganglia: brown; (**D**) abdominal organs [30]: spleen: cream; right kidney: red; left kidney: cyan; gallbladder: olive; esophagus: gray-green; liver: fuchsia; stomach: tan; aorta: yellow; inferior vena cava: light green-gray; portal vein and splenic vein: cyan; pancreas: purple; right adrenal gland: dark purple; left adrenal gland: bluish cyan; (**E**) hippocampus [8]: anterior hippocampus: forest green; posterior hippocampus: cyan; and (**F**) dental structures [32]: upper skull: dark red; mandible: violet; upper teeth: light red; lower teeth: green; mandibular canal: red. Ground-truth segmentations are shown as filled colored structures, whereas predicted segmentations are displayed as 2D contours and as semi-transparent 3D structures.

Table 2. Models and dataset used by SegMed.

Source	Framework	Segmentation Task	Type of Input	Number of Segmented Classes	Source of the Example Image
[28]	PyTorch using MONAI	Spleen	CT	1	[10]
[29]	PyTorch using MONAI	Pancreas and Tumor	CT	2	[10]
[13]	Keras	Brain	T1-weighted 7-Tesla MRI	6	[13]
[30]	PyTorch using MONAI	Abdomen	CT	13	[31]
[8]	PyTorch using nnU-Net v1	Hippocampus	T1-weighted MRI	2	[10]
[32]	PyTorch using nnU-Net v2	Dental	CT	5	[14–16]

Table 3. Training settings and reported Dice scores of the used pretrained models.

Source	Training/Evaluation Protocol	Reported Test D Across Structures (%)
[28]	282 training/139 testing images	62.0
[29]	32 training/9 validation images	96.0
[13]	120 training/4 validation/21 testing images	NR
[30]	24 training/6 testing images	82.8
[8]	5-fold cross-validation on the 195-image training set + held-out test submission	89.7
[32]	470 training/validation images; 133 internal test images; 123 external test images	92.2/94.2

D: Dice similarity coefficient, NR: Not reported.

Table 4. Quantitative comparison of the models used in the illustrative examples.

Segmentation Task	Class	D (%)	I (%)
Brain	Gray matter	89.7	81.4
	Basal ganglia	89.0	80.2
	White matter	93.1	87.1
	Ventricles	88.2	79.0
	Cerebellum	91.5	84.4
	Brainstem	93.0	87.0
Pancreas and Tumor	Pancreas	74.7	59.7
	Tumor	64.5	47.6
Spleen	Spleen	94.4	89.3
Abdomen	Spleen	91.8	84.9
	Right kidney	88.9	80.0
	Left kidney	90.1	82.0
	Gallbladder	41.1	25.8
	Esophagus	69.0	52.7
	Liver	96.8	93.8
	Stomach	85.3	74.4
	Aorta	77.6	63.4
	Inferior vena cava	78.4	64.5
	Portal vein and splenic vein	78.8	65.0
	Pancreas	72.7	57.1
	Right adrenal gland	68.4	51.9
	Left adrenal gland	77.8	63.7

Table 4. *Cont.*

Segmentation Task	Class	D (%)	I (%)
Hippocampus	Anterior hippocampus	80.4	67.3
	Posterior hippocampus	84.0	72.3
Dental	Upper Skull	61.0	43.9
	Mandible	85.8	75.1
	Upper Teeth	96.1	92.5
	Lower Teeth	87.4	77.7
	Mandibular Canal	83.5	71.7

D: Dice similarity coefficient, I: Intersection over union.

4. Discussion

This work presents SegMed, a standalone desktop application aimed at reducing the gap between state-of-the-art deep learning-based medical image segmentation methods and their practical adoption by researchers and clinicians. Indeed, SegMed provides an end-to-end graphical environment that abstracts low-level implementation details while preserving methodological fidelity to the original models.

A major strength of SegMed is its broad and unified support for pretrained models, including native handling of nnU-Net (both v1 and v2), in addition to standard PyTorch, Keras/TensorFlow, and MONAI-based networks. In fact, nnU-Net has become a de facto reference standard in medical image segmentation due to its self-configuring design and strong empirical performance across a wide range of segmentation tasks. However, deploying nnU-Net models outside their original training environment typically requires command-line interaction, strict adherence to directory conventions, and familiarity with the nnU-Net codebase, with these scripts usually supporting only NIfTI images.

By integrating the official nnU-Net inference pipelines directly into a desktop graphical interface, SegMed allows users to run these models while preserving the preprocessing, and postprocessing strategies defined during training. This reduces the risk of implementation deviations and lowers the barrier for non-expert users to benefit from nnU-Net-based methods.

More generally, SegMed preserves the original inference workflows of supported frameworks. For MONAI bundles, the software invokes the official MONAI application programming interface (API) and executes the transformation chain specified in the bundle configuration without reimplementation. For nnU-Net v2 models, the official nnU-Net predictor class is used together with the configuration files generated during training (e.g., plans and preprocessing specifications). Preprocessing, resampling, sliding-window inference, and fold ensembling are therefore performed according to the procedures defined in the original framework. For nnU-Net v1 models, the inference pipeline is reconstructed using the stored configuration and postprocessing files to replicate the official behavior while maintaining compatibility with modern software environments; for native Keras and PyTorch the weights are never changed. This design minimizes implementation deviations and preserves alignment with the reference inference workflows, thereby supporting methodological fidelity and reproducible deployment.

Another important contribution of SegMed is its emphasis on visual comparison of segmentation results, which is essential for qualitative assessment and methodological transparency. The software allows users to load ground-truth segmentations independently and directly compare them with predicted outputs using synchronized multi-planar views and interactive 3D visualizations. In 2D, predicted and reference segmentations can be displayed using complementary visual encodings (filled regions versus contours), while in 3D the use of transparency enables intuitive inspection of spatial disagreements. These

features support rapid error identification, anatomical plausibility checks, and visual validation of model behavior, which are often difficult to achieve using command-line tools or fragmented visualization pipelines. Such visual comparison capabilities are particularly valuable for researchers evaluating pretrained models on new datasets or assessing generalization performance beyond the original training domain. While visual inspection is important, numerical measurements are the way to measure the performance of a model. Thus, SegMed integrates numerical evaluation metrics, including the D and I metrics, computed on a per-class basis when reference segmentations are available. By embedding quantitative evaluation directly within the deployment environment, SegMed facilitates systematic and reproducible performance assessment without requiring additional scripting or external software. This functionality is especially relevant for replication studies, and methodological comparisons. SegMed's support for multiple medical image formats, including DICOM, NIfTI, NRRD and NumPy arrays, further enhances its applicability across clinical and research settings. DICOM compatibility facilitates adoption in radiological workflows, whereas NIfTI, NRRD and NumPy support aligns with common research practices in medical image analysis and deep learning. Crucially, SegMed preserves spatial metadata throughout the workflow, ensuring correct anatomical alignment and enabling downstream operations such as mesh generation and 3D visualization without additional transformation steps. The export of segmentation results as surface meshes (STL/OBJ) extends the utility of SegMed beyond voxel-based analysis and supports applications in anatomical modeling, surgical planning, and immersive visualization environments, which are rarely addressed by research code accompanying pretrained models.

SegMed positions itself as a standalone environment for deploying heterogeneous pretrained segmentation models within a unified graphical interface. Unlike existing solutions that typically focus on either visualization platforms, annotation workflows, or framework-specific pipelines, SegMed aims to bridge these approaches by enabling reproducible deployment of diverse models while preserving their original inference configurations, as summarized in Table 5, which compares SegMed to existing software.

Table 5. Comparison of SegMed with related open-source medical image software.

Feature	SegMed	3D Slicer	MONAI Label	nnU-Net (Official Implementation)
Standalone desktop GUI	Yes	Yes	No [1]	No [2]
Arbitrary pretrained model loading	Yes	Limited [3]	No	No
Native nnU-Net support	Yes	No	No	Yes [4]
Framework-agnostic (PyTorch, Keras, MONAI)	Yes	No	PyTorch only	PyTorch only
Multi-format input	Yes	Yes	Via host platform [5]	NIfTI only
Integrated quantitative evaluation	Yes	Limited	No	No
3D mesh export	Yes	Yes	Via host platform	No
Preserves official inference pipelines	Yes	No	Partial	Yes

[1]—Server-based (integrated into host platforms); [2]—Command-line-based; [3]—Requires extensions or scripting; [4]—Version-specific; [5]—Via host platform (e.g., 3D Slicer).

Regarding computational performance and hardware considerations, SegMed was evaluated on a workstation equipped with an NVIDIA RTX 5060 Ti GPU and 32 GB of RAM. Inference was tested in both GPU and CPU modes. As expected, GPU acceleration significantly reduced inference time, with performance improvements of up to 80% depending on the model architecture and input volume size. CPU-only execution remained functional but resulted in longer processing times, particularly for 3D sliding-window inference pipelines and multi-fold nnU-Net models. For MONAI bundle models, device configuration is

typically stored within the bundle settings (often defaulting to GPU execution). SegMed automatically overrides these device specifications when GPU resources are unavailable or when the user explicitly selects CPU execution. This ensures robust cross-device compatibility without requiring manual modification of the bundle configuration files. The available 32 GB of system memory was sufficient for all evaluated segmentation tasks. However, because SegMed generates and stores surface meshes for 3D visualization and export, memory consumption may increase substantially in cases involving very high-resolution volumes or meshes with a large number of vertices and faces. In such scenarios, hardware limitations may affect rendering responsiveness or mesh handling performance.

Despite these advantages, several limitations should be acknowledged. The current implementation assumes single-channel image inputs and single-stage models, which may restrict compatibility with multi-modal models (such as multi-contrast MRI) or special multi-stages models (such nnU-Net based on 3D cascade fullres configuration). In addition, long-term compatibility with rapidly evolving deep learning frameworks remains an inherent challenge, particularly for legacy models relying on deprecated APIs or custom components. While SegMed includes mitigation strategies, such as placeholder definitions for missing custom layers, full backward compatibility cannot be guaranteed. These limitations reflect broader issues in the field, where many published models lack standardized, deployment-ready configurations.

While SegMed does not introduce new segmentation algorithms, its scientific contribution lies in enabling the reproducible and standardized deployment of pretrained models across heterogeneous computing environments. By preserving official inference pipelines and spatial metadata handling within a unified interface, SegMed addresses a reproducibility bottleneck that currently limits the translation of published segmentation models into applied research and clinical experimentation. It provides a practical and reproducible solution for deploying pretrained medical image segmentation models in a standalone graphical environment.

As an open-source project released under the Apache 2.0 license, SegMed offers a flexible foundation for future development. Its modular architecture facilitates systematic extension, including planned support for multi-channel inputs and additional model configurations. This design ensures that current limitations are implementation-specific rather than structural and can be addressed in subsequent versions without major architectural redesign.

Author Contributions: Conceptualization, M.J.M. methodology, M.J.M.; software, M.J.M.; formal analysis, M.J.M., K.P.-v.D. and L.B.; investigation, M.J.M., A.S. and L.B.; resources, P.M.V.D. and L.B.; data curation, M.J.M.; writing—original draft preparation, M.J.M.; writing—review and editing, L.B., A.S., P.M.V.D. and K.P.-v.D.; visualization, M.J.M. and A.S.; supervision, P.M.V.D. and L.B.; project administration, L.B.; funding acquisition, L.B. and P.M.V.D. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partially supported by the Italian National Recovery and Resilience Plan (PNRR), Mission 4 Education and Research, Component 2 From Research to the Company, Investment 3.3 Introduction of innovative doctoral programs that meet the innovation needs of companies and promote the recruitment of researchers from companies (Ministerial Decree DM 117/2023), grant no. 39-033-01-DOT1301942-8805.

Data Availability Statement: The software source code is available at https://github.com/univpm-br3in/SegMed (accessed on 10 February 2026). All pretrained models and datasets are publicly available.

Pretrained Models:

1. MONAI Spleen CT Segmentation (v0.6.0), Pancreas CT DiNTS Segmentation (v0.5.1) and Swin UNETR BTCV Multi-organ Segmentation (v0.5.8) available via the MONAI Hugging Face Hub at https://huggingface.co/MONAI/ (accessed on 9 February 2026).
2. CEREBRUM-7T brain segmentation model, available at https://github.com/rockNroll87q/cerebrum7t (accessed on 9 February 2026).
3. Pretrained nnU-Net models for Hippocampus and Dental segmentation (DentalSegmentator), available on Zenodo at https://doi.org/10.5281/zenodo.4003545 and https://doi.org/10.5281/zenodo.10829675 (accessed on 9 February 2026).

Datasets:

1. Medical Segmentation Decathlon (MSD) datasets, including Spleen, Pancreas, and Hippocampus tasks, available at http://medicaldecathlon.com/ (accessed on 9 February 2026).
2. EBRAINS dataset (ID: 2b24466d-f1cd-4b66-afa8-d70a6755ebea), available via EBRAINS Search Service (accessed 9 February 2026).
3. ToothFairy2 dataset, available through Grand Challenge at https://toothfairy2.grand-challenge.org/ (accessed on 9 February 2026).
4. Multi-Atlas Labeling Beyond the Cranial Vault (Abdomen) dataset, available via Synapse (ID: syn3193805) at https://www.synapse.org/ (accessed on 9 February 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

CPU	Central Processing Unit
MRI	Magnetic Resonance Imaging
CT	Computed Tomography
DICOM	Digital Imaging and Communications in Medicine
GUI	Graphical User Interface
GPU	Graphics Processing Unit
HU	Hounsfield Unit
MONAI	Medical Open Network for AI
NIfTI	Neuroimaging Informatics Technology Initiative
NRRD	Nearly Raw Raster Data

References

1. Diaz, O.; Kushibar, K.; Osuala, R.; Linardos, A.; Garrucho, L.; Igual, L.; Radeva, P.; Prior, F.; Gkontra, P.; Lekadir, K. Data preparation for artificial intelligence in medical imaging: A comprehensive guide to open-access platforms and tools. *Phys. Medica* **2021**, *83*, 25–37. [CrossRef] [PubMed]
2. Hesamian, M.H.; Jia, W.; He, X.; Kennedy, P. Deep Learning Techniques for Medical Image Segmentation: Achievements and Challenges. *J. Digit. Imaging* **2019**, *32*, 582–596. [CrossRef]
3. Amiriebrahimabadi, M.; Rouhi, Z.; Mansouri, N. A Comprehensive Survey of Multi-Level Thresholding Segmentation Methods for Image Processing. *Arch. Comput. Methods Eng.* **2024**, *31*, 3647–3697. [CrossRef]
4. Xia, Q.; Zheng, H.; Zou, H.; Luo, D.; Tang, H.; Li, L.; Jiang, B. A comprehensive review of deep learning for medical image segmentation. *Neurocomputing* **2025**, *613*, 128740. [CrossRef]
5. Gao, Y.; Jiang, Y.; Peng, Y.; Yuan, F.; Zhang, X.; Wang, J. Medical Image Segmentation: A Comprehensive Review of Deep Learning-Based Methods. *Tomography* **2025**, *11*, 52. [CrossRef]
6. Ronneberger, O.; Fischer, P.; Brox, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. *arXiv* **2015**. [CrossRef]
7. Çiçek, Ö.; Abdulkadir, A.; Lienkamp, S.S.; Brox, T.; Ronneberger, O. 3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation. *arXiv* **2016**. [CrossRef]

8. Isensee, F.; Jaeger, P.F.; Kohl, S.A.A.; Petersen, J.; Maier-Hein, K.H. nnU-Net: A self-configuring method for deep learning-based biomedical image segmentation. *Nat. Methods* **2021**, *18*, 203–211. [CrossRef] [PubMed]
9. Hatamizadeh, A.; Nath, V.; Tang, Y.; Yang, D.; Roth, H.R.; Xu, D. Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images. *arXiv* **2022**. [CrossRef]
10. Antonelli, M.; Reinke, A.; Bakas, S.; Farahani, K.; Kopp-Schneider, A.; Landman, B.A.; Litjens, G.; Menze, B.; Ronneberger, O.; Summers, R.M.; et al. The Medical Segmentation Decathlon. *Nat. Commun.* **2022**, *13*, 4128. [CrossRef]
11. Bakas, S.; Akbari, H.; Sotiras, A.; Bilello, M.; Rozycki, M.; Kirby, J.S.; Freymann, J.B.; Farahani, K.; Davatzikos, C. Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features. *Sci. Data* **2017**, *4*, 170117. [CrossRef] [PubMed]
12. Menze, B.H.; Jakab, A.; Bauer, S.; Kalpathy-Cramer, J.; Farahani, K.; Kirby, J.; Burren, Y.; Porz, N.; Slotboom, J.; Wiest, R.; et al. The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS). *IEEE Trans. Med. Imaging* **2015**, *34*, 1993–2024. [CrossRef] [PubMed]
13. Svanera, M.; Benini, S.; Bontempi, D.; Muckli, L. CEREBRUM-7T: Fast and fully volumetric brain segmentation of 7 Tesla MR volumes. *Hum. Brain Mapp.* **2021**, *42*, 5563–5580. [CrossRef] [PubMed]
14. Lumetti, L.; Pipoli, V.; Bolelli, F.; Ficarra, E.; Grana, C. Enhancing Patch-Based Learning for the Segmentation of the Mandibular Canal. *IEEE Access* **2024**, *12*, 79014–79024. [CrossRef]
15. Bolelli, F.; Marchesini, K.; Van Nistelrooij, N.; Lumetti, L.; Pipoli, V.; Ficarra, E.; Vinayahalingam, S.; Grana, C. Segmenting Maxillofacial Structures in CBCT Volume. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: New York, NY, USA, 2025; pp. 5238–5248. [CrossRef]
16. Bolelli, F.; Lumetti, L.; Vinayahalingam, S.; Di Bartolomeo, M.; Pellacani, A.; Marchesini, K.; van Nistelrooij, N.; van Lierop, P.; Xi, T.; Liu, Y.; et al. Segmenting the Inferior Alveolar Canal in CBCT Volumes: The ToothFairy Challenge. *IEEE Trans. Med. Imaging* **2025**, *44*, 1890–1906. [CrossRef]
17. Bhati, D.; Neha, F.; Amiruzzaman, M. A Survey on Explainable Artificial Intelligence (XAI) Techniques for Visualizing Deep Learning Models in Medical Imaging. *J. Imaging* **2024**, *10*, 239. [CrossRef]
18. Paszke, A.; Gross, S.; Massa, F.; Lerer, A.; Bradbury, J.; Chanan, G.; Killeen, T.; Lin, Z.; Gimelshein, N.; Antiga, L.; et al. PyTorch: An Imperative Style, High-Performance Deep Learning Library. *arXiv* **2019**. [CrossRef]
19. TensorFlow Developers. *TensorFlow (v2.20.0)*; Zenodo: Geneva, Switzerland, 2025. [CrossRef]
20. Cardoso, M.J.; Li, W.; Brown, R.; Ma, N.; Kerfoot, E.; Wang, Y.; Murrey, B.; Myronenko, A.; Zhao, C.; Yang, D.; et al. MONAI: An open-source framework for deep learning in healthcare. *arXiv* **2022**. [CrossRef]
21. Kikinis, R.; Pieper, S.D.; Vosburgh, K.G. 3D Slicer: A platform for subject-specific image analysis, visualization, and clinical support. In *Intraoperative Imaging and Image-Guided Therapy*; Springer: Berlin/Heidelberg, Germany, 2013; pp. 277–289. [CrossRef]
22. Wasserthal, J.; Breit, H.-C.; Meyer, M.T.; Pradella, M.; Hinck, D.; Sauter, A.W.; Heye, T.; Boll, D.T.; Cyriac, J.; Yang, S.; et al. TotalSegmentator: Robust segmentation of 104 anatomic structures in CT images. *Radiol. Artif. Intell.* **2023**, *5*, e230024. [CrossRef]
23. Gao, S.; Zhou, H.; Gao, Y.; Zhuang, X. BayeSeg: Bayesian Modeling for Medical Image Segmentation with Interpretable Generalizability. *Med. Image Anal.* **2023**, *89*, 102889. [CrossRef]
24. Wu, F.; Zhuang, X. Minimizing Estimated Risks on Unlabeled Data: A New Formulation for Semi-Supervised Medical Image Segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **2023**, *45*, 6021–6036. [CrossRef] [PubMed]
25. Zhuang, X.; Shen, J. Multi-scale Patch and Multi-modality Atlases for Whole Heart Segmentation of MRI. *Med. Image Anal.* **2016**, *31*, 77–87. [CrossRef] [PubMed]
26. Zhuang, X. Multivariate Mixture Model for Myocardial Segmentation Combining Multi-Source Images. *IEEE Trans. Pattern Anal. Mach. Intell.* **2019**, *41*, 2933–2946. [CrossRef]
27. Lorensen, W.E.; Cline, H.E. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH'87)*; Association for Computing Machinery: New York, NY, USA, 1987; pp. 163–169. [CrossRef]
28. He, Y.; Yang, D.; Roth, H.; Zhao, C.; Xu, D. DiNTS: Differentiable Neural Network Topology Search for 3D Medical Image Segmentation. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*; IEEE: New York, NY, USA, 2021; pp. 5837–5846. [CrossRef]
29. Xia, Y.; Liu, F.; Yang, D.; Cai, J.; Yu, L.; Zhu, Z.; Xu, D.; Yuille, A.; Roth, H. 3D Semi-Supervised Learning with Uncertainty-Aware Multi-View Co-Training. *arXiv* **2020**. [CrossRef]
30. Tang, Y.; Yang, D.; Li, W.; Roth, H.R.; Landman, B.; Xu, D.; Nath, V.; Hatamizadeh, A. Self-Supervised Pre-Training of Swin Transformers for 3D Medical Image Analysis. *arXiv* **2022**. [CrossRef]

31. Harrigan, R. Segmentation Outside the Cranial Vault Challenge. Synapse. 2015. Available online: https://repo-prod.prod.sagebase.org/repo/v1/doi/locate?id=syn3193805&type=ENTITY (accessed on 10 February 2026).
32. Dot, G.; Chaurasia, A.; Dubois, G.; Savoldelli, C.; Haghighat, S.; Azimian, S.; Taramsari, A.R.; Sivaramakrishnan, G.; Issa, J.; Dubey, A.; et al. DentalSegmentator: Robust open source deep learning-based CT and CBCT image segmentation. *J. Dent.* **2024**, *147*, 105130. [CrossRef] [PubMed]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.