PyMIC: A deep learning toolkit for annotation-efficient medical image segmentation

Guotai Wang [a,b,*], Xiangde Luo [a,b], Ran Gu [a], Shuojue Yang [c], Yijie Qu [a], Shuwei Zhai [a], Qianfei Zhao [a], Kang Li [d], Shaoting Zhang [a,b]

[a] School of Mechanical and Electrical Engineering, University of Electronic Science and Technology of China, Chengdu, China
[b] Shanghai Artificial Intelligence Laboratory, Shanghai, China
[c] Laboratory for Computational Sensing and Robotics, Johns Hopkins University, Baltimore, USA
[d] West China Biomedical Big Data Center, West China Hospital, Sichuan University, Chengdu, China

ARTICLE INFO

ABSTRACT

Background and Objective: Open-source deep learning toolkits are one of the driving forces for developing medical image segmentation models that are essential for computer-assisted diagnosis and treatment procedures. Existing toolkits mainly focus on fully supervised segmentation that assumes full and accurate pixel-level annotations are available. Such annotations are time-consuming and difficult to acquire for segmentation tasks, which makes learning from imperfect labels highly desired for reducing the annotation cost. We aim to develop a new deep learning toolkit to support annotation-efficient learning for medical image segmentation, which can accelerate and simplify the development of deep learning models with limited annotation budget, e.g., learning from partial, sparse or noisy annotations.

Methods: Our proposed toolkit named PyMIC is a modular deep learning library for medical image segmentation tasks. In addition to basic components that support development of high-performance models for fully supervised segmentation, it contains several advanced components that are tailored for learning from imperfect annotations, such as loading annotated and unannounced images, loss functions for unannotated, partially or inaccurately annotated images, and training procedures for co-learning between multiple networks, etc. PyMIC is built on the PyTorch framework and supports development of semi-supervised, weakly supervised and noise-robust learning methods for medical image segmentation.

Results: We present several illustrative medical image segmentation tasks based on PyMIC: (1) Achieving competitive performance on fully supervised learning; (2) Semi-supervised cardiac structure segmentation with only 10% training images annotated; (3) Weakly supervised segmentation using scribble annotations; and (4) Learning from noisy labels for chest radiograph segmentation.

Conclusions: The PyMIC toolkit is easy to use and facilitates efficient development of medical image segmentation models with imperfect annotations. It is modular and flexible, which enables researchers to develop high-performance models with low annotation cost. The source code is available at:https://github.com/HiLab-git/PyMIC.

1. Introduction

Automatic segmentation of medical images is essential for computer-assisted diagnosis and treatment planning/guidance of disease [1]. Recently, with the fast development of deep learning techniques, the segmentation performance has been largely improved and even becomes comparable to that of human experts for clinical use [2]. In addition to hardware revolution and increased data availability, open-source deep learning toolkits are one of the driving forces for the boom of deep learning methods and their applications in the medical image segmentation community [3,4]. Existing general-purpose deep learning frameworks such as TensorFlow [5], PyTorch [6] and Keras[1] provide modular and lower-level functions for deep learning in a wider community of computer vision and natural language processing, without specific functionali-

---

* Corresponding author at: 2006 Xiyuan Avenue, West Hi-Tech Zone, Chengdu 611731, China.
  E-mail address: guotai.wang@uestc.edu.cn (G. Wang).

[1] https://keras.io/

ties for medical image segmentation, and adapting them for medical images requires substantial implementation efforts.

There have been several open-source projects proposed for medical image computing including segmentation tasks in the last couple of decades. Some early toolkits implemented traditional algorithms for medical image computing, such as NiftyReg [7] and Elastix [8] for registration and NiftySeg [9] for segmentation. Researchers also developed some application-specific toolkits, such as Freesurfer [10] and FSL [11] for structural and functional neuroimaging analysis. However, they did not support deep learning models for medical image analysis. In recent years, several research groups have extended general deep learning toolkits with more domain-specific functionalities to ease medical image computing based on deep learning. NiftyNet [12] and DLTK [13] enable fast development of deep learning models for medical image analysis using TensorFlow [5]. However, they are not actively maintained and updated now. DeepNeuro [14] is also a framework built on the top of TensorFlow, and provides a templating language for designing and implementing deep learning pipelines for neuroimaging. MONAI (Medical Open Network for AI) [4] that is the predecessor of NiftyNet and built on PyTorch provides modules for data loading, training paradigm, network structure and loss functions for medical image computing models. TorchIO [15] is a library for data loading, prepossessing, augmentation and patch sampling of medical images in deep learning. Pymia [16] also provides functionalities for handling medical images in different formats, in addition to evaluation of deep learning models. It can be integrated into both TensorFlow and PyTorch pipelines. The nnU-Net [3] framework based on PyTorch is specifically designed for deep learning-based medical image segmentation, and it automatically configures itself for data preprocessing, network architecture, training and post-processing, leading to state-of-the-art performance on many international biomedical image segmentation challenges.

However, the existing toolkits only considered fully supervised learning for medical image segmentation tasks, with the assumption that pixel-level dense and accurate annotations are available for the training set. The implementation of such a supervised learning paradigm in these toolkits is simple and straightforward: loading image and label pairs from a fully annotated dataset and training a single network with common fully-supervised loss functions. For medical image segmentation, the acquisition of pixel-level annotations is extremely time-consuming, labor-insensitive and difficult with high cost [17]. First, differently from natural images, annotation of medical images requires many expertise, where the availability of experts will limit the annotation process. Second, medical images such as Computed Tomography (CT) and Magnetic Resonance Imaging (MRI) are volumetric data often with hundreds of 2D slices, and delineating multiple organs [2] or lesions in a volume would take several hours, making annotating a large dataset very expensive and time-consuming. Thirdly, due to the low contrast of medical images and varying experience of annotators, the annotation may be inaccurate and contain some noise [18]. The standard supervised learning paradigm in existing toolkits do not provide support for dealing with such datasets with incomplete or imperfect annotations, which impedes researchers from fast developing their models when the annotations are limited.

In recent years, annotation-efficient learning for medical image segmentation has attracted increasing attention as it can largely reduce the annotation cost for training deep learning models [17]. For example, Semi-Supervised Learning (SSL) leverages a small set of annotated images and a large set of unannotated images for training, which avoids annotating the entire dataset. Some Weakly Supervised Learning (WSL) methods only require sparse or coarse annotations (e.g., scribble and bounding box that are much cheaper than pixel-level annotations. In addition, some researchers use noisy annotations given by non-experts for training

when access to experts is limited [18,19], i.e., Noisy Label Learning (NLL).

The implementation of such methods is largely different from that of supervised learning, and requires a lot of advanced functionalities for data handling, network design, loss function and training pipeline that are not accessible in existing toolkits. For example, semi-supervised learning methods often load annotated images and unannotated images in a single batch for training [20,21], which requires redesign of the data loader and batch sampler. The loss function in semi- and weakly supervised learning is often a combination of supervised loss for annotated pixels and different types of regularization for unannotated pixels [20,22], and such regularization and their combination are ignored in existing toolkits. In addition, modification of network structures with multiple branches or co-learning between different networks [19,21,23] are often involved in annotation-efficient learning. Therefore, a lot of implementation efforts are needed by these algorithms for learning from imperfect annotations. As independent reimplementation of all of these customized modules leads to substantial duplication of efforts and inhibits fair comparison between competing methods, a unified software toolkit with built-in support for annotation-efficient learning is highly desired.

This work presents the open-source PyMIC (PyTorch for Medical Image Computing) toolkit based on PyTorch for annotation-efficient learning for medical image segmentation. In addition to flexible and modular functionalities for training, inference and evaluation of medical image segmentation models, it provides advanced built-in modules for data handling, network design, loss functions and training procedures required by semi-supervised, weakly supervised and noisy label learning. It allows fast development of deep learning segmentation models using limited annotations, and facilitates comparison between different methods.

2. Methods

The PyMIC toolkit aims to fill the gap between current deep learning toolkits for fully supervised learning and learning with limited (e.g., partial, sparse and noisy) annotations in medical image segmentation. PyMIC is built on PyTorch that provides basic ingredients for deep learning but without specific functionalities for dealing with medical images and annotation-efficient learning. PyMIC implements higher-level interfaces for medical image segmentation with advanced functionalities optimized for learning from imperfect annotations.

2.1. System overview

Fig. 1 shows an overview of PyMIC that contains several modular components. Firstly, it leverages lower-level functions in PyTroch to implement a modularized and simplified deep learning pipeline, and defines a SegmentationAgent to connect and encapsulate several basic core components for training and inference of deep learning models: Dataset, BatchGenerator and Network for common dataflow, Loss and Optimizer used in Training procedure and an Inferer for inference. The SegmentationAgent can easily handle supervised learning tasks, and accepts configuration files to run the deep learning pipeline with different settings of datasets, data augmentation strategies, networks and loss functions, etc.

Secondly, we extend the basic core components with advanced functionalities that are specifically designed for handling and training with datasets with imperfect annotations, such as batch aggregation that aggregates samples from annotated and unannotated images into a single batch, multi-branch networks, uncertainty estimation and regularization losses used in semi- and weakly supervised segmentation. We also provide advanced training procedures such as co-learning between different networks simultane-
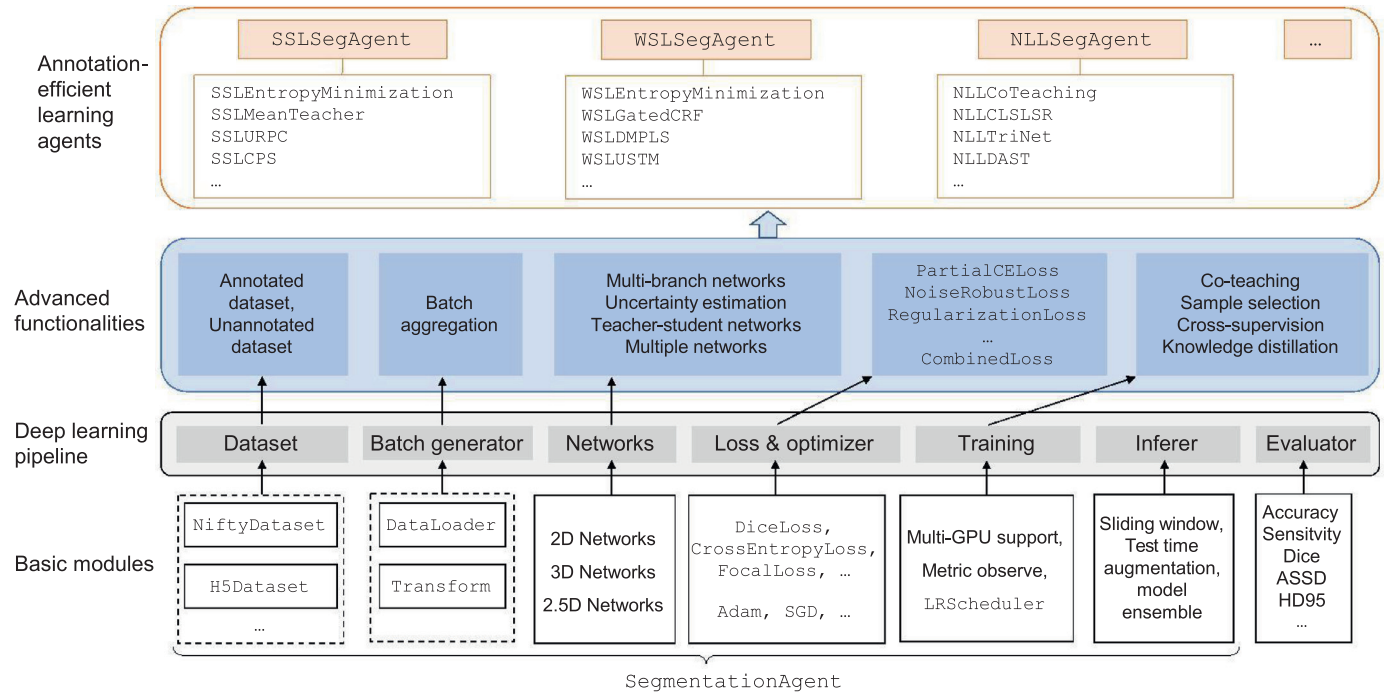
Fig. 1. An overview of the PyMIC framework for medical image segmentation with annotation-efficient learning.

ously and sample selection for dynamically filtering out samples with low-quality annotations.

Thirdly, based on SegmentationAgent and the advanced functionalities, we define derived classes for annotation-efficient learning agents including SSLSegAgent, WSLSegAgent and NLLSegAgent that provide general frameworks for semi-supervised learning, weakly supervised learning and noisy label learning for segmentation, respectively. As shown in Fig. 1, these classes are parent classes of specific annotation-efficient learning algorithms, such as SSLUAMT that uses Uncertainty-Aware Mean Teacher (UAMT) for semi-supervised learning [21] and WSLGatedCRF that uses gated Conditional Random Field (CRF) for weakly supervised learning [24]. In the following subsections, we give detailed description of these modules respectively.

2.2. Basic modules for fully supervised learning

PyMIC implements several core components of a deep learning pipeline that are specialized for medical image segmentation and encapsulated into the SegmentationAgent class. These components include dataset and batch generator, network, loss and optimizer, training and inference procedures, etc. An Evaluator is also implemented for quantitative evaluation of segmentation results with various evaluation metrics.

Medical images such as CT and MRI are saved in special formats such as DICOM, Nifti and Analyze that are not supported by general deep learning frameworks including Tensorflow and Py-Torch. The Nifti format is the most widely used in the community, and the other formats can be converted to Nifti that saves image data array together with metadata including spatial information, patient information and acquisition information. Therefore, we define a NiftyDataset to load medical images, which uses load_image_as_nd_array() as a general interface to read different types of images as N-dimensional arrays with meta information including pixel spacing. For example, a 3D image is loaded as a 4D array with the shape of $C \times D \times H \times W$, where C, D, H and W represent the channel number, slice number, height and width, respectively. Note that this function can load a regular 2D image with

different formats such as.jpg,.png and.bmp as a 3D array with the shape of $C \times H \times W$ as well. In addition, to load large volumes efficiently, PyMIC defines a H5Dataset to support HDF5 (hierarchical data format version 5) that allows access of chunks of data without loading the entire volume.

With NiftyDataset or H5Dataset, BatchGenerator is implemented to sample a batch from the dataset. It reuses DataLoader in PyTorch and Transform for data preprocessing and augmentation. Transform includes a set of intensity and spatial transforms, such as NormalizeWithMeanStd and NormalizeWithMinMax for intensity normalization, GammaCorrection and GaussianNoise for random intensity augmentation and RandomCrop, RandomFlip, RandomRescale, RandomRotate, Pad and others for spatial augmentation. Note that these transforms support 2D and 3D images in PyMIC, and the inverse operation is allowed for spatial transforms.

Several types of neural networks are defined in PyMIC for medical image segmentation. 2D networks such as UNet2D [25], AttentionUNet2D [26] and NestedUNet2D [27] are implemented to segment 2D images or slices of 3D images. Some 3D networks such as UNet3D [28] and UNet3D_ScSE [29] are available for segmentation of 3D volumes. In addition, many medical image datasets have high intra-plane resolution and low inter-plane resolution, making networks using purely 2D or 3D convolutions less effective. PyMIC supports 2.5D networks such as UNet2D5 that combines 2D and 3D convolutions in a single network to better deal with such images. What's more, PyMIC supports several loss functions widely used for image segmentation, such as DiceLoss, CrossEntropyLoss, FocalLoss and ExpLogLoss.

During training, PyMIC reuses built-in optimizers and Learning Rate (LR) schedulers in PyTorch. It also allows training with multiple GPUs by reusing DataParallel that parallelizes the network by splitting the input across a list of devices. In addition, PyMIC will measure the segmentation performance via some evaluation metrics such as Dice on the training and validation sets during training, which is better than just using the loss values for selecting the best checkpoint.

For inference, an Inferer object is defined in PyMIC with functionalities for medical images that are not readily available in Py-

Torch. It allows sliding window inference for a large volume by automatically spiting the input into a list of patches and aggregating the corresponding results into a single segmentation volume in the original image space. Test time augmentation is also supported by Inferer, where different transformed versions of the input are segmented and transformed inversely to the original spatial coordinates for segmentation fusion. In addition, Inferer allows ensemble of multiple checkpoints for higher segmentation accuracy and robustness.

With the segmentation ground truth and network's prediction, an Evaluator object allows quantitative evaluation of the segmentation results. It supports a range of metrics including sensitivity, specificity, accuracy, precision, Dice coefficient, Average Symmetric Surface distance (ASSD), Hausdorff Distance (HD) and 95 percentile of HD (HD95), etc. Users can specify the list of classes and the names of metrics for evaluation, and the results are saved in.csv format via a CSVWriter. Evaluator also allows metric observation during training.

2.3. Advanced functionalities for learning from imperfect annotations

Based on the above modules that facilitate fully supervised learning for segmentation, PyMIC further extends them in several aspects with advanced functionalities to support annotation-efficient learning, including data handling, networks, loss functions and training procedures specified for semi-supervised, weakly supervised and noisy-label learning.

For semi-supervised setting where the dataset consists of an annotated subset and an unannotated subset, NiftyDataset and H5Dataset are extended by allowing loading images without labels. In addition, two instances of such datasets are created in the segmentation agent, and they are for the annotated and unannotated subsets, respectively. SSL methods usually require annotated and unannotated images in a single batch [20,21], and PyMIC supports this by loading N_L and N_U images from the two dataset instances respectively to compose a batch with $N_L + N_U$ images, i.e., batch aggregation.

PyMIC supports several advanced and specialized networks for annotation-efficient learning. For example, multi-branch networks are often used for SSL [20,30], WSL [31] and NLL [32] tasks, and PyMIC extends encoder-decoder networks like U-Net with multiple decoders or multi-level predictions for this purpose. Different uncertainty estimation methods are also supported in PyMIC, e.g., Monte Carlo Dropout [21], test-time augmentation and multi-branch predictions [20] that are commonly used for SSL tasks. In addition, PyMIC has implementations of self-ensembling of networks that are widely used in Mean Teacher-based methods [21,22], and it extends the SegmentationAgent to allow multiple instances of networks that are required in some SSL [33] and NLL [19,34] methods.

In addition to general supervised loss functions, PyMIC has implementations of several losses specialized for annotation-efficient learning. For example, in SSL and WSL, the predictions for unannotated pixels are often regularized. The RegularizationLoss is implemented to support different kinds of regularization, such as entropy-minimization, total variation, and size and shape regularization, etc. NoiseRobustLoss is implemented for dealing with noisy labels, such as noise-robust Dice loss [18], Generalized Cross Entropy (GCE) loss [35] and Mean Absolute Value (MAE) loss [36], etc. In addition, to facilitate the combination of different kinds of loss functions, a CombinedLoss allows users to select the loss functions for combination and specify their corresponding weights.

The Training procedure of SegmentationAgent is derived to support different training paradigms for annotation-efficient learning. For example, in co-teaching [34], the training involves two networks where each network selects samples based on loss values

for the other network in a batch. In Mean Teacher [37], a self-ensembling of the student network serves as a teacher the output of which is used to supervise the student. NLLCoTeaching and SLLMeanTeacher are derived from SegmentationAgent by redefining the training process to support these two training paradigms, respectively. Other types of training paradigms, such as knowledge distillation and confidence learning are also supported in similar ways in PyMIC.

2.4. Annotation-efficient leaning agents

With the above advanced functionalities, three derivatives of SegmentationAgent have been currently implemented in PyMIC for semi-supervised learning, weakly supervised learning and noisy label learning, respectively: SSLSegAgent, WSLSegAgent and NLLSegAgent, as shown in Fig. 1. Then, different kinds of specific annotation-efficient learning methods are derived from one of these abstract classes. Table 1 shows the detailed annotation-efficient learning methods currently implemented in PyMIC.

The SSLSegAgent is an abstract agent for SSL and provides the general setting of a combination of annotated and unannotated subsets for training. It is the parent class of different semi-supervised segmentation methods, such as SSLMean-Teacher [37] based on mean teacher and SSLURPC [20] that uses uncertainty rectified pyramid consistency. Other methods including entropy minimization [38], UAMT [21], Cross-Pseudo Supervision (CPS) [33] and Cross-Consistency Training (CCT) [30] are also implemented as child classes of SSLSegAgent.

For weakly supervised segmentation, PyMIC supports learning from partial annotations (e.g., scribbles), and a PartialCELoss is implemented for computing cross entropy loss only for the annotated pixels. It is combined with different regularization losses for training in derivatives of WSLSegAgent, such as WSLEntropyMinimization [38], WSLTotalVariation, WSLMumfordShah [39] and WSLGatedCRF [24]. These agents can reuse networks for supervised learning during training. In addition, some WSL methods [22,31] require modified neural networks or multiple forward passes in each iteration, and PyMIC implemented their training procedures accordingly, such as WSLUSTM [22] that combines UAMT [21] with transform consistency for regularizing predictions of unannotated pixels, and WSLDMPLS [31] that dynamically mixes predictions at two decoders as the pseudo label for training with scribble annotations.

For noisy label learning, PyMIC has implementations of several noise-robust loss functions, such as GCELoss [35], MAELoss [36] and NoiseRobustDiceLoss [18]. They can be used by SegmentationAgent for training with a standard supervised learning paradigm. In addition, some NLL methods require special training procedures like interaction between multiple networks, sample selection and cross-supervision. Child classes of NLLSegAgent are implemented to support them, such as NLLCoTeaching [34], NLLCLSLSR [40] that combines confident learning with spatial label smoothing, NLLTriNet [19] that uses three networks for training, and NLLDAST [32] that combines a dual-branch network with divergence-aware selective training.

2.5. Flexibility and extendability

The modular design of PyMIC provides high flexibility and extensibility to different levels of users. PyMIC supports configuration files for specifying the dataset, data augmentation methods, network, loss functions and training parameters such as batch size, optimizer and learning rate scheduler. Users are allowed to just edit the configuration file without writing code to run the training and inference pipeline with different settings. In addition, different modules in PyMIC can be customized by a user, so that the user only needs to define and set the customized module in the

Table 1

Overview of annotation-efficient learning methods currently implemented in PyMIC for medical image segmentation. SSL, WSL and NLL represent semi-supervised learning, weakly supervised learning and noisy label learning, respectively.

Category	Method	Reference	Remarks
SSL	SSLEntropyMinimization	Grandvalet et al., 2005 [38]	Originally proposed for classification
	SSLMeanTeacher	Tarvainen et al., 2017 [37]	Originally proposed for classification
	SSLUAMT	Yu et al., 2019 [21]	Uncertainty-aware mean teacher
	SSLURPC	Luo et al., 2022 [20]	Uncertainty rectified pyramid consistency
	SSLCCT	Ouali et al., 2020 [30]	Cross-consistency training
	SSLCPS	Chen et al., 2021 [33]	Cross-pseudo supervision
WSL	WSLEntropyMinimization	-	Adapted from [38]
	WSLTotalVariation	-	Total variation loss for regularization
	WSLMumfordShah	Kim et al. 2020 [39]	Mumford-shah loss for regularization
	WSLGatedCRF	Obukhov et al., 2019 [24]	Gated CRF for regularization
	WSLUSTM	Liu et al., 2022 [22]	Adapt UAMT with transform-consistency
	WSLDMPLS	Luo et al., 2022 [31]	Dynamically mixed pseudo label supervision
NLL	NoiseRobustDiceLoss	Wang et al., 2020 [18]	Train with SegmentationAgent
	GCELoss	Zhang et al., 2018 [35]	Train with SegmentationAgent
	MAELoss	Ghosh et al., 2017 [36]	Train with SegmentationAgent
	NLLCoTeaching	Han et al., 2018 [34]	Co-teaching between two networks
	NLLCLSLSR	Zhang et al., 2020 [40]	Confident learning with spatial label smoothing
	NLLTriNet	Zhang et al., 2020 [19]	Tri-network combined with sample selection
	NLLDAST	Yang et al., 2022 [32]	Divergence-aware selective training

segmentation agent without re-implementing the remaining parts. For example, a user may define a customized network by inheriting nn.Module and use SegmentationAgent.set_network() to easily integrate it into the existing pipeline.

2.6. Platform and dependencies

PyMIC is implemented in Python (Python Software Foundation, Wilmington, DA, U.S., version 3.6 or higher). It depends on PyTorch, Numpy, Pandas, SciPy and SimpleITK. The detailed dependent libraries and versions are encoded in a requirements.txt file in the code repository, which enables automatic installation of compatible libraries. Users can easily install PyMIC via the pip installation tool, i.e., running the command of pip install PYMIC. The documentation is hosted on Read the Docs[2], which facilitates users starting with PyMIC quickly.\

3. Results: Applications of PyMIC

In this section, we present several user cases of PyMIC for 2D and 3D medical image segmentation with full annotations, partial annotations, weak annotations and noisy annotations, respectively. These user cases are available on GitHub[3]. The experiments were conducted on a Ubuntu desktop with two NVIDIA GTX 1080 Ti GPUs available.

3.1. Fully supervised segmentation

Based on PyMIC, we have previously won two medical image segmentation challenges hold on MICCAI conference: MyoPS 2020 [41] for myocardial pathology segmentation and CATARACTS 2020 [42] for automatic tool annotation for cataRACT surgery. Here we demonstrate how PyMIC can be easily used for these two fully supervised segmentation tasks.

The MyoPS 2020 challenge aims to segment five structures from Cardiac Magnetic Resonance (CMR) images: Left Ventricular (LV) Normal Myocardium (NM), LV Blood Pool (BP), Right Ventricular (RV) BP, LV myocardial edema and scars. It provides 45 cases of three-sequence CMR images (bSSFP, LGE and T2 CMR), where the

sequences in each case had been aligned and resampled to the same resolution by the organizers. Each case had 2–6 slices with inter-slice spacing 12.0–23.0 mm. The matrix size was around 480 × 480 with pixel size around 0.73 mm × 0.73 mm. Due to the small slice number, we used 2D networks for experiments with this dataset. The images was split into 25 cases for training and 20 cases for testing. As the labels of the testing cases are not publicly available, we use the official training cases for experiments, where 5-fold cross validation on the 25 cases is conducted in this paper.

We used SegmentationAgent with a configuration file for training and inference, where a UNet2D was trained by a combination of DiceLoss and CrossEntropyLoss. A lot of Transform objects, including RandomRotate, RandomCrop, RandomFlip, NormalizeWithMeanStd, GammaCorrection, GaussianNoise, were used for data augmentation. The Adam optimizer was used with ReduceLROnPlateau scheduler for training, with initial learning rate 0.001 and maximal iteration number $30k$. During inference, test time augmentation was used in Inferer. The quantitative evaluation metrics were computed using PyMIC's evaluation method and aggregated over all folds.

The segmentation results by PyMIC was compared with those by nnU-Net[4] [3]. As shown in Fig. 2, PyMIC performed better than nnU-Net for the segmentation task. The average Dice value obtained by PyMIC was 0.795, 0.922, 0.935, 0.400, 0.613 for LV NM, LV BP, RV BP, edema and scars, respectively. Note that the edema and scars are small objects that are hard to segment, some advanced strategies such as model ensemble and coarse-to-fine segmentation can be combined with PyMIC for better performance, as demonstrated in [43].

The CATARACTS 2020 [42] challenge is about semantic segmentation of cataract surgery videos. There were three sub-tasks for segmentation of different numbers of classes, and in this paper we deal with Task I where 8 classes need to be segmented: 4 for anatomy (pupil, iris, cornea and skin), 1 for all instruments and 3 for other objects (surgical tap, eye retractors and hand). The challenge dataset contains 4670 images from 25 videos for training and 531 images from 10 videos for testing. As the labels of testing images are not publicly available, we only used the official training set for experiment, and it was split into 3550 (19 videos), 534 (3
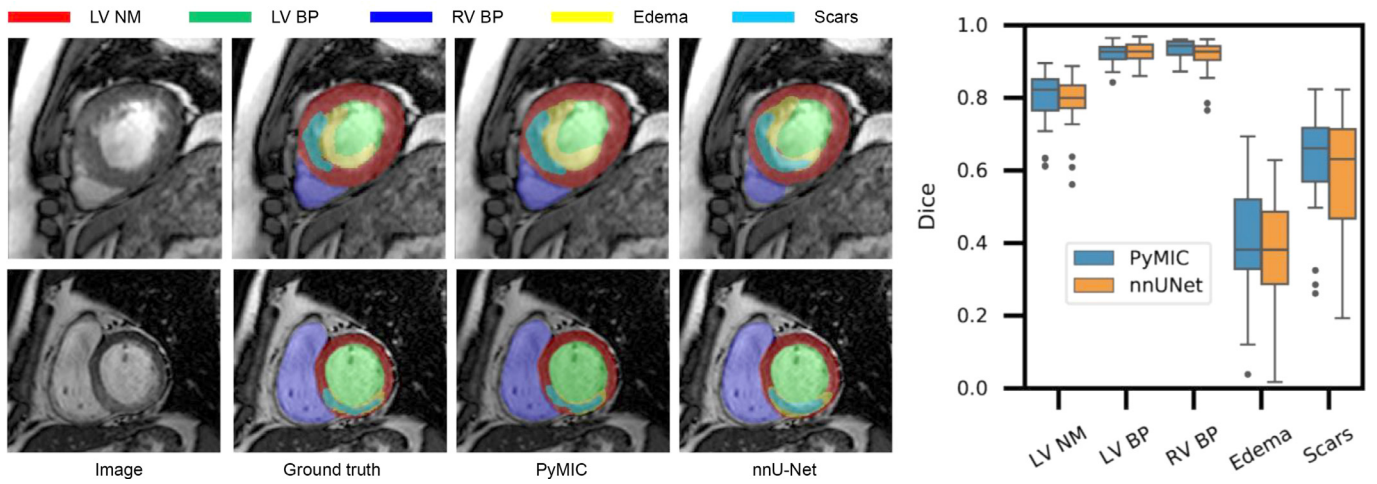

Fig. 2. Myocardial pathology segmentation results obtained by PyMIC and nnU-Net [3]. Left: qualitative comparison. Right: quantitative evaluation in terms of Dice. LV: Left ventricle; RV: Right ventricle; NM: Normal myocardium; BP: Blood pool.
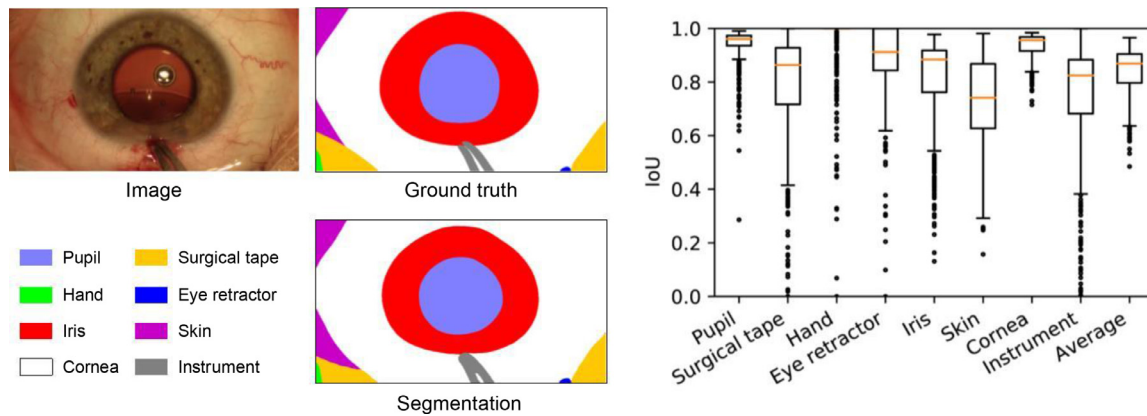

Fig. 3. Semantic segmentation of cataract surgery images obtained by PyMIC. Left: qualitative comparison. Right: quantitative evaluation in terms of IoU.

videos) and 586 (3 videos) for training, validation and testing, respectively.\

For this task we combined PyMIC with a customized CNN, where the network in SegmentationAgent was set as a pre-trained HRNet [44] implemented in a third-party repository[5]. The data augmentation methods were similar to those used in the MyoPS challenge, and we replaced RandomRotate by RandomRescale. We used a batch size of 12 and trained HRNet on two GPUs with DiceLoss and CrossEntropyLoss. The initial learning rate was 10^{-4} with maximal iteration number of 100k. Test time augmentation and sliding window inference were enabled in Inferer. The segmentation results are shown in Fig. 3, where the left side shows a qualitative comparison between the segmented objects and the ground truth, and the right side shows quantitative evaluation in terms of Intersection over Union (IoU) as used in [42].

3.2. Semi-supervised segmentation

To demonstrate semi-supervised segmentation in PyMIC, we first used the Automatic Cardiac Diagnosis Challenge (ACDC) dataset [45] containing 200 short-axis cardiac cine-MR volumes of 100 patients for segmentation of the Right Ventricle (RV), Myocardium (Myo) and Left Ventricle (LV). Each volume has 6–18 slices, with an inter-slice spacing of 5–10 mm. We randomly selected 20 volumes of 10 patients for validation, and 40 volumes of

20 patients for testing. The remaining 140 volumes of 70 patients were used for training, where 14 volumes from 7 patients (10%) and the other 126 volumes were used as the annotated and unannotated samples, respectively.

We compared several SSL methods implemented in PyMIC with the baseline of learning only from the annotated samples: SSLEntropyMinimization [38], SSLUAMT [21], SSLURPC [20], SSLCCT [30], and SSLCPS [33]. All these methods used a 2D network (UNet2D) as the backbone due to the small number of slices and large inter-slice spacing in the volumetric images, and were trained with a batch size of 8 that consists of 4 annotated and 4 unannotated samples. Adam optimizer was used for training, with initial learning rate of 10^{-3} and maximal iteration number of 30k. The results were post-processed by selecting the largest connected component implemented in post_process module of PyMIC. Quantitative evaluation results in terms of Dice and HD95 are shown in Table 2. The baseline method obtained an average Dice of 83.32%, 85.93% and 91.76% for RV, Myo and LV, respectively. The SSL methods significantly improved the performance, e.g., CPS [33] improved the average Dice to 89.09%, 87.09% and 92.49% for the three classes, respectively.

In addition, we also demonstrate PyMIC's support for 3D semi-supervised segmentation with the left atrial dataset [46] that consists of 100 3D atrial gadolinium-enhanced MR images. The original images have a resolution of $0.625 \times 0.625 \times 0.625$ mm^3, with spatial dimensions of $576 \times 576 \times 88$ or $640 \times 640 \times 88$ pixels. For preprocessing, we cropped the images into a dimension of 256

---

[5] https://github.com/HRNet/HRNet-Semantic-Segmentation

Table 2

Quantitative comparison of different semi-supervised methods for cardiac MRI segmentation on the ACDC dataset, with 10% of training images annotated. * denotes significant improvement from the baseline (*p*-value < 0.05) based on a paired t-test..

| Methods	Dice (%)				HD95 (mm)			
	RV	Myo	LV	Average	RV	Myo	LV	Average
Baseline	83.32±11.91	85.93±5.30	91.76±5.87	87.00±6.39	11.34±10.72	7.64±8.15	7.27±8.66	8.75±7.35
EM [38]	84.25±11.47	86.35±5.45	**92.96±4.53***	87.86±5.97	12.89±17.69	5.75±6.69*	6.62±8.23	8.42±7.87
UAMT [21]	86.90±8.73*	86.23±4.54	92.38±5.01	88.50±4.62*	10.60±12.15	6.12±6.77	7.03±6.85	7.92±6.27
URPC [20]	88.70±7.20*	87.09±4.73*	92.78±4.78	89.53±4.31*	8.81±10.96*	5.73±6.84*	**5.63±6.62**	6.73±6.24*
CCT [30]	85.98±10.75*	86.59±5.00*	91.98±5.42	88.18±5.48*	9.24±8.03	6.28±7.75	7.07±8.32	7.53±5.92
CPS [33]	**89.09±7.32***	**87.09±4.44***	92.49±5.18	**89.56±4.11***	**6.78±6.56***	**5.02±4.96***	6.42±8.01	**6.08±4.87***
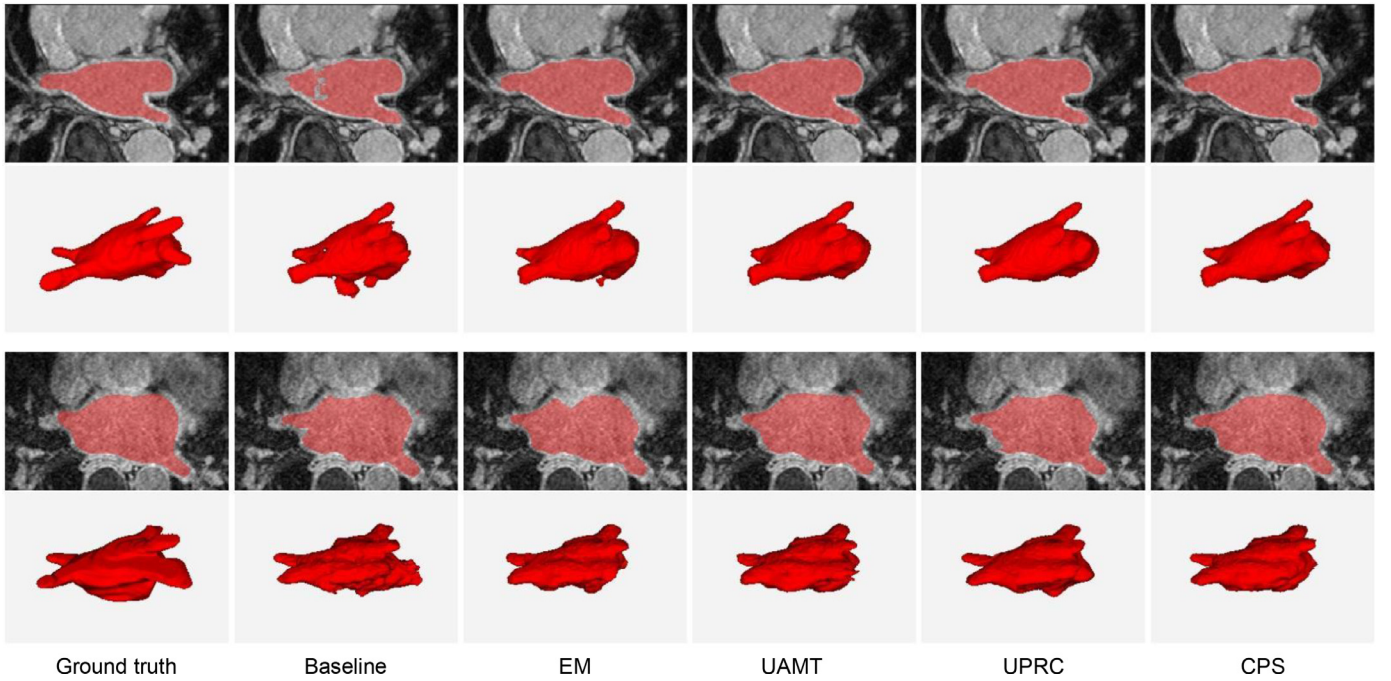
Fig. 4. Semi-supervised 3D left atrial segmentation results. The models were trained with 7 annotated and 65 unannotated volumes. The first two rows show the segmentation results of one patient with 2D and 3D visualizations, respectively, and the last two rows show the results of another patient.

× 160 × 88. The dataset was split into 72, 8 and 20 cases for training, validation and testing, respectively. For the 72 training images, we took 7 (∼10%) as annotated images and the other 65 (∼90%) as unannotated images. UNet3D [28] was used as the backbone network, and we compared the baseline of learning only from the annotated images with: SSLEntropyMinimization [38], SSLUAMT [21], SSLURPC [20], and SSLCPS [33]. Note that SSLCCT [30] was not used in this experiment due to the limit of memory. The patch size was 72 × 96 × 112, with a batch size of 4 consisting of 2 annotated and 2 unannotated samples. We also used Adam optimizer for training, with initial learning rate of 10^{-3} and maximal iteration number of 20*k*.

Table 3 shows the quantitative evaluation results of these compared methods. The baseline obtained an average Dice of 82.58%,

Table 3

Quantitative comparison of different semi-supervised methods for 3D left atrial segmentation, with 10% of training images annotated. * denotes significant improvement from the baseline (*p*-value < 0.05) based on a paired t-test.

	Dice (%)	HD95 (mm)	ASSD (mm)
Baseline	82.58±8.62	20.19±14.75	3.62±2.96
EM [38]	84.62±9.14	15.75±13.74	2.83±2.87*
UAMT [21]	84.85±6.97	24.00±18.73	3.93±3.17
URPC [20]	85.06±8.01*	18.44±13.99	3.23±2.83
CPS [33]	**86.50±4.68***	**13.36±12.92***	**2.39±2.30***

and CPS [33] improved it to 86.50%, which outperformed the other semi-supervised learning methods. CPS also obtained the lowest HD95 and ASSD among the compared methods. URPC [20] ranked the second place in terms of average Dice, and EM [38] ranked the second place in terms of HD95 and ASSD. Fig. 4 shows a visual comparison between these different methods. It can be observed that the baseline obtained obvious under-segmentation and over-segmentation, while the semi-supervised methods improved the segmentation quality by leveraging unannotated images for learning.

3.3. Learning from sparse annotations

We also used the ACDC dataset as an example for weakly supervised segmentation, where scribble annotations provided by Valvano et al. [47] instead of full annotations were used for training, as shown in the left side of Fig. 5. The data split, backbone network, optimizer, learning rate and post-process method were the same as in Section 3.2. Five WSL methods implemented in PyMIC were evaluated: WSLEntropyMinimization, WSLTotalVariation, WSLGatedCRF [24], WSLUSTM [22] and WSLDMPLS [31], and they were compared with the baseline of learning only from the annotated pixels with PartialCELoss. Quantitative evaluation of these methods are shown in Table 4. Using partial CE loss only obtained an average Dice of 86.39%, and DMPLS [31] improved it to 89.28%, which outperformed the other compared methods. Note
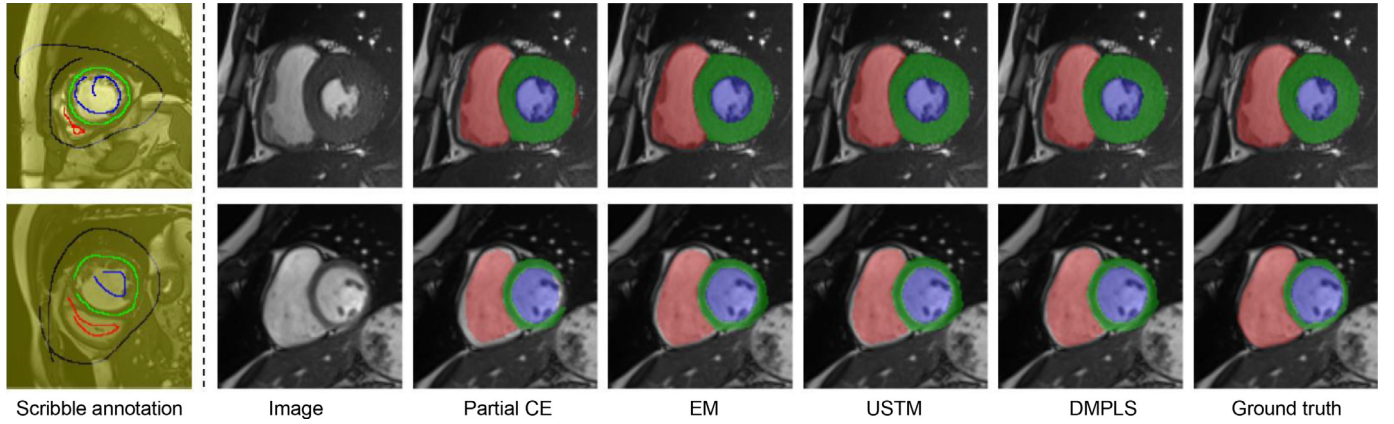
Fig. 5. Weakly supervised cardiac structure segmentation on ACDC dataset.

Table 4
Quantitative comparison of different methods learning from scribbles for cardiac MRI segmentation on the ACDC dataset. * denotes significant improvement from the baseline (*p*-value < 0.05) based on a paired t-test..

| Methods	Dice (%)				HD95 (mm)			
	RV	Myo	LV	Average	RV	Myo	LV	Average
Partial CE	85.96±5.37	83.66±5.87	89.54±7.79	86.39±4.86	7.30±4.25	7.08±8.16	6.84±7.55	7.07±5.15
EM [38]	87.83±7.19*	85.93±4.31*	92.42±5.01*	88.73±4.00*	6.10±4.01*	**4.07±4.29***	4.29±4.66*	4.82±3.39*
TV	87.93±6.17*	85.84±4.11*	**92.71±4.13***	88.83±3.27*	6.68±6.11	4.26±4.05*	5.07±10.56	5.34±5.01
Gated CRF [24]	86.94±5.77*	85.40±4.47*	91.79±5.44*	88.05±3.86*	6.49±3.69	4.73±4.65*	5.35±6.65	5.53±3.78*
USTM [22]	87.00±6.59*	84.74±4.83*	91.84±4.42*	87.86±3.80*	6.87±4.73	4.74±3.99*	4.27±3.29*	5.30±2.78*
DMPLS [31]	**89.02±5.28***	**86.17±4.50***	92.66±4.22*	**89.28±3.08***	**5.21±3.46***	4.26±3.89*	**3.97±4.09***	**4.48±2.96***
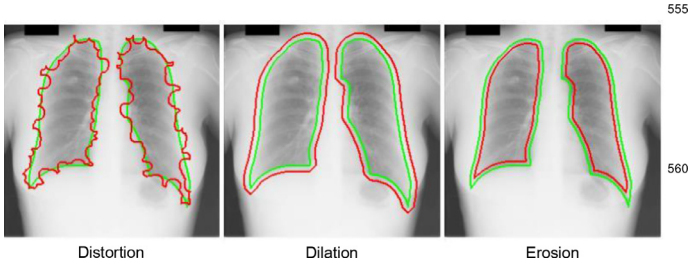
Fig. 6. Examples of simulated noisy labels (red) compared with clean label (green) for lung segmentation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Table 5
Quantitative comparison of different methods learning from noisy labels for lung segmentation. † denotes the best value that is significantly better than all the others (*p*-value < 0.05) based on a paired t-test.

	Dice (%)	HD95 (pix)	ASSD (pix)
Baseline (CE)	86.87±3.43	25.57±30.00	6.83±3.06
GCE loss [35]	87.87±3.31	14.05±10.94	5.58±1.68
CL-SLSR [40]	89.52±3.22	12.19±11.29	4.50±1.35
Co-teaching [34]	94.25±2.45	11.92±8.08	2.60±1.13
TriNet [19]	94.87±1.62	**7.82±3.87†**	2.26±0.66
DAST [32]	**96.94±1.57†**	11.85±13.74	**1.84±1.51**

that the performance of DMPLS implemented in this paper is better than that in the original paper [31], showing the effectiveness of PyMIC. The right side of Fig. 5 shows qualitative comparisons of these methods. It can observed that the partial CE baseline obtained noisy segmentation in the first row and under-segmentation in the second row respectively, while the other weakly supervised methods obtained better results, and DMPLS outperformed the other compared methods.

3.4. Dealing with noisy labels

The JSRT dataset [48] with 247 chest radiographs was used for experiments with NLL. The images have a size of 2048 × 2048, and were resampled to 256 × 256. We split the dataset into 180, 20 and 47 for training, validation and testing, respectively. We only consider the lung for segmentation, and simulated noisy annotations for 95% of the training images (171 cases) by applying dilation, erosion or edge distortion to the ground truth clean labels, as shown in Fig. 6.

We compared six methods implemented in PyMIC for this NLL task: 1) the baseline of training with cross entropy loss and stan-

dard setting in SegmentationAgent; 2) training with GCELoss [35]; 3) Confident Learning with Spatial Label Smoothing Regularization (CL-SLSR) [40] implemented in NLLCLSLSR; 4) NLLCoTeaching [34], 5) NLLTriNet [19]; and 6) NLLDAST [32]. All these methods used UNet2D [25] as the backbone network for segmentation, and were trained with a batch size of 8, Adam optimizer, initial learning rate of 10^{-3} and iteration number of 10k. No post-processing was used at inference time. Quantitative evaluation results are listed in Table 5. It shows that the baseline method obtained an average Dice of 86.87%. GCE loss and CL-SLSR improved it to 87.87% and 89.52%, respectively. Co-teaching, TriNet and DAST obtained a higher performance improvement, with average Dice of 94.25%, 94.87% and 96.94%, respectively, showing the effectiveness of dealing with noisy labels in PyMIC.

4. Discussion and conclusions

The current success of deep learning for medical image segmentation mostly relies on a large annotated dataset for fully supervised learning, which is well supported by several existing deep learning toolkits for medical image computing [3,4,12,49]. However, in a wider range of scenarios, the annotations of medical images are expensive and time-consuming to acquire, leading to de-

mand for annotation-efficient training methods for deep learning models. Recent years have seen a rapid increase of researches on annotation-efficient learning algorithms [17,23], but there is a lack of toolkits with built-in and modular functionalities for fast implementation of annotation-efficient learning methods.

We present PyMIC, a Python toolkit for annotation-efficient deep learning for medical image segmentation. PyMIC shares some similarity with MONAI [4] with key differences. MONAI is a PyTorch-based framework for deep learning for healthcare. In terms of supervised segmentation, both PyMIC and MONAI have domain-specialized modules for image I/O, data transforms, network structures, loss functions, evaluation metrics, and training and inference, etc. However, MONAI aims to support a wider range of healthcare data such as images, videos, tubular data and EEG signals, and it mainly covers training pipelines for fully supervised learning. In contrast, PyMIC is specialized for annotation-efficient medical image segmentation, and it focuses on functionalities for training with imperfect annotations, such as partial, sparse and noisy annotations, although it also contains basic modules to support fast development of competitive fully supervised segmentation models.

With the built-in modules in PyMIC, users can easily run state-of-the-art annotation-efficient learning algorithms to train their own models by just editing a configuration file. PyMIC is easy to install, and the source code can be accessed from the Git repository[6]. PyMIC provides a high flexibility to configure a training pipeline with 2D and 3D networks in different segmentation tasks. For example, for segmentation tasks on the ACDC and left atrial datasets in Section 3.2, the different SSL methods implemented in PyMIC are independent of the network structures, and the user only needs to set the backbone network as a 2D or a 3D network in the two tasks, without changing the code for training. The WSL and NLL methods in PyMIC also allow users to freely set different network structures in the configure file. In addition, PyMIC supports customized modules, where the user can reuse most part of the training pipeline in PyMIC and only needs to focus on implementing the customized component, which has a potential for boosting researches and applications of annotation-efficient medical image segmentation algorithms.

The current released version of PyMIC is v0.3, and it has been successfully used for several research projects in medical image segmentation. For example, in fully supervised learning, some novel networks such as CA-Net [50], MG-Net [51] and COPLE-Net [18] have been developed on the top of PyMIC, and they have been applied to segmentation of organs at risks for radiotherapy [52], fetal brain and placenta [50], and COVID-19 lesions [18,32]. Segmentation models implemented in PyMIC have won some MICCAI segmentation challenges, including MyoPS 2020 [43] and CATARACTS 2020 [42]. In addition, it has been used in researches on annotation-efficient learning, such as semi-supervised segmentation of pulmonary fibrosis from CT volumes [53], scribble-based weakly supervised learning [31], and noise-robust learning [18,32].

There are also some limitations in the current version of PyMIC that are expected to be addressed in the future. First, the current built-in networks mainly include some UNet-like structures, and we are working on implementing more state-of-the-art networks in PyMIC to facilitate the users. Second, there are some other types of annotation-efficient learning approaches not considered in PyMIC currently, such as self-supervised learning approaches [54,55] and domain adaption methods [56,57] for medical image segmentation. Thirdly, despite that our framework achieved comparable and even better performance than the self-configuring

nnU-Net [3] in fully supervised learning (Fig. 2), it requires the user's expertise to design a suitable set of configuration parameters. In addition, the semi- and weakly supervised learning pipelines usually have a lower training efficiency than fully supervised approaches, and their interpretability need to be improved. In the future, PyMIC will be extended to support more medical image computing tasks, such as image classification and object detection. It will be actively updated to support more state-of-the-art deep learning models and annotation-efficient learning methods for medical image computing.

In summary, we present the open-source toolkit PyMIC to fill the gap between existing deep learning toolkits and annotation-efficient medical image segmentation. It's modular functionalities support fast development of medical image segmentation models with limited annotations based on the built-in implementations of semi-supervised, weakly-supervised and noisy-label learning algorithms. It's flexibility allows customized modules with minimized implementation burden on users, which facilitates researches and applications of deep learning for medical image computing with low annotation cost.

Declaration of Competing Interest

The authors declare no conflicts of interest.

Acknowledgments

References

[1] D. Shen, G. Wu, H.-I. Suk, Deep learning in medical image analysis, Annu Rev Biomed Eng 19 (1) (2017) 221–248.
[2] H. Tang, X. Chen, Y. Liu, Z. Lu, J. You, M. Yang, S. Yao, G. Zhao, Y. Xu, T. Chen, Y. Liu, X. Xie, et al., Clinically applicable deep learning framework for organs at risk delineation in CT images, Nature Machine Intelligence 1 (10) (2019) 480–491, doi:10.1038/s42256-019-0099-z.
[3] F. Isensee, P.F. Jaeger, S.A.A. Kohl, J. Petersen, K.H. Maier-Hein, et al., NnU-Net: a self-configuring method for deep learning-based biomedical image segmentation, Nat. Methods 18 (2) (2021) 203–211.
[4] M.J. Cardoso, W. Li, R. Brown, N. Ma, E. Kerfoot, Y. Wang, B. Murrey, A. Myronenko, C. Zhao, D. Yang, V. Nath, Y. He, Z. Xu, A. Hatamizadeh, A. Myronenko, W. Zhu, Y. Liu, M. Zheng, Y. Tang, I. Yang, M. Zephyr, B. Hashemian, S. Alle, M.Z. Darestani, C. Budd, M. Modat, T. Vercauteren, G. Wang, Y. Li, Y. Hu, Y. Fu, B. Gorman, H. Johnson, B. Genereaux, B.S. Erdal, V. Gupta, A. Diaz-Pinto, A. Dourson, L. Maier-Hein, P.F. Jaeger, M. Baumgartner, J. Kalpathy-Cramer, M. Flores, J. Kirby, L.A.D. Cooper, H.R. Roth, D. Xu, D. Bericat, R. Floca, S.K. Zhou, H. Shuaib, K. Farahani, K.H. Maier-Hein, S. Aylward, P. Dogra, S. Ourselin, A. Feng, MONAI: An open-source framework for deep learning in healthcare. arXiv:2211.02701.
[5] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D.G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, X. Zheng, G. Brain, TensorFlow: a system for large-scale machine learning, in: USENIX Symposium on Operating Systems Design and Implementation, 2016, pp. 265–284.
[6] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala, et al., PyTorch: an imperative style, high-performance deep learning library, in: Advances in Neural Information Processing Systems, 2019, pp. 8024–8035.
[7] M. Modat, G.R. Ridgway, Z.A. Taylor, M. Lehmann, J. Barnes, D.J. Hawkes, N.C. Fox, S. Ourselin, et al., Fast free-form deformation using graphics processing units, Comput Methods Programs Biomed 98 (3) (2010) 278–284, doi:10.1016/j.cmpb.2009.09.002.
[8] S. Klein, M. Staring, K. Murphy, M.A. Viergever, J.P.W. Pluim, Elastix: a toolbox for intensity-based medical image registration, IEEE Trans Med Imaging 29 (1) (2010) 196–205.
[9] M.J. Cardoso, M. Clarkson, M. Modat, S. Ourselin, NiftySeg: Open-source software for medical image segmentation, label fusion and cortical thickness estimation, ISBI Workshop on Open Source Medical Image Analysis Software, 2012.

[10] B. Fischl, FreeSurfer, Neuroimage 62 (2) (2012) 774–781.

[11] S.M. Smith, M. Jenkinson, M.W. Woolrich, C.F. Beckmann, T.E.J. Behrens, H. Johansen-Berg, P.R. Bannister, M. De Luca, I. Drobnjak, D.E. Flitney, R.K. Niazy, J. Saunders, J. Vickers, Y. Zhang, N. De Stefano, J.M. Brady, P.M. Matthews, Advances in functional and structural MR image analysis and implementation as FSL, in: NeuroImage, volume 23, 2004, pp. S208–S219, doi:10.1016/j.neuroimage.2004.07.051.

[12] E. Gibson, W. Li, C. Sudre, L. Fidon, D.I. Shakir, G. Wang, Z. Eaton-Rosen, R. Gray, T. Doel, Y. Hu, T. Whyntie, P. Nachev, M. Modat, D.C. Barratt, S. Ourselin, M.J. Cardoso, T. Vercauteren, et al., NiftyNet: a deep-learning platform for medical imaging, Comput Methods Programs Biomed 158 (2018) 113–122.

[13] N. Pawlowski, S.I. Ktena, M.C.H. Lee, B. Kainz, D. Rueckert, B. Glocker, M. Rajchl, et al., DLTK: State of the art reference implementations for deep learning on medical images, CoRR (2017) 1–4 arXiv:1711.06853.

[14] A. Beers, J. Brown, K. Chang, K. Hoebel, J. Patel, K.I. Ly, S.M. Tolaney, P. Brastianos, B. Rosen, E.R. Gerstner, J. Kalpathy-Cramer, Deepneuro: an open–source deep learning toolbox for neuroimaging, Neuroinformatics 19 (1) (2021) 127–140.

[15] F. Pérez-García, R. Sparks, S. Ourselin, TorchIO: a Python library for efficient loading, preprocessing, augmentation and patch-based sampling of medical images in deep learning, Comput Methods Programs Biomed 208 (2021) 106236.

[16] A. Jungo, O. Scheidegger, M. Reyes, F. Balsiger, et al., Pymia: A Python package for data handling and evaluation in deep learning-based medical image analysis, Comput Methods Programs Biomed 198 (2021) 105796, doi:10.1016/j.cmpb.2020.105796.

[17] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J. Chiang, Z. Wu, X. Ding, et al., Embracing imperfect datasets: a review of deep learning solutions for medical image segmentation, Med Image Anal 63 (2020) (2019) 101693.

[18] G. Wang, X. Liu, C. Li, Z. Xu, J. Ruan, H. Zhu, T. Meng, K. Li, N. Huang, S. Zhang, et al., A noise-robust framework for automatic segmentation of COVID-19 pneumonia lesions from CT images, IEEE Trans Med Imaging 39 (8) (2020) 2653–2663.

[19] T. Zhang, L. Yu, N. Hu, S. Lv, S. Gu, Robust medical image segmentation from non-expert annotations with tri-network, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, volume 12264 LNCS, 2020, pp. 249–258, doi:10.1007/978-3-030-59719-1_25.

[20] X. Luo, G. Wang, W. Liao, J. Chen, T. Song, Y. Chen, S. Zhang, D.N. Metaxas, S. Zhang, Semi-supervised medical image segmentation via uncertainty rectified pyramid consistency, Med Image Anal 80 (2022) 102517, doi:10.1016/j.media.2022.102517.

[21] L. Yu, S. Wang, X. Li, C.W. Fu, P.A. Heng, et al., Uncertainty-aware self-ensembling model for semi-supervised 3D left atrium segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, 2019, pp. 605–613.

[22] X. Liu, Q. Yuan, Y. Gao, K. He, S. Wang, X. Tang, J. Tang, D. Shen, Weakly supervised segmentation of COVID19 infection with scribble annotation on CT images, Pattern Recognit 122 (2022) 108341.

[23] S. Wang, C. Li, R. Wang, Z. Liu, M. Wang, T. Tan, Y. Wu, X. Liu, H. Sun, R. Yang, X. Liu, J. Chen, H. Zhou, I.B. Ayed, H. Zheng, et al., Annotation-efficient deep learning for automatic medical image segmentation, Nat Commun 12 (2021) 5915.

[24] A. Obukhov, S. Georgoulis, D. Dai, L. van Gool, et al., Gated CRF loss for weakly supervised semantic image segmentation, CoRR (2019) 1–14 arXiv:1906.04651.

[25] O. Ronneberger, P. Fischer, T. Brox, U-Net: Convolutional networks for biomedical image segmentation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, 2015, pp. 234–241.

[26] O. Oktay, J. Schlemper, L. Le Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. Mcdonagh, N.Y. Hammerla, B. Kainz, B. Glocker, D. Rueckert, et al., Attention U-Net: learning where to look for the pancreas, in: Medical Imaging with Deep Learning, 2018, pp. 1–10.

[27] Z. Zhou, M.M. Rahman Siddiquee, N. Tajbakhsh, J. Liang, et al., Unet++: A nested u-net architecture for medical image segmentation, in: MICCAI Workshop on DLMIA, volume 11045, 2018, pp. 3–11.

[28] A. Abdulkadir, S.S. Lienkamp, T. Brox, O. Ronneberger, et al., 3D U-Net: Learning dense volumetric segmentation from sparse annotation, in: International Conference on Medical Image Computing and Computer-Assisted Intervention, 2016, pp. 424–432.

[29] A.G. Roy, N. Navab, C. Wachinger, Recalibrating fully convolutional networks with spatial and channel 'squeeze and excitation' blocks, IEEE Trans Med Imaging 38 (2) (2019) 540–549.

[30] Y. Ouali, C. Hudelot, M. Tami, Semi-Supervised Semantic Segmentation with Cross-Consistency Training, in: IEEE Conference on Computer Vision and Pattern Recognition, 2020, pp. 12674–12684.

[31] X. Luo, M. Hu, W. Liao, S. Zhai, T. Song, G. Wang, S. Zhang, Scribble-Supervised Medical Image Segmentation via Dual-Branch Network and Dynamically Mixed Pseudo Labels Supervision, in: International Conference on Medical Image Computing and Computer Assisted Intervention, 2022, pp. 528–538. arXiv:2203.02106.

[32] S. Yang, G. Wang, H. Sun, X. Luo, P. Sun, K. Li, Q. Wang, S. Zhang, Learning COVID-19 pneumonia lesion segmentation from imperfect annotations via divergence-aware selective training, IEEE J Biomed Health Inform 26 (8) (2022) 3673–3684.

[33] X. Chen, Y. Yuan, G. Zeng, J. Wang, et al., Semi-supervised semantic segmentation with cross pseudo supervision, in: IEEE Conference on Computer Vision and Pattern Recognition, 2021, pp. 2613–2622.

[34] B. Han, Q. Yao, X. Yu, G. Niu, M. Xu, W. Hu, I.W. Tsang, M. Sugiyama, et al., Co-teaching: robust training of deep neural networks with extremely noisy labels, in: Advances in Neural Information Processing Systems, 2018, pp. 8527–8537. arXiv:1804.06872.

[35] Z. Zhang, M.R. Sabuncu, Generalized cross entropy loss for training deep neural networks with noisy labels, in: Advances in Neural Information Processing Systems, 2018, pp. 8778–8788. arXiv:1805.07836.

[36] A. Ghosh, H. Kumar, P.S. Sastry, Robust loss functions under label noise for deep neural networks, in: AAAI Conference on Artificial Intelligence, 2017, pp. 1919–1925.

[37] A. Tarvainen, H. Valpola, Mean teachers are better role models: weight-averaged consistency targets improve semi-supervised deep learning results, in: Advances in Neural Information Processing Systems, 2017, pp. 1195–1204. arXiv:1703.01780.

[38] Y. Grandvalet, Y. Bengio, Semi-supervised learning by entropy minimization, in: Advances in Neural Information Processing Systems, 2005, pp. 1–8.

[39] B. Kim, J.C. Ye, Mumford-shah loss functional for image segmentation with deep learning, IEEE Trans. Image Process. 29 (2020) 1856–1866, doi:10.1109/TIP.2019.2941265.

[40] M. Zhang, J. Gao, Z. Lyu, W. Zhao, Q. Wang, W. Ding, S. Wang, Z. Li, S. Cui, Characterizing label errors: confident learning for noisy-labeled image segmentation, in: International Conference on Medical Image Computing and Computer Assisted Intervention, volume 12261 LNCS, 2020, pp. 721–730.

[41] L. Li, F. Wu, S. Wang, X. Luo, C. Martin-Isla, S. Zhai, J. Zhang, Y. Liu, Z. Zhang, M.J. Ankenbrand, H. Jiang, X. Zhang, L. Wang, T.W. Arega, E. Altunok, Z. Zhao, F. Li, J. Ma, X. Yang, E. Puybareau, I. Oksuz, S. Bricq, W. Li, K. Punithakumar, S.A. Tsaftaris, L.M. Schreiber, M. Yang, G. Liu, Y. Xia, G. Wang, S. Escalera, X. Zhuang, MyoPS: a benchmark of myocardial pathology segmentation combining three-Sequence cardiac magnetic resonance images. arXiv:2201.03186.

[42] I. Luengo, M. Grammatikopoulou, R. Mohammadi, C. Walsh, C.I. Nwoye, D. Alapatt, N. Padoy, Z.-L. Ni, C.-C. Fan, G.-B. Bian, Z.-G. Hou, H. Ha, J. Wang, H. Wang, D. Guo, L. Wang, G. Wang, M. Islam, B. Giddwani, R. Hongliang, T. Pissas, C. Ravasio, M. Huber, J. Birch, J.M.N.D. Rio, L. da Cruz, C. Bergeles, H. Chen, F. Jia, N. KumarTomar, D. Jha, M.A. Riegler, P. Halvorsen, S. Bano, U. Vaghela, J. Hong, H. Ye, F. Huang, D.-H. Wang, D. Stoyanov, 2020 CATARACTS semantic segmentation challenge. arXiv:2110.10965.

[43] S. Zhai, R. Gu, W. Lei, G. Wang, et al., Myocardial edema and scar segmentation using a coarse-to-fine framework with weighted ensemble, in: MyoPS 2020: Myocardial Pathology Segmentation Combining Multi-Sequence Cardiac Magnetic Resonance Images, volume 1, Springer International Publishing, 2020, pp. 49–59.

[44] J. Wang, K. Sun, T. Cheng, B. Jiang, C. Deng, Y. Zhao, D. Liu, Y. Mu, M. Tan, X. Wang, W. Liu, B. Xiao, et al., Deep high-resolution representation learning for visual recognition, IEEE Trans Pattern Anal Mach Intell 43 (10) (2021) 3349–3364.

[45] O. Bernard, A. Lalande, C. Zotti, F. Cervenansky, X. Yang, P.-a. Heng, I. Cetin, K. Lekadir, O. Camara, M. Angel, G. Ballester, G. Sanroma, S. Napel, S. Petersen, G. Tziritas, E. Grinias, M. Khened, V.A. Kollerathu, G. Krishnamurthi, M.-m. Rohé, X. Pennec, M. Sermesant, F. Isensee, P. Jäger, K.H. Maier-hein, P.M. Full, I. Wolf, S. Engelhardt, C.F. Baumgartner, L.M. Koch, J.M. Wolterink, I. Išgum, J. Yang, Y. Hong, J. Patravali, S. Jain, O. Humbert, P.-m. Jodoin, Deep learning techniques for automatic MRI cardiac multi-structures segmentation and diagnosis : is the problem solved ? IEEE Trans Med Imaging 37 (11) (2018) 2514–2525.

[46] Z. Xiong, Q. Xia, Z. Hu, N. Huang, C. Bian, Y. Zheng, S. Vesal, N. Ravikumar, A. Maier, X. Yang, P.A. Heng, D. Ni, C. Li, Q. Tong, W. Si, E. Puybareau, Y. Khoudli, T. Géraud, C. Chen, W. Bai, D. Rueckert, L. Xu, X. Zhuang, X. Luo, S. Jia, M. Sermesant, Y. Liu, K. Wang, D. Borra, A. Masci, C. Corsi, C. De Vente, M. Veta, R. Karim, C.J. Preetha, S. Engelhardt, M. Qiao, Y. Wang, Q. Tao, M. Nuñez Garcia, O. Camara, N. Savioli, P. Lamata, J. Zhao, et al., AGlobal benchmark of algorithms for segmentinglate gadolinium-enhanced cardiac magnetic resonance imaging, Med Image Anal 67 (September 2020) (2021) 101832, doi:10.1016/j.media.2020.101832.

[47] G. Valvano, A. Leo, S.A. Tsaftaris, Learning to segment from scribbles using multi-Scale adversarial attention gates, IEEE Trans Med Imaging 40 (8) (2021) 1990–2001.

[48] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto, T. Kobayashi, K.I. Komatsu, M. Matsui, H. Fujita, Y. Kodera, K. Doi, Development of a digital image database for chest radiographs with and without a lung nodule: receiver operating characteristic analysis of radiologists' detection of pulmonary nodules, American Journal of Roentgenology 174 (1) (2000) 71–74, doi:10.2214/ajr.174.1.1740071.

[49] Q. Duan, G. Wang, R. Wang, C. Fu, X. Li, M. Gong, X. Liu, Q. Xia, X. Huang, Z. Hu, N. Huang, S. Zhang, et al., Sensecare: a research platform for medical image informatics and interactive 3D visualization, CoRR (2020) arXiv:2004.07031.

[50] R. Gu, G. Wang, T. Song, R. Huang, M. Aertsen, J. Deprest, S. Ourselin, T. Vercauteren, S. Zhang, et al., CA-Net: Comprehensive attention convolutional neural networks for explainable medical image segmentation, IEEE Trans Med Imaging 40 (2) (2021) 699–711.

[51] G. Wang, M. Aertsen, J. Deprest, S. Ourselin, T. Vercauteren, S. Zhang, et al., Uncertainty-guided efficient interactive refinement of fetal brain segmentation from stacks of MRI slices, in: International Conference on Medical Image Computing and Computer Assisted Intervention, volume 4, 2020, pp. 279–288.

[52] Z. Liu, C. Sun, H. Wang, Z. Li, Y. Gao, W. Lei, S. Zhang, G. Wang, S. Zhang, Automatic segmentation of organs-at-risks of nasopharynx cancer and lung cancer by cross-layer attention fusion network with TELD-Loss, Med Phys 48 (11) (2021) 6987–7002.

[53] G. Wang, S. Zhai, G. Lasio, B. Zhang, B. Yi, S. Chen, T.J. Macvittie, D. Metaxas, J. Zhou, S. Zhang, Semi-supervised segmentation of radiation-induced pulmonary fibrosis from lung CT scans with multi-scale guided dense attention, IEEE Trans Med Imaging 41 (3) (2022) 531–542.

[54] L. Chen, P. Bentley, K. Mori, K. Misawa, M. Fujiwara, D. Rueckert, et al., Self-supervised learning for medical image analysis using image context restoration, Med Image Anal 58 (2019) 101539, doi:10.1016/j.media.2019.101539.

[55] J. Zhu, Y. Li, Y. Hu, K. Ma, S.K. Zhou, Y. Zheng, et al., Rubik's cube+: a self-supervised feature learning framework for 3D medical image analysis, Med Image Anal 64 (2020) 101746, doi:10.1016/j.media.2020.101746.

[56] H. Guan, M. Liu, Domain adaptation for medical image analysis: asurvey, IEEE Trans. Biomed. Eng. 69 (3) (2022) 1173–1185 arXiv:2102.09508.

[57] J. Wu, R. Gu, G. Dong, G. Wang, S. Zhang, FPL-UDA: filtered pseudo label-based unsupervised cross-modality adaptation for vestibular schwannoma segmentation, in: IEEE International Symposium on Biomedical Imaging (ISBI), IEEE, 2022, pp. 1–5.