Project-MONAI/MONAILabel

Issues gathered from the github repository regarding MONAILabel

More documentation on using local (non-zoo) bundles with MONAILabel

**Issue No. 1490 opened by wyli on 2023-07-14 at 15:33:21 Type: None**

bring your own bundle from local directories

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Error while training with MONAILabel's deepedit

**Issue No. 1489 opened by lukasvanderstricht on 2023-07-14 at 06:51:00 Type: None**

Dear all  
  
I am currently using 3D Slicer and its MONAILabel extension to train a  
segmentation model using the DeepEdit model from the predefined  
radiology app. Both manual segmentation and training have been going  
smoothly up till now and the automatic segmentation functionality seems  
to be doing its job. However, when I want to further train the model at  
this point, without having added any new labels (so just starting the  
training process again), I always get one of the two following errors  
  
1. Exit code -9  
  
> \[2023-07-13 08:52:08,765\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.utils.sessions:51) - Session Path:  
> /home/dellxpsazdelta/.cache/monailabel/sessions \[2023-07-13  
> 08:52:08,765\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.utils.sessions:52) - Session Expiry (max): 3600  
> \[2023-07-13 08:52:08,765\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:353) - Train Request (input):  
> {'model': 'deepedit', 'name': 'train\_01', 'pretrained': 1, 'device':  
> 'cuda', 'max\_epochs': 50, 'early\_stop\_patience': -1, 'val\_split': 0.2,  
> 'train\_batch\_size': 1, 'val\_batch\_size': 1, 'multi\_gpu': True, 'gpus':  
> 'all', 'dataset': 'CacheDataset', 'dataloader': 'ThreadDataLoader',  
> 'client\_id': 'user-xyz', 'local\_rank': 0} \[2023-07-13 08:52:08,766\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:363) - CUDA\_VISIBLE\_DEVICES: 0,1  
> \[2023-07-13 08:52:08,767\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:368) - Distributed/Multi GPU is  
> limited \[2023-07-13 08:52:08,767\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:381) - Distributed Training =  
> FALSE \[2023-07-13 08:52:08,767\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:408) - 0 - Train Request (final):  
> {'name': 'train\_01', 'pretrained': 1, 'device': 'cuda', 'max\_epochs':  
> 50, 'early\_stop\_patience': -1, 'val\_split': 0.2, 'train\_batch\_size':  
> 1, 'val\_batch\_size': 1, 'multi\_gpu': False, 'gpus': 'all', 'dataset':  
> 'CacheDataset', 'dataloader': 'ThreadDataLoader', 'model': 'deepedit',  
> 'client\_id': 'user-xyz', 'local\_rank': 0, 'run\_id': '20230713\_0852'}  
> \[2023-07-13 08:52:08,768\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:504) - 0 - Using Device: cuda;  
> IDX: None \[2023-07-13 08:52:08,768\] \[24408\] \[MainThread\]  
> \[INFO\] (monailabel.tasks.train.basic\_train:331) - Total Records for  
> Training: 5 \[2023-07-13 08:52:08,768\] \[24408\] \[MainThread\]  
> \[INFO\] (monailabel.tasks.train.basic\_train:332) - Total Records for  
> Validation: 1 Loading dataset: 0%\| \| 0/1 \[00:00\<?, ?it/s\] Loading  
> dataset: 100%\|██████████\| 1/1 \[00:10\<00:00, 10.10s/it\] Loading  
> dataset: 100%\|██████████\| 1/1 \[00:10\<00:00, 10.10s/it\]  
> \[2023-07-13 08:53:05,649\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:275) - 0 - Records for Validation:  
> 1 \[2023-07-13 08:53:05,748\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:265) - 0 - Adding Validation to  
> run every '1' interval \[2023-07-13 08:53:05,749\] \[24408\]  
> \[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:591) - 0 -  
> Load Path  
> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet/train\_01/model.pt  
> Loading dataset: 0%\| \| 0/5 \[00:00\<?, ?it/s\] Loading dataset:  
> 20%\|██ \| 1/5 \[00:05\<00:20, 5.20s/it\] Loading dataset: 40%\|████  
> \| 2/5 \[00:10\<00:15, 5.05s/it\] Loading dataset: 60%\|██████ \| 3/5  
> \[00:13\<00:08, 4.37s/it\] Loading dataset: 80%\|████████ \| 4/5  
> \[00:18\<00:04, 4.36s/it\] Loading dataset: 100%\|██████████\| 5/5  
> \[00:23\<00:00, 4.62s/it\] Loading dataset: 100%\|██████████\| 5/5  
> \[00:23\<00:00, 4.63s/it\] \[2023-07-13 08:53:29,092\] \[24408\]  
> \[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:227) - 0 -  
> Records for Training: 5 \[2023-07-13 08:53:29,098\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:697) -  
> Engine run resuming from iteration 0, epoch 0 until 50 epochs  
> \[2023-07-13 08:53:30,700\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:138) - Restored all variables  
> from  
> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet/train\_01/model.pt  
> \[2023-07-13 08:53:38,945\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 1/50, Iter: 1/5  
> -- train\_loss: 0.8791 2023-07-13 08:53:40,341 - INFO - Number of  
> simulated clicks: 11 \[2023-07-13 08:53:41,639\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:269) -  
> Epoch: 1/50, Iter: 2/5 -- train\_loss: 0.8105 \[2023-07-13  
> 08:53:42,155\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 1/50, Iter: 3/5  
> -- train\_loss: 0.8015 \[2023-07-13 08:53:42,632\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:269) -  
> Epoch: 1/50, Iter: 4/5 -- train\_loss: 0.8339 2023-07-13 08:53:45,352 -  
> INFO - Number of simulated clicks: 9 \[2023-07-13 08:53:46,177\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 1/50, Iter: 5/5  
> -- train\_loss: 0.8071 \[2023-07-13 08:53:46,433\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:265) -  
> Got new best metric of train\_dice: 0.678305447101593 \[2023-07-13  
> 08:53:46,434\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:85) - Current learning rate:  
> 0.0001 \[2023-07-13 08:53:46,434\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:199) - Epoch\[1\] Metrics --  
> left psoas\_dice: 0.7228 right psoas\_dice: 0.6338 train\_dice: 0.6783  
> \[2023-07-13 08:53:46,434\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:209) - Key metric: train\_dice  
> best value: 0.678305447101593 at epoch: 1 \[2023-07-13 08:53:46,435\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:697) - Engine run resuming  
> from iteration 0, epoch 0 until 1 epochs 2023-07-13 08:53:48,278 -  
> INFO - Number of simulated clicks: 10 \[2023-07-13 08:53:49,005\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:265) - Got new best metric  
> of val\_mean\_dice: 0.6918515563011169 \[2023-07-13 08:53:49,005\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:199) - Epoch\[1\] Metrics --  
> left psoas\_dice: 0.7021 right psoas\_dice: 0.6816 val\_mean\_dice: 0.6919  
> \[2023-07-13 08:53:49,005\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:209) - Key metric:  
> val\_mean\_dice best value: 0.6918515563011169 at epoch: 1 \[2023-07-13  
> 08:53:51,338\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.handler:86) - New Model published:  
> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet/train\_01/model.pt  
> =\> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet.pt  
> \[2023-07-13 08:53:51,339\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:765) - Epoch\[1\] Complete.  
> Time taken: 00:00:05 \[2023-07-13 08:53:51,339\] \[24408\]  
> \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:778) - Engine run complete.  
> Time taken: 00:00:05 \[2023-07-13 08:53:52,354\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:765) -  
> Epoch\[1\] Complete. Time taken: 00:00:22 2023-07-13 08:53:54,353 -  
> INFO - Number of simulated clicks: 3 \[2023-07-13 08:53:55,735\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 2/50, Iter: 1/5  
> -- train\_loss: 0.7936 \[2023-07-13 08:53:56,209\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:269) -  
> Epoch: 2/50, Iter: 2/5 -- train\_loss: 0.8046 2023-07-13 08:53:57,274 -  
> INFO - Number of simulated clicks: 8 \[2023-07-13 08:53:58,583\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 2/50, Iter: 3/5  
> -- train\_loss: 0.7968 \[2023-07-13 08:53:59,060\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:269) -  
> Epoch: 2/50, Iter: 4/5 -- train\_loss: 0.8337 \[2023-07-13  
> 08:53:59,851\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:269) - Epoch: 2/50, Iter: 5/5  
> -- train\_loss: 0.8011 \[2023-07-13 08:53:59,853\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:265) -  
> Got new best metric of train\_dice: 0.8020626306533813 \[2023-07-13  
> 08:53:59,853\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:85) - Current learning rate:  
> 0.0001 \[2023-07-13 08:53:59,853\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:199) - Epoch\[2\] Metrics --  
> left psoas\_dice: 0.7723 right psoas\_dice: 0.8318 train\_dice: 0.8021  
> \[2023-07-13 08:53:59,853\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedTrainer:209) - Key metric: train\_dice  
> best value: 0.8020626306533813 at epoch: 2 \[2023-07-13 08:53:59,854\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:697) - Engine run resuming  
> from iteration 0, epoch 1 until 2 epochs 2023-07-13 08:54:01,733 -  
> INFO - Number of simulated clicks: 9 \[2023-07-13 08:54:02,684\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:265) - Got new best metric  
> of val\_mean\_dice: 0.709175705909729 \[2023-07-13 08:54:02,685\]  
> \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:199) - Epoch\[2\] Metrics --  
> left psoas\_dice: 0.7219 right psoas\_dice: 0.6965 val\_mean\_dice: 0.7092  
> \[2023-07-13 08:54:02,685\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:209) - Key metric:  
> val\_mean\_dice best value: 0.709175705909729 at epoch: 2 \[2023-07-13  
> 08:54:05,116\] \[24408\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.handler:86) - New Model published:  
> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet/train\_01/model.pt  
> =\> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet.pt  
> \[2023-07-13 08:54:05,116\] \[24408\] \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:765) - Epoch\[2\] Complete.  
> Time taken: 00:00:05 \[2023-07-13 08:54:05,116\] \[24408\]  
> \[MainThread\] \[INFO\]  
> (ignite.engine.engine.SupervisedEvaluator:778) - Engine run complete.  
> Time taken: 00:00:05 \[2023-07-13 08:54:05,418\] \[24408\]  
> \[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:765) -  
> Epoch\[2\] Complete. Time taken: 00:00:13 \[2023-07-13 08:54:14,842\]  
> \[26118\] \[ThreadPoolExecutor-0\_0\] \[INFO\]  
> (monailabel.utils.async\_tasks.utils:76) - Return code: -9  
  
2. Exit code 1  
  
> \[2023-07-13 12:20:04,630\] \[23149\] \[MainThread\] \[INFO\]  
> (monailabel.utils.async\_tasks.task:36) - Train request: {'model':  
> 'deepedit', 'name': 'train\_01', 'pretrained': 1, 'device': 'cuda',  
> 'max\_epochs': 50, 'early\_stop\_patience': -1, 'val\_split': 0.2,  
> 'train\_batch\_size': 1, 'val\_batch\_size': 1, 'multi\_gpu': True, 'gpus':  
> 'all', 'dataset': 'CacheDataset', 'dataloader': 'ThreadDataLoader',  
> 'client\_id': 'user-xyz'} \[2023-07-13 12:20:04,632\] \[23149\]  
> \[ThreadPoolExecutor-0\_0\] \[INFO\]  
> (monailabel.utils.async\_tasks.utils:58) - COMMAND::  
> /opt/conda/bin/python -m monailabel.interfaces.utils.app -m train -r  
> {"model":"deepedit","name":"train\_01","pretrained":1,"device":"cuda","max\_epochs":50,"early\_stop\_patience":-1,"val\_split":0.2,"train\_batch\_size":1,"val\_batch\_size":1,"multi\_gpu":true,"gpus":"all","dataset":"CacheDataset","dataloader":"ThreadDataLoader","client\_id":"user-xyz"}  
> \[2023-07-13 12:20:05,144\] \[5998\] \[MainThread\] \[INFO\]  
> (main:38) - Initializing App from:  
> /home/dellxpsazdelta/radiology\_psoas\_azd; studies:  
> /home/dellxpsazdelta/psoas-azd-images/train-images; conf: {'models':  
> 'deepedit'} \[2023-07-13 12:20:30,698\] \[5998\] \[MainThread\]  
> \[INFO\] (monailabel.utils.others.class\_utils:36) - Subclass for  
> MONAILabelApp Found: \<class 'main.MyApp'\> \[2023-07-13  
> 12:20:30,748\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.others.class\_utils:36) - Subclass for TaskConfig  
> Found: \<class 'lib.configs.segmentation.Segmentation'\> \[2023-07-13  
> 12:20:30,749\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.others.class\_utils:36) - Subclass for TaskConfig  
> Found: \<class 'lib.configs.deepgrow\_3d.Deepgrow3D'\> \[2023-07-13  
> 12:20:30,770\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.others.class\_utils:36) - Subclass for TaskConfig  
> Found: \<class 'lib.configs.deepedit.DeepEdit'\> \[2023-07-13  
> 12:20:30,771\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.others.class\_utils:36) - Subclass for TaskConfig  
> Found: \<class 'lib.configs.segmentation\_spleen.SegmentationSpleen'\>  
> \[2023-07-13 12:20:30,772\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.others.class\_utils:36) - Subclass for TaskConfig  
> Found: \<class 'lib.configs.deepgrow\_2d.Deepgrow2D'\> \[2023-07-13  
> 12:20:30,773\] \[5998\] \[MainThread\] \[INFO\] (main:83) - +++ Adding  
> Model: deepedit =\> lib.configs.deepedit.DeepEdit \[2023-07-13  
> 12:20:32,241\] \[5998\] \[MainThread\] \[INFO\]  
> (lib.configs.deepedit:145) - EPISTEMIC Enabled: 0; Samples: 5  
> \[2023-07-13 12:20:32,241\] \[5998\] \[MainThread\] \[INFO\]  
> (lib.configs.deepedit:149) - TTA Enabled: 0; Samples: 5 \[2023-07-13  
> 12:20:32,241\] \[5998\] \[MainThread\] \[INFO\] (main:87) - +++ Using  
> Models: \['deepedit'\] \[2023-07-13 12:20:32,241\] \[5998\]  
> \[MainThread\] \[INFO\] (monailabel.interfaces.app:126) - Init  
> Datastore for: /home/dellxpsazdelta/psoas-azd-images/train-images  
> \[2023-07-13 12:20:32,242\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.datastore.local:125) - Auto Reload: False; Extensions:  
> \['.nii.gz', \'.nii', '.nrrd', \'.jpg', '.png', \'.tif', '.svs',  
> \'.xml'\] \[2023-07-13 12:20:32,300\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.datastore.local:540) - Invalidate count: 0 \[2023-07-13  
> 12:20:32,300\] \[5998\] \[MainThread\] \[INFO\] (main:112) - +++  
> Adding Inferer:: deepedit =\> \<lib.infers.deepedit.DeepEdit object at  
> 0x7f395c5280d0\> \[2023-07-13 12:20:32,300\] \[5998\] \[MainThread\]  
> \[INFO\] (main:112) - +++ Adding Inferer:: deepedit\_seg =\>  
> \<lib.infers.deepedit.DeepEdit object at 0x7f395c12b990\> \[2023-07-13  
> 12:20:32,301\] \[5998\] \[MainThread\] \[INFO\] (main:161) - +++  
> Adding Trainer:: deepedit =\> \<lib.trainers.deepedit.DeepEdit object  
> at 0x7f395c12b7d0\> \[2023-07-13 12:20:32,301\] \[5998\]  
> \[MainThread\] \[INFO\] (monailabel.utils.sessions:51) - Session Path:  
> /home/dellxpsazdelta/.cache/monailabel/sessions \[2023-07-13  
> 12:20:32,301\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.utils.sessions:52) - Session Expiry (max): 3600  
> \[2023-07-13 12:20:32,301\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:353) - Train Request (input):  
> {'model': 'deepedit', 'name': 'train\_01', 'pretrained': 1, 'device':  
> 'cuda', 'max\_epochs': 50, 'early\_stop\_patience': -1, 'val\_split': 0.2,  
> 'train\_batch\_size': 1, 'val\_batch\_size': 1, 'multi\_gpu': True, 'gpus':  
> 'all', 'dataset': 'CacheDataset', 'dataloader': 'ThreadDataLoader',  
> 'client\_id': 'user-xyz', 'local\_rank': 0} \[2023-07-13 12:20:32,301\]  
> \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:363) - CUDA\_VISIBLE\_DEVICES: None  
> \[2023-07-13 12:20:32,302\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:368) - Distributed/Multi GPU is  
> limited \[2023-07-13 12:20:32,302\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:381) - Distributed Training =  
> FALSE \[2023-07-13 12:20:32,302\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:408) - 0 - Train Request (final):  
> {'name': 'train\_01', 'pretrained': 1, 'device': 'cuda', 'max\_epochs':  
> 50, 'early\_stop\_patience': -1, 'val\_split': 0.2, 'train\_batch\_size':  
> 1, 'val\_batch\_size': 1, 'multi\_gpu': False, 'gpus': 'all', 'dataset':  
> 'CacheDataset', 'dataloader': 'ThreadDataLoader', 'model': 'deepedit',  
> 'client\_id': 'user-xyz', 'local\_rank': 0, 'run\_id': '20230713\_1220'}  
> \[2023-07-13 12:20:32,302\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:504) - 0 - Using Device: cuda;  
> IDX: None \[2023-07-13 12:20:32,303\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:331) - Total Records for Training:  
> 7 \[2023-07-13 12:20:32,303\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:332) - Total Records for  
> Validation: 2 Loading dataset: 0%\| \| 0/2 \[00:00\<?, ?it/s\] Loading  
> dataset: 50%\|█████ \| 1/2 \[00:09\<00:09, 9.83s/it\] Loading dataset:  
> 100%\|██████████\| 2/2 \[00:13\<00:00, 6.29s/it\] Loading dataset:  
> 100%\|██████████\| 2/2 \[00:13\<00:00, 6.82s/it\] \[2023-07-13  
> 12:21:44,249\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:275) - 0 - Records for Validation:  
> 2 \[2023-07-13 12:21:44,258\] \[5998\] \[MainThread\] \[INFO\]  
> (monailabel.tasks.train.basic\_train:265) - 0 - Adding Validation to  
> run every '1' interval \[2023-07-13 12:21:44,258\] \[5998\]  
> \[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:591) - 0 -  
> Load Path  
> /home/dellxpsazdelta/radiology\_psoas\_azd/model/deepedit\_dynunet/train\_01/model.pt  
> Loading dataset: 0%\| \| 0/7 \[00:00\<?, ?it/s\] Loading dataset:  
> 14%\|█▍ \| 1/7 \[00:04\<00:25, 4.25s/it\] Loading dataset: 29%\|██▊ \|  
> 2/7 \[00:08\<00:21, 4.35s/it\] Loading dataset: 43%\|████▎ \| 3/7  
> \[00:12\<00:16, 4.16s/it\] Loading dataset: 57%\|█████▋ \| 4/7  
> \[00:16\<00:11, 3.93s/it\] Loading dataset: 71%\|███████▏ \| 5/7  
> \[00:25\<00:11, 5.88s/it\] Loading dataset: 71%\|███████▏ \| 5/7  
> \[00:29\<00:11, 5.87s/it\] Traceback (most recent call last): File  
> "/opt/conda/lib/python3.7/site-packages/monai/transforms/transform.py",  
> line 89, in apply\_transform return \\_apply\_transform(transform, data,  
> unpack\_items) File  
> "/opt/conda/lib/python3.7/site-packages/monai/transforms/transform.py",  
> line 53, in \\_apply\_transform return transform(parameters) File  
> "/opt/conda/lib/python3.7/site-packages/monai/apps/deepedit/transforms.py",  
> line 99, in call label = np.zeros(d\[key\].shape)  
> numpy.core.\\_exceptions.MemoryError: Unable to allocate 502. MiB for  
> an array with shape (512, 512, 251) and data type float64 The above  
> exception was the direct cause of the following exception: Traceback  
> (most recent call last): File "/opt/conda/lib/python3.7/runpy.py",  
> line 193, in \\_run\_module\_as\_main "main", mod\_spec) File  
> "/opt/conda/lib/python3.7/runpy.py", line 85, in \\_run\_code exec(code,  
> run\_globals) File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/interfaces/utils/app.py",  
> line 132, in run\_main() File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/interfaces/utils/app.py",  
> line 117, in run\_main result = a.train(request) File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/interfaces/app.py",  
> line 380, in train result = task(request, self.datastore()) File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/tasks/train/basic\_train.py",  
> line 382, in call res = self.train(0, world\_size, req, datalist) File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/tasks/train/basic\_train.py",  
> line 428, in train context.trainer = self.\\_create\_trainer(context)  
> File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/tasks/train/basic\_train.py",  
> line 575, in \\_create\_trainer  
> train\_data\_loader=self.train\_data\_loader(context), File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/tasks/train/basic\_train.py",  
> line 226, in train\_data\_loader dataset, datalist =  
> self.\\_dataset(context, context.train\_datalist) File  
> "/opt/conda/lib/python3.7/site-packages/monailabel/tasks/train/basic\_train.py",  
> line 200, in \\_dataset if context.dataset\_type == "CacheDataset" File  
> "/opt/conda/lib/python3.7/site-packages/monai/data/dataset.py", line  
> 723, in init self.set\_data(data) File  
> "/opt/conda/lib/python3.7/site-packages/monai/data/dataset.py", line  
> 748, in set\_data self.\\_cache = \\_compute\_cache() File  
> "/opt/conda/lib/python3.7/site-packages/monai/data/dataset.py", line  
> 737, in \\_compute\_cache return self.\\_fill\_cache() File  
> "/opt/conda/lib/python3.7/site-packages/monai/data/dataset.py", line  
> 761, in \\_fill\_cache desc="Loading dataset", File  
> "/opt/conda/lib/python3.7/site-packages/tqdm/std.py", line 1195, in  
> iter for obj in iterable: File  
> "/opt/conda/lib/python3.7/multiprocessing/pool.py", line 748, in next  
> raise value File "/opt/conda/lib/python3.7/multiprocessing/pool.py",  
> line 121, in worker result = (True, func(\\*args, \\*\\*kwds)) File  
> "/opt/conda/lib/python3.7/site-packages/monai/data/dataset.py", line  
> 777, in \\_load\_cache\_item item = apply\_transform(\\_xform, item) File  
> "/opt/conda/lib/python3.7/site-packages/monai/transforms/transform.py",  
> line 113, in apply\_transform raise RuntimeError(f\"applying transform  
> {transform}\") from e RuntimeError: applying transform  
> \<monai.apps.deepedit.transforms.NormalizeLabelsInDatasetd object at  
> 0x7f393a2fdfd0\> \[2023-07-13 12:22:24,359\] \[23149\]  
> \[ThreadPoolExecutor-0\_0\] \[INFO\]  
> (monailabel.utils.async\_tasks.utils:76) - Return code: 1  
  
It seems weird to me that without changing anything (such as adding new  
labels), the training suddenly starts to systematically fail while it  
was working fine before. Does anyone have any clue as to why these  
errors occur?  
  
Thanks in advance!

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Monai label server add API to return number of requests in the queue

**Issue No. 1487 opened by AHarouni on 2023-07-10 at 20:20:14 Type: None**

\*\*Please describe.\*\* My model is very large and takes up to 60 seconds  
on my gpu. I run with \--workers=1 to avoid having 2 requests run on the  
same gpu and get into out of memory. Now when multiple users each run a  
request they are in a queue that is some times too long.  
  
\*\*Describe the solution you\'d like\*\* We should have an api to show how  
many requests are ahead of them in the queue and give an estimate on the  
wait time This should take into account if we have multiple gpus.  
Basically this allows the UI to set the users expectations on the wait  
time  
  
\*\*Describe alternatives you\'ve considered\*\* None

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Monai label should take advantage of all gpus while running inference

**Issue No. 1486 opened by AHarouni on 2023-07-10 at 18:43:39 Type: None**

\\*\\*Is your feature request related to a problem? \\*\\* When we run monai  
label with 1 model and 4 gpus. It should be the servers responsibility  
to determine with gpu to use. Currently the user in his request  
specifies which gpu to use. in my mind the user can be a radiologist  
using ohif interface who has no clue if there is even any gpus in the  
back  
  
\*\*Describe the solution you\'d like\*\* Monai label would start with a  
parameter of gpu=0,2 or all by default Monai label should spin up  
workers as inference request come so each worker runs inference on  
different gpu to make use of all gpus Monai label should also handle the  
gpus for batch inference request ( this is new feature in 0.7) but I  
think it uses all gpus. Some how monai label needs to manage the gpus  
available to it in a smart way  
  
\*\*Describe alternatives you\'ve considered\*\* start monai label with  
\--worker flag that is dependent on the number of gpus. \--\> this  
creates multiple process that are have consciences PIDs in my app I  
added  
  
 class SAMMPF(BasicInferTask):  
 """  
 This provides Inference Engine for pre-trained spleen segmentation (UNet) model over MSD Dataset.   
 """  
 def \_\_call\_\_(  
 self, request, callbacks = None  
 ) -> Union[Dict, Tuple[str, Dict[str, Any]]]:  
 request["device"]="cuda:"+self.gpu\_no  
 return super().\_\_call\_\_(request, callbacks)  
   
 def \_\_init\_\_(  
 self,path,network=None,  
 target\_spacing=(1.5, 1.5, 1.5),type=InferType.SEGMENTATION,  
 labels=None,dimension=3,  
 description="A test",  
 \*\*kwargs,  
 ):  
 super().\_\_init\_\_(path=path,network=network,type=type,labels=labels,  
 dimension=dimension,description=description,\*\*kwargs,)  
   
 import os  
 NUMBER\_OF\_WORKERS=4 <-- set as the number of gpus which is equal the number of workers   
 pid=os.getpid()  
 self.gpu\_no=str(pid%NUMBER\_OF\_WORKERS)  
  
 def pre\_transforms(self, data=None) -> Sequence[Callable]:  
 return [  
 LoadImaged(keys="image"),  
 ....  
 EnsureTyped(keys="image",device='cuda:'+self.gpu\_no),  
 ]   
  
\*\*Additional context\*\* For Batch inference new in 0.7 we do start  
workers and loop use all the gpus. This is the great and is the correct  
behavior until inference requests starts coming in.  
  
For example  
  
- say the batch infer for 1k patients will take 10 hours using all 4  
 gpus.  
- now if 4 users starting sending infer requests for the same model on  
 new patients will they wait for 10 hours till the batch infer  
 finishes? you could say the batch infer with use all gpu -1, but  
 what if no user sends any requests then you have wasted that last  
 gpu  
- some how monai label need to be able to scale down the works for  
 batch inference  
  
Another option would be to return a descriptive response of server is  
busy doing batch work, try again in X hours

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

3D Slicer fetching wrong sample due to local cache

**Issue No. 1484 opened by elialombardo on 2023-07-06 at 07:25:29 Type: None**

I am using MONAI label with the radiology app and 3D Slicer to annotate  
and auto-segment MRI images. I want to create a new model for every  
patient so I re-start the MONAI label server and specify a new study  
directory. Recently I had the problem that Slicer was fetching samples  
from a previous patient even though that patient was not selected\...  
talking with \@nvahmadi, who has had a similar issue, it turned out that  
the MONAI label client UI checks whether a file of that name is already  
in Slicer\'s cache. If so, it saves the download time from the server  
and uses the image from Slicer\'s cache. As my two patients had the same  
filenames, Slicer was using the image from the cache.  
  
A nice solution would be that the MONAI Label settings in Slicer\'s  
application settings panel include a checkbox \"Use Slicer local cache  
to avoid downloads\". Alternatively, the UI could have a button \"Clear  
Slicer local cache\" that can be pressed if such issues occur.  
  
My workaround at the moment is to manually empty the Slicer cache in the  
Application Settings menu. Thanks to \@nvahmadi for the hint!

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

the first '/' is missing if we use 'Utils.getFileName' on Linux system

**Issue No. 1480 opened by hubutui on 2023-06-28 at 07:00:04 Type: None**

The URI in qupath looks like `file:/home/username/myimage.png` on Linux,  
and `file:/C:/Users/username/Desktop/myimage.png` on Windows.So  
`Utils.getFileName` works for Windows but not Linux. We would get  
`home/username/myimage.png`. This could leads to bugs when we use the  
result return fromt this method, eg:  
<https://github.com/Project-MONAI/MONAILabel/blob/d95c8a2ea6c8b9dd0bc6d9a06cd3bd8767df3da0/plugins/qupath/src/main/java/qupath/lib/extension/monailabel/commands/RunInference.java#L207-L208>  
quick fix, insert these lines:  
  
``` java  
if (System.getProperty("os.name").toLowerCase().contains("linux")) {  
 file = "/" + file;  
}  
```  
  
just before  
<https://github.com/Project-MONAI/MONAILabel/blob/d95c8a2ea6c8b9dd0bc6d9a06cd3bd8767df3da0/plugins/qupath/src/main/java/qupath/lib/extension/monailabel/Utils.java#L34>  
  
Maybe we should also check for MacOS or other non-Windows systems? How  
about this:  
  
``` java  
if (!System.getProperty("os.name").toLowerCase().contains("windows")) {  
 file = "/" + file;  
}  
```

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Use existing segmentation as segmentNode

**Issue No. 1477 opened by EvaHeffinck on 2023-06-21 at 14:03:17 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* If  
you have a volume node and a segmentation node already available in  
slicer and you go to the MONAILabel module, the existing volume node  
gets recognized and used as master volume, but a new segmentation node  
is always created. Due to this it is not possible to start annotating a  
segment, close slicer and continue later on with the same segmentation  
or to get a quality check by a second annotator who can then submit the  
label.  
  
A possible workaround is now to drag the existing segment in the new  
segmentation node and remove the new empty segment, but it would be nice  
to be able to just save everything to an .mrb and continue later.  
  
\*\*Describe the solution you\'d like\*\* This issue can be fixed by a small  
update in `MONAILabelWidget.createSegmentNode()` that checks if a  
segmentation node with the correct name is already available.  
  
 def createSegmentNode(self):  
 if self.\_volumeNode is None:  
 return  
 if self.\_segmentNode is None:  
 name = "segmentation\_" + self.\_volumeNode.GetName()  
 if slicer.util.getFirstNodeByName(name):  
 self.\_segmentNode = slicer.util.getFirstNodeByName(name)  
 else:  
 self.\_segmentNode = slicer.mrmlScene.AddNewNodeByClass("vtkMRMLSegmentationNode")  
 self.\_segmentNode.SetReferenceImageGeometryParameterFromVolumeNode(self.\_volumeNode)  
 self.\_segmentNode.SetName(name)

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Monailabel is ignoring arguments (issue with PYTHONPATH?)

**Issue No. 1469 opened by dorkylever on 2023-06-11 at 07:29:27 Type: None**

\*\*Describe the bug\*\* Hi All,  
  
I\'m trying to set up an easy install of monailabel on a PBS-based  
computing cluster (AWS is unavailable on our computing environment).\  
The computing enviroment mainly loads packages by the creation and use  
of Tool Command Language (TCL) modules which typically contains a pip  
install e.g.  
`pip3 install -I --prefix="${APP\_PATH}" monailabel==0.6.0 --no-warn-script-location`  
  
and then the required PATHs are adjusted accordingly:  
  
> mkdir -p \${APP\_MODULE\_PATH} cat\<`<EOF >`{=html}  
> ${APP\_MODULE\_PATH}/${APP\_VERSION} #%Module1.0 source  
> /opt/Modules/extensions/extensions.tcl soft-prereq python3/3.10.0  
> prepend-path PYTHONPATH /apps/python3.10.0/ prepend-path PYTHONPATH  
> \${APP\_PATH}/lib/python3.10/site-packages prepend-path PATH  
> \${APP\_PATH}/bin EOF  
  
However, when I load the module and run the monailabel command line, the  
command is reached but there is no following output, I supect due to  
something missing PYTHONPATHS. Example runs are below:  
  
\*\*Server logs\*\* bash file used for qsubbing:  
  
> \`\[kd0793@gadi-login-05 0.6.0\]\$ cat bash\_file.sh #!/bin/bash #PBS  
> -P nm24 #PBS -q gpuvolta #PBS -l walltime=48:00:00 #PBS -l mem=100GB  
> #PBS -l jobfs=200GB #PBS -l ngpus=4 #PBS -l ncpus=48 #PBS  
> -lstorage=gdata/if89  
  
`module unload LAMA python3` `module load monailabel/0.6.0`  
`python3 -c "import torch; print(torch.cuda.is\_available())"`  
`python -c 'import monai; monai.config.print\_debug\_info()'`  
`monailabel --help`  
  
output (standard output)  
  
True Using  
PYTHONPATH=/g/data/if89/apps/monailabel:/g/data/if89/apps/monailabel/0.6.0/lib/python3.10/site-packages:/apps/python3.10.0/  
  
\*\*Expected behavior\*\* Should bring up the help arguments for monailabel

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

[Slicer] Hide button for control points in the SmartEdit/DeepGrow section

**Issue No. 1463 opened by MattTheCuber on 2023-06-08 at 20:20:11 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* It is  
annoying to paint with colored dots on the images.  
  
\*\*Describe the solution you\'d like\*\* A simple hide button to toggle the  
visibility of these control points.  
  
\*\*Describe alternatives you\'ve considered\*\* None  
  
\*\*Additional context\*\* Open 3D Slicer, open the MONAI Label module,  
expand the SmartEdit/DeepGrow section, add some control points. Once you  
have done this, it is hard to see under the control points if you wanted  
to view/paint the images under it.  
  
Something like this: Before: ![vncviewer\_R9tx4XmaY6 -  
Copy](https://github.com/Project-MONAI/MONAILabel/assets/32849887/5ddf435a-a977-454f-a78b-a35acb70b3e5)  
After:  
![vncviewer\_R9tx4XmaY6](https://github.com/Project-MONAI/MONAILabel/assets/32849887/deb5df06-835a-48ed-b3ef-4d1a3efb6922)

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Import multiple labels

**Issue No. 1459 opened by haochenz23 on 2023-06-02 at 21:16:50 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* I\'m  
trying to import existing mask files(label), bringing them into the the  
MONAI plugin and continue working with them. Currently I have three  
labels, and I tried to import three masks that correlated with each  
label, but the plugin associate the mask only with one label with each  
import. \*\*What I\'ve tried.\*\* I attempted to select the labels in  
\"Segment Editor\", and import the file in \"Tools\". The selection  
doesn\'t seem to be effective.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

MONAI Label extension in Qupath (pathology app startup failed)

**Issue No. 1457 opened by CZXHPB on 2023-06-02 at 10:43:13 Type: None**

Dear friends： I want to install MONAI Label extension in the Qupath for  
efficient annotation, but the extension cannot work well (pathology  
application startup failed). The picture shows an error message. How can  
I solve it? I have tried to consult chatGPT, but it still didn't work  
out. Thanks!  
![image](https://github.com/Project-MONAI/MONAILabel/assets/82327798/a2debf4d-b20d-4fe8-8613-9f2a3c84268f)  
![image](https://github.com/Project-MONAI/MONAILabel/assets/82327798/c1c381df-18a4-4323-87c9-f34b4cf860f4)  
![image](https://github.com/Project-MONAI/MONAILabel/assets/82327798/d279ee28-e6f9-4cdc-87e8-0bd632e73169)

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Need a way to provide patient/ volume level annotations

**Issue No. 1446 opened by AHarouni on 2023-05-23 at 02:15:43 Type: None**

\*\*Describe the solution you\'d like\*\* In addition to segmentation we can  
do using slicer, I would like to provide volume level annotations as  
sex, disease severity, tumor location, disease location, etc. This would  
be the result of a classification models. it could also be helpful while  
selecting the data to train, as we can add annotations as difficulty,  
  
\*\*Describe alternatives you\'ve considered\*\* I have tried to reuse the  
drop down menu of scoring in slicer since it is not used ( no way to  
trigger scoring from slicer). This needs code changes on the slicer code  
as: 1- After pressing submit label to do another call to update label  
info \--\> calling the datastore method 2 - To retrieve values when we  
do inference after pressing \"next sample\"

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Need a scoring method to compare inference of 2 models

**Issue No. 1445 opened by AHarouni on 2023-05-23 at 02:06:45 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* This  
is related/ similar to  
<https://github.com/Project-MONAI/MONAILabel/issues/1444>  
  
\*\*Describe the solution you\'d like\*\* After training a model\_v1 then a  
new model\_v2. User would want to compare where the models agreed and  
where it was improved.  
  
\*\*Describe alternatives you\'ve considered\*\*  
  
\*\*Additional context\*\* We need to consider multiple issues here as we do  
xor between the 2 inference and store a new inference with a new tag.  
  
- For multiple labels we might need to loop over each labels  
- need to count similar and difference pixel

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Need new scoring method to count inferred labels

**Issue No. 1444 opened by AHarouni on 2023-05-23 at 01:57:14 Type: None**

\*\*Describe the solution you\'d like\*\* After running batch inference on  
your data (labeled and/or unlabeled ), It would be useful to find  
volumes where the model missed an organ / disease. Or over estimated /  
under estimated an organ or disease. For example:  
  
- Data set is for liver tumor, where are the images without tumors?  
 This is either the AI missed the tumor and it must be labeled or it  
 is a true normal and the data scientiest want to exclude this normal  
 images  
- Where are the images with 3 kidneys ? or kidney pixel count \> xx  
 pixels  
  
This request may need to count pixels as well as calculate the actual  
volume in mm\^3  
  
\*\*Describe alternatives you\'ve considered\*\* I managed to adapt the the  
code from the  
[sum](https://github.com/Project-MONAI/MONAILabel/blob/main/monailabel/tasks/scoring/sum.py)  
scoring as  
  
 import logging  
 import numpy as np  
 import torch  
 from monai.transforms import LoadImage  
 from monailabel.interfaces.datastore import Datastore, DefaultLabelTag  
 from monailabel.interfaces.tasks.scoring import ScoringMethod  
 from monailabel.interfaces.app import MONAILabelApp  
 from monailabel.interfaces.utils.app import app\_instance  
  
 logger = logging.getLogger(\_\_name\_\_)  
  
 class LabelCount(ScoringMethod):  
 """  
 Compute pixel count for each label  
 """  
 def \_\_init\_\_(self):  
 super().\_\_init\_\_("Compute label count for an inference ")  
  
 def info(self):  
 instance: MONAILabelApp = app\_instance()  
 dataStore: Datastore = instance.datastore()  
 status = dataStore.status()  
 label\_tag = list(status['label\_tags'].keys())  
  
 return {  
 "description": self.description,  
 "config":  
 {"label\_tag": label\_tag  
 }  
 }  
  
 def \_\_call\_\_(self, request, datastore: Datastore):  
 loader = LoadImage(image\_only=True)  
 tag = request.get("label\_tag", "")  
 if not tag:  
 logger.error(" scoring error! Need to pass a label\_tag")  
 return {}  
  
 result = {}  
 for image\_id in datastore.list\_images():  
 label\_id: str = datastore.get\_label\_by\_image\_id(image\_id, tag)  
 if label\_id:  
 uri = datastore.get\_label\_uri(label\_id, tag)  
 # logger.info(f" ============{label\_id=} ===={uri=}")  
 label = loader(uri)  
 if isinstance(label, torch.Tensor):  
 label = label.numpy()  
  
 lbs,lbs\_count=np.unique(label, return\_counts=True)  
 lbs\_count\_dict={}  
 for i, lb in enumerate(lbs):  
 lbs\_count\_dict[str(int(lb))]=int(lbs\_count[i]) # int conversion is needed to avoid json error  
 logger.info(f"============{label\_id=} ===={uri=} organs found for {image\_id} are {lbs\_count\_dict} ")  
  
 # datastore.update\_image\_info(image\_id, lbs\_count\_dict)  
 datastore.update\_label\_info(label\_id, tag, {"label\_count": lbs\_count\_dict})  
 result[label\_id] = {"label\_count": lbs\_count\_dict}  
 return result  
  
\*\*Additional context\*\* This needs an active learning to sort by the most  
or least number of pixels for each label.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Allow the use of cpu when having GPUs

**Issue No. 1442 opened by AHarouni on 2023-05-19 at 17:35:54 Type: None**

\*\*Describe the solution you\'d like\*\* In some cases with large models as  
total segmentator, or when we have large full body CT running inference  
can result in OOM errors using gpus. It would be nice to have the  
ability to pick cpu from device list  
  
we can change the lines of  
<https://github.com/Project-MONAI/MONAILabel/blob/main/monailabel/utils/others/generic.py#L192-L200>  
  
to  
  
 def device\_list():  
 if torch.cuda.is\_available()   
 devices = []   
 else:  
 return ["cpu"]  
 if torch.cuda.device\_count() == 1:  
 devices.append(torch.cuda.get\_device\_name(0))  
 else:  
 for i in range(torch.cuda.device\_count()):  
 devices.append(f"{torch.cuda.get\_device\_name(i)}:{i}")  
 devices.append("cpu") < ------- add cpu in the end   
 return devices  
  
\*\*Describe alternatives you\'ve considered\*\* For now I manaually do that  
in my \*\*init\*\* of my infer as  
  
 class myNetwork(BasicInferTask):  
 def \_\_init\_\_(self,path,network,  
 parameters,  
 type=InferType.SEGMENTATION,labels=None,dimension=3,  
 \*\*kwargs,  
 ):  
 super().\_\_init\_\_(path=path,network=network,type=type,  
 labels=labels,dimension=dimension,description=description,  
 \*\*kwargs,)  
  
 self.\_config["device"]= device\_list()+["cpu"] ## adding cpu to use

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

qMRMLSegmentEditorWidget has no attribute named 'setSourceVolumeNode'

**Issue No. 1440 opened by wyli on 2023-05-18 at 09:11:20 Type: None**

\*\*Describe the bug\*\* with slicer 5.0.2:  
  
 Traceback (most recent call last):  
 File "/Users/Documents/MONAILabel/plugins/slicer/MONAILabel/MONAILabel.py", line 1292, in onNextSampleButton  
 File "/Users/Documents/MONAILabel/plugins/slicer/MONAILabel/MONAILabel.py", line 1315, in initSample  
 self.ui.embeddedSegmentEditorWidget.setSegmentationNode(self.\_segmentNode)  
 AttributeError: qMRMLSegmentEditorWidget has no attribute named 'setSourceVolumeNode'  
  
It seems `setSourceVolumeNode` is introduced since slicer 5.2  
<https://github.com/Slicer/Slicer/commit/5dfb6fb6945d3fb56320c2920571f3ffcc5b48fb>  
  
Upgrading slicer to version 5.2.2 addressed the issue. Would be great to  
clarify the version requirements

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

default server address slicer plugin

**Issue No. 1438 opened by wyli on 2023-05-18 at 07:14:14 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* might  
be convenient to use a default value for the MONAI label server  
property:  
<https://github.com/Project-MONAI/MONAILabel/blob/b501ea0eebffe92e8c9c6daa043f0b43df85d50c/plugins/slicer/MONAILabel/Resources/UI/MONAILabel.ui#L72-L74>  
  
the value could be `http://0.0.0.0:8000` according to  
<https://github.com/Project-MONAI/MONAILabel/blob/b501ea0eebffe92e8c9c6daa043f0b43df85d50c/monailabel/main.py#L54-L55>

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Add option to run batch inference from slicer UI

**Issue No. 1437 opened by AHarouni on 2023-05-17 at 21:44:41 Type: None**

\*\*Describe the solution you\'d like\*\* From slicer UI user can trigger  
segmentation with multiple config parameters that he can set. Similarly  
we need an option to easily call batch inference with the same configs.  
Moreover, we should have a progress bar showing the progress of the  
batch infer job.  
  
\*\*Describe alternatives you\'ve considered\*\* I can use the monai label  
swagger UI to call batch infer but I then need to type in all the  
configs which is error prone.  
  
\*\*Additional context\*\* I managed to hack a solution in the slicer client  
by having a pop up to ask if I should do batch infer or run single  
image. Cleaner option is to add a new button along with may be more  
parameters related to batch inference as number of gpus to use. Code  
below works on my end  
  
 def onClickSegmentation(self): # AEH added batch inference support   
 .... initial code is as is   
 model = self.ui.segmentationModelSelector.currentText  
 image\_file = self.current\_sample["id"]  
 params = self.getParamsFromConfig("infer", model)  
   
 ############### adding batch inference   
 if not slicer.util.confirmOkCancelDisplay("Run batch single image ? (cancel will run batch inference"):  
 return self.logic.batch\_infer(model, params, session\_id=self.getSessionId())  
 ..... the rest of function is unchanged   
  
added function below in class  
MONAILabelLogic(ScriptedLoadableModuleLogic):  
  
 def batch\_infer(self, model, params={}, session\_id=None):  
 return self.\_client().batch\_infer(model, params,session\_id)  
  
in the client.py file I added the batch\_infer function call  
  
 def batch\_infer(self, model, params, session\_id=None):  
 """  
 Run Infer  
  
 :param model: Name of Model  
 :param params: Additional configs/json params as part of Infer request  
 :param session\_id: Session ID (use existing session id instead of image\_id)  
 """  
 selector = "/batch/infer/{}".format(  
 MONAILabelUtils.urllib\_quote\_plus(model),  
 )  
 if session\_id:  
 selector += f"&session\_id={MONAILabelUtils.urllib\_quote\_plus(session\_id)}"  
  
 params = self.\_update\_client\_id(params)  
 #fields = {"params": json.dumps(params) if params else "{}"}  
  
 status, response, \_, \_ = MONAILabelUtils.http\_method(  
 "POST", self.\_server\_url, selector, params, headers=self.\_headers  
 )  
 if status != 200:  
 raise MONAILabelClientException(  
 MONAILabelError.SERVER\_ERROR,  
 f"Status: {status}; Response: {bytes\_to\_str(response)}",  
 )  
  
 response = bytes\_to\_str(response)  
 logging.debug(f"Response: {response}")  
 return json.loads(response)

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Add ability to load different label to current image

**Issue No. 1436 opened by AHarouni on 2023-05-16 at 22:49:03 Type: None**

\*\*Describe the solution you\'d like\*\* clicking next image would load  
\'orignal\' label when that setting is enabled. If would be nice to have  
a user select the tag in the active learning strategy options, then  
slicer would only call server to get the label with that tag.  
  
![image](https://github.com/Project-MONAI/MONAILabel/assets/61221417/76409034-72bb-4f76-8662-a32f447ce969)  
  
\*\*Describe alternatives you\'ve considered\*\* I managed to get it to work  
after modifying slicer code. It may be a bit hacky as I assume a cmd  
name in the strategy to skip calling the main image  
  
 def onNextSampleButton(self):  
 if not self.logic:  
 return  
   
 curr\_image\_id = self.ui.inputSelector.currentText # aeh avoid unneeded reload   
 curr\_image\_id = curr\_image\_id[:curr\_image\_id.find(".nii")] if curr\_image\_id else ""  
 print(f"========{curr\_image\_id}")  
  
 loadNewImage=True  
 try: ## moved code up duplicate to check command and keep same image   
 # qt.QApplication.setOverrideCursor(qt.Qt.WaitCursor)  
 #self.updateServerSettings()  
 strategy = self.ui.strategyBox.currentText  
 if not strategy:  
 slicer.util.errorDisplay("No Strategy Found/Selected\t")  
 return  
 strategyParams=self.getParamsFromConfig("activelearning", strategy)  
 label\_tag2get = strategyParams.get('label\_tag',"original")  
 cmd=strategyParams.get("cmd", None)  
 if cmd=="LoadLabelTag" and len(curr\_image\_id)>0:  
 loadNewImage=False  
 except BaseException as e:  
 msg = f" aeh new code Message:: {e.msg}" if hasattr(e, "msg") else ""  
 slicer.util.errorDisplay(  
 f"Failed to fetch Sample from MONAI Label Server.\n{msg}",  
 detailedText=traceback.format\_exc(),  
 )  
 print(f"========{curr\_image\_id} {loadNewImage=}")  
   
 if loadNewImage and (self.\_volumeNode or len(slicer.util.getNodesByClass("vtkMRMLScalarVolumeNode")) ):  
 if not slicer.util.confirmOkCancelDisplay(  
 "This will close current scene. Please make sure you have saved your current work.\n"  
 "Are you sure to continue?"  
 ):  
 return  
 self.onResetScribbles()  
 slicer.mrmlScene.Clear(0)  
  
 start = time.time()  
 try:  
 qt.QApplication.setOverrideCursor(qt.Qt.WaitCursor)  
  
 self.updateServerSettings()  
 #strategy = self.ui.strategyBox.currentText # moved up   
 #if not strategy:  
 # slicer.util.errorDisplay("No Strategy Found/Selected\t")  
 # return  
   
 #strategyParams=self.getParamsFromConfig("activelearning", strategy) #moved up  
 #strategyParams["current\_loaded\_image\_id"] = curr\_image\_id ## will use this id when asking to compare models   
  
 if not loadNewImage: #aeh avoid un needed reload   
 image\_id = curr\_image\_id  
 print(f" keeping same sample {image\_id}")  
 else:   
  
 sample = self.logic.next\_sample(strategy, strategyParams )  
 logging.debug(sample)  
 if not sample.get("id"):  
 slicer.util.warningDisplay(  
 "Unlabled Samples/Images Not Found at server. Instead you can load your own image."  
 )  
 return  
  
 if loadNewImage and self.samples.get(sample["id"]) is not None:  
 self.current\_sample = self.samples[sample["id"]]  
 name = self.current\_sample["VolumeNodeName"]  
 index = self.ui.inputSelector.findText(name)  
 self.ui.inputSelector.setCurrentIndex(index)  
 return  
  
 logging.info(sample)  
 image\_id = sample["id"]  
 image\_file = sample.get("path")  
 image\_name = sample.get("name", image\_id)  
 node\_name = sample.get("PatientID", sample.get("name", image\_id))  
 checksum = sample.get("checksum")  
 local\_exists = image\_file and os.path.exists(image\_file)  
   
 print(f"------- aeh next sample id is {image\_id} labeltag= {label\_tag2get} ")  
 logging.info(f"Check if file exists/shared locally: {image\_file} => {local\_exists}")  
 if local\_exists:  
 self.\_volumeNode = slicer.util.loadVolume(image\_file)  
 self.\_volumeNode.SetName(node\_name)  
 else:  
 download\_uri = f"{self.serverUrl()}/datastore/image?image={quote\_plus(image\_id)}"  
 logging.info(download\_uri)  
  
 sampleDataLogic = SampleData.SampleDataLogic()  
 self.\_volumeNode = sampleDataLogic.downloadFromURL(  
 nodeNames=node\_name, fileNames=image\_name, uris=download\_uri, checksums=checksum  
 )[0]  
  
 if slicer.util.settingsValue("MONAILabel/originalLabel", True, converter=slicer.util.toBool):  
 try:  
 datastore = self.logic.datastore()  
 label\_info = datastore["objects"][image\_id]["labels"][label\_tag2get]["info"]  
 labels = label\_info.get("params", {}).get("label\_names", {})  
  
 if labels:  
 # labels are available in original label info  
 labels = labels.keys()  
 else:  
 # labels not available  
 # assume labels in app info are valid for original label file  
 labels = self.logic.info().get("labels")  
  
 # ext = datastore['objects'][image\_id]['labels']['original']['ext']  
 maskFile = self.logic.download\_label(image\_id, label\_tag2get)  
 self.updateSegmentationMask(maskFile, list(labels)) ## aeh copied above  
 print(f"label tag {label\_tag2get} label loaded! ")  
 except:  
 print(f"label tag {label\_tag2get} label not found /errors ... ")  
   
 if loadNewImage: ## aeh added condition  
 self.initSample(sample)  
  
 except BaseException as e:  
 msg = f"Message:: {e.msg}" if hasattr(e, "msg") else ""  
 slicer.util.errorDisplay(f"Failed to fetch Sample from MONAI Label Server.\n{msg}",detailedText=traceback.format\_exc(), )  
 finally:  
 qt.QApplication.restoreOverrideCursor()  
  
 self.updateGUIFromParameterNode()  
 logging.info(f"Time consumed by next\_sample: {time.time() - start:3.1f}")

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

OHIF Stack Image Sync

**Issue No. 1435 opened by ReubenDo on 2023-05-16 at 20:50:30 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* The  
current OHIF viewer doesn\'t allow for synchronizing views.  
Consequently, segmentation cannot be performed in a multi-modal setting.  
  
\*\*Describe the solution you\'d like\*\* A button in the OHIF viewer allows  
to synchronize views entitled \"Stack Image Sync\". It would be great to  
have this option.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Upcoming OpenSlide release will change library filename

**Issue No. 1433 opened by bgilbert on 2023-05-13 at 17:45:32 Type: None**

The next OpenSlide release, 4.0.0, will include some incompatible API  
changes. Those shouldn\'t affect you directly, but as a consequence,  
OpenSlide\'s [soname](https://en.wikipedia.org/wiki/Soname) will change.  
This will change the library filename to `libopenslide-1.dll` on  
Windows, `libopenslide.1.dylib` on macOS, and `libopenslide.so.1` on  
Linux. Since MONAI Label preloads the OpenSlide DLL by filename on  
Windows, you may want to proactively add support for the new name so  
MONAI Label will work with the new release.  
  
For more information, see  
<https://github.com/openslide/openslide/pull/413>. If you have any  
questions, please ask!

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Label Reviewer plugin Doesn't load labels

**Issue No. 1432 opened by AHarouni on 2023-05-12 at 18:19:53 Type: None**

\*\*Describe the bug\*\* I want to load images and go through the labels,  
either the final labels or the original labels infered by my model.  
loading up the reviewer app I can go through the images. However it  
doesn\'t seem to load the labels although the combobox (version of  
labels) displays all list correctly  
  
\*\*Server logs\*\* Nothing relevant there  
  
 [MainThread:INFO] (lib.datastore.local:502) - Adding Label: images\_s0157 => version\_102 => /tmp/tmp3bwqmfec.nrrd  
 [MainThread:INFO] (lib.datastore.local:506) - Adding Label: images\_s0157 => version\_102 => /tmp/tmp3bwqmfec.nrrd dest at /data/Challenges/totalsegmentator/monailabel/labels/version\_102/images\_s0157.nrrd  
 [MainThread:INFO] (lib.datastore.local:530) - Label Info: {'segmentationMeta': {'status': 'not segmented', 'approvedBy': '', 'level': '', 'comment': '', 'editTime': 1683911717}, 'ts': 1683911717, 'checksum': 'SHA256:de4c7060f9fb6caff4707ae072ecda0eacd8e4044a57865ba9f3473196a44385', 'name': 'images\_s0157.nrrd'}  
 [MainThread:INFO] (monailabel.interfaces.app:496) - New label saved for: images\_s0157 => images\_s0157  
  
\*\*To Reproduce\*\* Steps to reproduce the behavior:  
  
1. open monai label reviewer after starting monai label  
2. connect to the server  
3. after image loads change the selected label  
  
\*\*Expected behavior\*\* I expect to see the label after the images are  
loaded, it should also reload on changing the name from the drop down  
list of labels  
  
\*\*Screenshots\*\*  
![image](https://github.com/Project-MONAI/MONAILabel/assets/61221417/2c7b1fcd-f396-42bb-951e-7678b63df401)  
  
\*\*Additional context\*\* I am running:  
  
- latest slicer 5.3.0-4-27  
- monai label 0.6

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Adding near-real time collaborative manual segmentation

**Issue No. 1422 opened by kchawla-pi on 2023-05-01 at 16:48:55 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* I am  
looking into setting up a way to collaboratively annotate and segment  
sMRI. The idea is that we can invite specific people via email and they  
can all join and collaborate a volume as a team via a low latency  
connection, everything running on our local servers so the hospital's  
data does not leave the servers.  
  
Is such a feature possible with MONAI as it currently stands? If not,  
can we build one for it? If yes, what would that entail? Where should I  
ask these questions and get mentoring and stewardship?  
  
Is it more feasible to simply build a Slicer plugin and MONAI should not  
be involved with this?  
  
\*\*Describe the solution you\'d like\*\*  
  
1. Ultimately, a near-real time solution which works like a shared  
 whiteboard, where multiple people can manually segment a dataset  
 collaboratively, accessed via Slicer.  
2. For an easier initial product, it could be collaborative viewing  
 plugin, with only one editor.  
  
\*\*Describe alternatives you\'ve considered\*\*  
  
1. Simply use Dropbox like apps to sync the data files and save  
 frequently as editing is performed, so the modified data files are  
 synced. One leader, others viewer only. Latency may be an issue  
 here.  
2. Use <http://brainbox.pasteur.fr>  
  
\*\*Potential Ideas for Implementation\*\*  
  
1. I reached out to Stephen Aylward at Kitware & he suggested \"Clara  
 Train\'s AI assisted annotation tool supports multiple clients  
 annotating a common dataset to train a single AI model:  
 \[<https://docs.nvidia.com/clara/clara-train-sdk/aiaa/index.html>\]\".  
 Then he found out & informed me that it is being deprecated so not a  
 viable solution anymore.  
2. Another solution he proposed, in-lieu of above: \"Implement this as  
 a feature in MONAI Labels using MONAI Label\'s python API and  
 extending it to support multiple clients annotating the same data  
 using a shared model.\"  
3. Open a secure WebSocket connection among participants. Saving a file  
 triggers a file sync. One leader other viewers only, to avoid  
 conflict resolution necessity.  
4. Use a tool like <https://convergence.io/> to be the backend and  
 build a Slicer plugin around it.  
  
\*\*Additional context\*\* Add any other context or screenshots about the  
feature request here.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Expose a simple way to add training handlers

**Issue No. 1420 opened by AHarouni on 2023-04-28 at 02:54:49 Type: None**

\*\*Describe the solution you\'d like\*\* Need a simple way to add a handler  
to the training.  
  
\*\*Describe alternatives you\'ve considered\*\* Not sure if this is the  
best way but I managed to override the finalize method as below where I  
basically called the super then added on to add my handler  
  
 def finalize(self, context):  
 super().finalize(context)  
  
 if context.trainer:  
 from ignite.engine import Events  
 context.trainer.add\_event\_handler(event\_name=Events.EPOCH\_COMPLETED, handler=self.\_myadditional\_handler)  
  
\*\*Additional context\*\* It would be nice to have a method we can override  
that we can simply add the handler. Or even better and cleaner to have  
the method called to add to the list before we call the trainer as shown  
in  
<https://github.com/Project-MONAI/MONAI/blob/d23221f06afdf5fa7d8e3fde6552b6cceaf4a0bc/tests/test\_integration\_workflows.py#L162-L190>

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Possible enhancements of `monailabel/utils/others/generic.py`

**Issue No. 1408 opened by yiheng-wang-nv on 2023-04-19 at 06:53:41 Type: cement**

I viewed the code of this file, and found some places that may have  
issues or can be enhanced.  
  
1. <https://github.com/Project-MONAI/MONAILabel/blob/f0d3c6b8258301aaff18606b2c5a5a376f4714d4/monailabel/utils/others/generic.py#L349:L356>  
 I think here can define a function for this part, and then [another  
 place](https://github.com/Project-MONAI/MONAILabel/blob/f0d3c6b8258301aaff18606b2c5a5a376f4714d4/monailabel/utils/others/generic.py#L308:L318)  
 can also call this function.  
2. It seems now we cannot support to download ngc bundles with  
 specified version number (see:  
 <https://github.com/Project-MONAI/MONAILabel/blob/f0d3c6b8258301aaff18606b2c5a5a376f4714d4/monailabel/utils/others/generic.py#L356>),  
 should we also support it?

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

DeepEdit click based refinement seems to be adding segmentation on wrong side

**Issue No. 1407 opened by VishalJ99 on 2023-04-17 at 10:10:23 Type: None**

Ive taken the default radiology app and used it to train a deepEdit  
model to do volumetric MRI pancreas segmentation.  
  
The auto segmentation works well however I am experiencing a strange  
issue when I try and make click based refinements.  
  
Here is an example. This is the auto segmentation generated by the  
model:  
![image](https://user-images.githubusercontent.com/51826812/232453250-e7500fa7-e8f4-4e4e-aae0-168d7cb98590.png)  
You can see the right side could use further filling in so i add a  
foreground click in the desired region and the segmentation is updated  
to give the following:  
![image](https://user-images.githubusercontent.com/51826812/232453678-3af3108d-c1a4-423a-a0b6-2c1348d518fd.png)  
  
The refinement seems to be based on a click that was defined on the left  
side when I placed it on the right\...  
  
Here is the output when I purposefully define the click on the wrong  
side:  
  
![image](https://user-images.githubusercontent.com/51826812/232454078-0f2ae44b-549d-4e76-8907-21b789ba7bc6.png)  
  
i get the desired output.  
  
I do not experience this issue when making click based refinements on a  
model trained on the CT spleen dataset downloaded from monai labels  
example datasets. I have not made any changes to any files other than  
renaming the labels.  
  
The only explanation i can think of is that my input data is of  
orientation LAS whereas the CT spleen data is of orientation RAS.  
However, i noticed that in the training and infer pre transforms, one of  
the transforms defined is the orientationD transform which re orientates  
the volumes to RAS, so this should not cause any issue\...

**Additional Information:**

* Milestones: None
* Assignees: diazandr3s

No comments at the moment!

Export torch script when training, Allow to import it for inference

**Issue No. 1402 opened by AHarouni on 2023-04-11 at 19:15:52 Type: None**

\*\*Describe the solution you\'d like\*\* As user trains models, we are  
changing a lot of hyperparameters which includes number or layer in the  
network architecture and even change the model architecture completely.  
This requires user to track the model weight and architecture pairs in  
the inference.  
  
\*\*Describe alternatives you\'ve considered\*\* We can export ts next to  
checkpoint models which holds the model weights  
  
\*\*Additional context\*\*  
  
- Exporting ts can affect performance so it should be a done at the  
 end of the training.  
- Need to accommodate for user stopping the training

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Monai Label holds on to GPU memory when facing OOM error

**Issue No. 1397 opened by AHarouni on 2023-04-10 at 22:32:46 Type: None**

\*\*Describe the bug\*\* When running inference of totalSegmentator with  
large volumes, I do get OOM error. I was expecting I can change the  
sw\_device to use CPU and rerun since I added a config var to the  
inference. However Monai label still holds on to the gpu memory so No  
one can rerun or use that memory unless the server is stopped and  
restarted  
  
\*\*Server logs\*\*  
  
 2023-04-10 22:11:29,405] [16464] [INFER\_0] [INFO] (monailabel.tasks.infer.basic\_infer:412) - Infer model path: /MONAILabel/sample-apps/radiology/model/segmentation.pt  
 100%|██████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████| 224/224 [00:18<00:00, 12.09it/s]  
 [2023-04-10 22:11:48,654] [16464] [MainThread] [ERROR] (uvicorn.error:369) - Exception in ASGI application  
 Traceback (most recent call last):  
 File "/opt/conda/lib/python3.8/site-packages/uvicorn/protocols/http/h11\_impl.py", line 366, in run\_asgi  
 result = await app(self.scope, self.receive, self.send)  
 File "/opt/conda/lib/python3.8/site-packages/uvicorn/middleware/proxy\_headers.py", line 75, in \_\_call\_\_  
 return await self.app(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/fastapi/applications.py", line 269, in \_\_call\_\_  
 await super().\_\_call\_\_(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/applications.py", line 124, in \_\_call\_\_  
 await self.middleware\_stack(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/middleware/errors.py", line 184, in \_\_call\_\_  
 raise exc  
 File "/opt/conda/lib/python3.8/site-packages/starlette/middleware/errors.py", line 162, in \_\_call\_\_  
 await self.app(scope, receive, \_send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/middleware/cors.py", line 84, in \_\_call\_\_  
 await self.app(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/exceptions.py", line 93, in \_\_call\_\_  
 raise exc  
 File "/opt/conda/lib/python3.8/site-packages/starlette/exceptions.py", line 82, in \_\_call\_\_  
 await self.app(scope, receive, sender)  
 File "/opt/conda/lib/python3.8/site-packages/fastapi/middleware/asyncexitstack.py", line 21, in \_\_call\_\_  
 raise e  
 File "/opt/conda/lib/python3.8/site-packages/fastapi/middleware/asyncexitstack.py", line 18, in \_\_call\_\_  
 await self.app(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/routing.py", line 670, in \_\_call\_\_  
 await route.handle(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/routing.py", line 266, in handle  
 await self.app(scope, receive, send)  
 File "/opt/conda/lib/python3.8/site-packages/starlette/routing.py", line 65, in app  
 response = await func(request)  
 File "/opt/conda/lib/python3.8/site-packages/fastapi/routing.py", line 227, in app  
 raw\_response = await run\_endpoint\_function(  
 File "/opt/conda/lib/python3.8/site-packages/fastapi/routing.py", line 160, in run\_endpoint\_function  
 return await dependant.call(\*\*values)  
 File "/MONAILabel/monailabel/endpoints/infer.py", line 179, in api\_run\_inference  
 return run\_inference(background\_tasks, model, image, session\_id, params, file, label, output)  
 File "/MONAILabel/monailabel/endpoints/infer.py", line 161, in run\_inference  
 result = instance.infer(request)  
 File "/MONAILabel/monailabel/interfaces/app.py", line 298, in infer  
 result\_file\_name, result\_json = f.result(request.get("timeout", settings.MONAI\_LABEL\_INFER\_TIMEOUT))  
 File "/opt/conda/lib/python3.8/concurrent/futures/\_base.py", line 444, in result  
 return self.\_\_get\_result()  
 File "/opt/conda/lib/python3.8/concurrent/futures/\_base.py", line 389, in \_\_get\_result  
 raise self.\_exception  
 File "/opt/conda/lib/python3.8/concurrent/futures/thread.py", line 57, in run  
 result = self.fn(\*self.args, \*\*self.kwargs)  
 File "/MONAILabel/monailabel/interfaces/app.py", line 295, in run\_infer\_in\_thread  
 return t(r)  
 File "/MONAILabel/monailabel/tasks/infer/basic\_infer.py", line 299, in \_\_call\_\_  
 data = self.run\_inferer(data, device=device)  
 File "/MONAILabel/monailabel/tasks/infer/basic\_infer.py", line 473, in run\_inferer  
 outputs = inferer(inputs, network)  
 File "/opt/monai/monai/inferers/inferer.py", line 202, in \_\_call\_\_  
 return sliding\_window\_inference(  
 File "/opt/monai/monai/inferers/utils.py", line 252, in sliding\_window\_inference  
 output\_image\_list[ss] = (output\_image\_list[ss] / count\_map\_list.pop(0)).to(compute\_dtype)  
 torch.cuda.OutOfMemoryError: CUDA out of memory. Tried to allocate 21.84 GiB (GPU 0; 31.75 GiB total capacity; 23.38 GiB already allocated; 6.73 GiB free; 23.39 GiB reserved in total by PyTorch) If reserved memory is >> allocated memory try setting max\_split\_size\_mb to avoid fragmentation. See documentation for Memory Management and PYTORCH\_CUDA\_ALLOC\_CONF  
  
\*\*Expected behavior\*\* There should be a try Except block or something to  
release the gpu memory after a failed inference. In real deployment  
systems there would be cases where inference would fail as OOM or bad  
data. System should always free up memory unless we want to keep the  
model hot and always loaded

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Simplify adding static parameters and metric to MLFlow

**Issue No. 1383 opened by AHarouni on 2023-04-08 at 01:44:45 Type: None**

MLflow feature tracking is great feature that allows user to run  
multiple experiments and see why.  
  
To do so we must be able to log all metric and parameters used in the  
experiment. It is not clear/simple to add any static variable to the  
experiment. For example I have added a lot of parameters which I can  
control from slicer UI. However, I don\'t see an easy way to add them to  
mlflow.  
  
Also I recently added a running val metric `val\_los` how can I add that  
to the tracked variables  
  
\*\*Describe the solution you\'d like\*\* It would be great if : 1- is there  
is a simple method to override and add new static global vars as form  
the config. 2- there is a simple method to override to add new running  
metrices 2- we can add all existing parameters as number of gpus used,  
datacaching used, loaders, time it took to run, etc

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Use BundleWorkflow for bundle related process | monaibundle app

**Issue No. 1352 opened by tangy5 on 2023-03-22 at 15:28:11 Type: None**

monaibundle app to support `BundleWorkflow` for initialize, run and  
finalize with variants of config file patterns. This will bring more  
flexibility, and reduce bundle maintenance effort for MONAI Label.  
  
For the 0.8.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

torch.cuda.OutOfMemoryError & RuntimeError: applying transform

**Issue No. 1351 opened by keyurradia on 2023-03-22 at 11:29:34 Type: None**

\*\*Describe the bug\*\* Before I got to update my cudatoolkit 11.8  
MonaiLabel was not recognizing cudatoolkit 11.3 and cuda was disabling.  
The training process was ran in CPU mode and was running fine. After I  
got to update cudatoolkit 11.8 and then cuda is not getting disabled but  
I am getting - RuntimeError: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>.\  
When I go through then I found the \\*\\*\\_\"torch.cuda.OutOfMemoryError:  
CUDA out of memory. Tried to allocate 1.69 GiB (GPU 0; 8.00 GiB total  
capacity; 6.14 GiB already allocated; 0 bytes free; 6.22 GiB reserved in  
total by PyTorch) If reserved memory is \>\> allocated memory try  
setting max\_split\_size\_mb to avoid fragmentation. See documentation for  
Memory Management and PYTORCH\_CUDA\_ALLOC\_CONF  
  
\*\*Server logs\*\* \[2023-03-22 11:57:12,660\] \[7720\] \[MainThread\]  
\[INFO\] (monailabel.endpoints.datastore:68) - Image: 23.01.03.18; File:  
\<starlette.datastructures.UploadFile object at 0x000001C073531D90\>;  
params: {\"client\_id\": \"user-xyz\"} \[2023-03-22 11:57:12,746\]  
\[7720\] \[MainThread\] \[INFO\] (monailabel.datastore.local:439) -  
Adding Image: 23.01.03.18 =\>  
C:\\Users\\keyur\\AppData\\Local\\Temp\\tmp1n82rfbc.nii.gz \[2023-03-22  
11:57:13,325\] \[7720\] \[MainThread\] \[INFO\]  
(monailabel.endpoints.datastore:101) - Saving Label for 23.01.03.18 for  
tag: final by admin \[2023-03-22 11:57:13,331\] \[7720\] \[MainThread\]  
\[INFO\] (monailabel.endpoints.datastore:112) - Save Label params:  
{\"label\_info\": \[{\"name\": \"liver\", \"idx\": 1}, {\"name\":  
\"venaporta\", \"idx\": 2}, {\"name\": \"livervein\", \"idx\": 3},  
{\"name\": \"venacava\", \"idx\": 4}, {\"name\": \"lesions\", \"idx\":  
5}\], \"client\_id\": \"user-xyz\"} \[2023-03-22 11:57:13,332\] \[7720\]  
\[MainThread\] \[INFO\] (monailabel.datastore.local:486) - Saving Label  
for Image: 23.01.03.18; Tag: final; Info: {\'label\_info\': \[{\'name\':  
\'liver\', \'idx\': 1}, {\'name\': \'venaporta\', \'idx\': 2},  
{\'name\': \'livervein\', \'idx\': 3}, {\'name\': \'venacava\', \'idx\':  
4}, {\'name\': \'lesions\', \'idx\': 5}\], \'client\_id\': \'user-xyz\'}  
\[2023-03-22 11:57:13,333\] \[7720\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:494) - Adding Label: 23.01.03.18 =\> final  
=\> C:\\Users\\keyur\\AppData\\Local\\Temp\\tmpm17x2cn6.nii.gz  
\[2023-03-22 11:57:13,338\] \[7720\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:510) - Label Info: {\'label\_info\':  
\[{\'name\': \'liver\', \'idx\': 1}, {\'name\': \'venaporta\', \'idx\':  
2}, {\'name\': \'livervein\', \'idx\': 3}, {\'name\': \'venacava\',  
\'idx\': 4}, {\'name\': \'lesions\', \'idx\': 5}\], \'client\_id\':  
\'user-xyz\', \'ts\': 1679482633, \'name\': \'23.01.03.18.nii.gz\'}  
\[2023-03-22 11:57:13,344\] \[7720\] \[MainThread\] \[INFO\]  
(monailabel.interfaces.app:492) - New label saved for: 23.01.03.18 =\>  
23.01.03.18 \[2023-03-22 11:57:16,062\] \[7720\] \[MainThread\] \[INFO\]  
(monailabel.utils.async\_tasks.task:41) - Train request: {\'model\':  
\'segmentation\', \'name\': \'train\_01\', \'pretrained\': True,  
\'device\': \'cuda\', \'max\_epochs\': 50, \'early\_stop\_patience\': -1,  
\'val\_split\': 0.2, \'train\_batch\_size\': 1, \'val\_batch\_size\': 1,  
\'multi\_gpu\': True, \'gpus\': \'all\', \'dataset\':  
\'SmartCacheDataset\', \'dataloader\': \'ThreadDataLoader\',  
\'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'client\_id\': \'user-xyz\'}  
\[2023-03-22 11:57:16,063\] \[7720\] \[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:49) - Before::  
C:\\Users\\keyur\\MONAILabel; \[2023-03-22 11:57:16,064\] \[7720\]  
\[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:53) - After::  
C:\\Users\\keyur\\MONAILabel; \[2023-03-22 11:57:16,065\] \[7720\]  
\[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:65) - COMMAND::  
C:\\Users\\keyur.conda\\envs\\monai\\python.exe -m  
monailabel.interfaces.utils.app -m train -r  
{\"model\":\"segmentation\",\"name\":\"train\_01\",\"pretrained\":true,\"device\":\"cuda\",\"max\_epochs\":50,\"early\_stop\_patience\":-1,\"val\_split\":0.2,\"train\_batch\_size\":1,\"val\_batch\_size\":1,\"multi\_gpu\":true,\"gpus\":\"all\",\"dataset\":\"SmartCacheDataset\",\"dataloader\":\"ThreadDataLoader\",\"tracking\":\"mlflow\",\"tracking\_uri\":\"\",\"tracking\_experiment\_name\":\"\",\"client\_id\":\"user-xyz\"}  
\[2023-03-22 11:57:17,250\] \[32928\] \[MainThread\] \[INFO\]  
(\*\*main\*\*:37) - Initializing App from:  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\apps\\radiology;  
studies:  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\datasets\\training;  
conf: {\'models\': \'segmentation\'} \[2023-03-22 11:57:22,938\]  
\[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for MONAILabelApp  
Found: \<class \'main.MyApp\'\> \[2023-03-22 11:57:22,947\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.utils.others.class\_utils:57) -  
Subclass for TaskConfig Found: \<class  
\'lib.configs.deepedit.DeepEdit\'\> \[2023-03-22 11:57:22,948\]  
\[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_2d.Deepgrow2D\'\> \[2023-03-22  
11:57:22,948\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_3d.Deepgrow3D\'\> \[2023-03-22  
11:57:22,949\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.localization\_spine.LocalizationSpine\'\>  
\[2023-03-22 11:57:22,949\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.localization\_vertebra.LocalizationVertebra\'\>  
\[2023-03-22 11:57:22,950\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation.Segmentation\'\> \[2023-03-22  
11:57:22,950\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation\_spleen.SegmentationSpleen\'\>  
\[2023-03-22 11:57:22,951\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.segmentation\_vertebra.SegmentationVertebra\'\>  
\[2023-03-22 11:57:22,951\] \[32928\] \[MainThread\] \[INFO\]  
(main:93) - +++ Adding Model: segmentation =\>  
lib.configs.segmentation.Segmentation \[2023-03-22 11:57:22,974\]  
\[32928\] \[MainThread\] \[INFO\] (main:96) - +++ Using Models:  
\[\'segmentation\'\] \[2023-03-22 11:57:22,974\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.interfaces.app:134) - Init Datastore  
for:  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\datasets\\training  
\[2023-03-22 11:57:22,975\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:130) - Auto Reload: False; Extensions:  
\[\'\*.nii.gz\', \'\*.nii\', \'\*.nrrd\', \'\*.jpg\', \'\*.png\', \'\*.tif\',  
\'\*.svs\', \'\*.xml\'\] \[2023-03-22 11:57:22,986\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.datastore.local:577) - Invalidate  
count: 0 \[2023-03-22 11:57:22,987\] \[32928\] \[MainThread\] \[INFO\]  
(main:126) - +++ Adding Inferer:: segmentation =\>  
\<lib.infers.segmentation.Segmentation object at 0x00000141821A56D0\>  
\[2023-03-22 11:57:22,987\] \[32928\] \[MainThread\] \[INFO\]  
(main:191) - {\'segmentation\': \<lib.infers.segmentation.Segmentation  
object at 0x00000141821A56D0\>, \'Histogram+GraphCut\':  
\<monailabel.scribbles.infer.HistogramBasedGraphCut object at  
0x000001418AA7F370\>, \'GMM+GraphCut\':  
\<monailabel.scribbles.infer.GMMBasedGraphCut object at  
0x000001418AA7F340\>} \[2023-03-22 11:57:22,987\] \[32928\]  
\[MainThread\] \[INFO\] (main:206) - +++ Adding Trainer:: segmentation  
=\> \<lib.trainers.segmentation.Segmentation object at  
0x000001418AA7F3A0\> \[2023-03-22 11:57:22,987\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.utils.sessions:51) - Session Path:  
C:\\Users\\keyur.cache\\monailabel\\sessions \[2023-03-22 11:57:22,987\]  
\[32928\] \[MainThread\] \[INFO\] (monailabel.utils.sessions:52) -  
Session Expiry (max): 3600 \[2023-03-22 11:57:22,987\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:432) - Train  
Request (input): {\'model\': \'segmentation\', \'name\': \'train\_01\',  
\'pretrained\': True, \'device\': \'cuda\', \'max\_epochs\': 50,  
\'early\_stop\_patience\': -1, \'val\_split\': 0.2, \'train\_batch\_size\':  
1, \'val\_batch\_size\': 1, \'multi\_gpu\': True, \'gpus\': \'all\',  
\'dataset\': \'SmartCacheDataset\', \'dataloader\':  
\'ThreadDataLoader\', \'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'client\_id\': \'user-xyz\',  
\'local\_rank\': 0} \[2023-03-22 11:57:22,987\] \[32928\] \[MainThread\]  
\[INFO\] (monailabel.tasks.train.basic\_train:442) -  
CUDA\_VISIBLE\_DEVICES: None \[2023-03-22 11:57:22,989\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:447) -  
Distributed/Multi GPU is limited \[2023-03-22 11:57:22,989\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:462) -  
Distributed Training = FALSE \[2023-03-22 11:57:22,989\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:489) - 0 -  
Train Request (final): {\'name\': \'train\_01\', \'pretrained\': True,  
\'device\': \'cuda\', \'max\_epochs\': 50, \'early\_stop\_patience\': -1,  
\'val\_split\': 0.2, \'train\_batch\_size\': 1, \'val\_batch\_size\': 1,  
\'multi\_gpu\': False, \'gpus\': \'all\', \'dataset\':  
\'SmartCacheDataset\', \'dataloader\': \'ThreadDataLoader\',  
\'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'model\': \'segmentation\',  
\'client\_id\': \'user-xyz\', \'local\_rank\': 0, \'run\_id\':  
\'20230322\_115722\'} \[2023-03-22 11:57:22,989\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:622) - 0 -  
Using Device: cuda; IDX: None \[2023-03-22 11:57:22,989\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:515) -  
Run/Output Path:  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\apps\\radiology\\model\\segmentation\\train\_01  
\[2023-03-22 11:57:22,989\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:531) - Tracking: mlflow \[2023-03-22  
11:57:22,989\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:532) - Tracking URI:  
file:///C:/Users/keyur/MONAILabel/monailabel/scripts/apps/radiology/model/segmentation/train\_01/mlruns;  
\[2023-03-22 11:57:22,989\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:533) - Tracking Experiment Name:  
segmentation; Run Name: run\_20230322\_115722 \[2023-03-22 11:57:22,989\]  
\[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:410) - Total Records for Training: 6  
\[2023-03-22 11:57:22,989\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:411) - Total Records for Validation:  
2 Loading dataset: 0%\| \| 0/2 \[00:00\<?, ?it/s\] Loading dataset:  
50%\|##### \| 1/2 \[00:11\<00:11, 11.37s/it\] Loading dataset:  
100%\|##########\| 2/2 \[00:21\<00:00, 10.42s/it\] Loading dataset:  
100%\|##########\| 2/2 \[00:21\<00:00, 10.57s/it\] cache\_num is greater  
or equal than dataset length, fall back to regular  
monai.data.CacheDataset. \[2023-03-22 11:57:44,226\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:328) - 0 -  
Records for Validation: 2 \[2023-03-22 11:57:44,237\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:318) - 0 -  
Adding Validation to run every \'1\' interval \[2023-03-22  
11:57:44,240\] \[32928\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:710) - 0 - Load Path  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\apps\\radiology\\model\\segmentation\\train\_01\\model.pt  
Loading dataset: 0%\| \| 0/6 \[00:00\<?, ?it/s\] Loading dataset:  
17%\|#6 \| 1/6 \[00:10\<00:52, 10.54s/it\] Loading dataset: 33%\|###3 \|  
2/6 \[00:15\<00:28, 7.00s/it\] Loading dataset: 50%\|##### \| 3/6  
\[00:26\<00:27, 9.16s/it\] Loading dataset: 67%\|######6 \| 4/6  
\[00:35\<00:17, 8.80s/it\] Loading dataset: 83%\|########3 \| 5/6  
\[00:47\<00:10, 10.14s/it\] Loading dataset: 100%\|##########\| 6/6  
\[01:02\<00:00, 11.67s/it\] Loading dataset: 100%\|##########\| 6/6  
\[01:02\<00:00, 10.37s/it\] \[2023-03-22 11:58:46,454\] \[32928\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:264) - 0 -  
Records for Training: 6 \[2023-03-22 11:58:46,458\] \[32928\]  
\[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:876) -  
Engine run resuming from iteration 0, epoch 0 until 50 epochs  
\[2023-03-22 11:58:46,617\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:138) - Restored all variables  
from  
C:\\Users\\keyur\\MONAILabel\\monailabel\\scripts\\apps\\radiology\\model\\segmentation\\train\_01\\model.pt  
\[2023-03-22 11:58:51,634\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:272) - Epoch: 1/50, Iter: 1/6  
\-- train\_loss: 0.9931 \[2023-03-22 11:58:52,005\] \[32928\]  
\[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:272) -  
Epoch: 1/50, Iter: 2/6 \-- train\_loss: 0.9202 \[2023-03-22  
11:58:52,382\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:272) - Epoch: 1/50, Iter: 3/6  
\-- train\_loss: 0.8346 \[2023-03-22 11:58:52,736\] \[32928\]  
\[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:272) -  
Epoch: 1/50, Iter: 4/6 \-- train\_loss: 0.8939 \[2023-03-22  
11:58:53,133\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:272) - Epoch: 1/50, Iter: 5/6  
\-- train\_loss: 0.9609 \[2023-03-22 11:58:53,435\] \[32928\]  
\[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:272) -  
Epoch: 1/50, Iter: 6/6 \-- train\_loss: 0.8174 \[2023-03-22  
11:58:53,442\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:257) - Got new best metric of  
train\_mean\_dice: 0.2552022635936737 \[2023-03-22 11:58:53,442\]  
\[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:201) - Epoch\[1\] Metrics \--  
train\_lesions\_mean\_dice: 0.0045 train\_liver\_mean\_dice: 0.6504  
train\_livervein\_mean\_dice: 0.3602 train\_mean\_dice: 0.2552  
train\_venacava\_mean\_dice: 0.0001 train\_venaporta\_mean\_dice: 0.2535  
\[2023-03-22 11:58:53,442\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedTrainer:212) - Key metric:  
train\_mean\_dice best value: 0.2552022635936737 at epoch: 1 \[2023-03-22  
11:58:53,448\] \[32928\] \[MainThread\] \[INFO\]  
(ignite.engine.engine.SupervisedEvaluator:876) - Engine run resuming  
from iteration 0, epoch 0 until 1 epochs  
  
### \[2023-03-22 11:58:57,944\] \[32928\] \[MainThread\] \[ERROR\] (ignite.engine.engine.SupervisedEvaluator:1086) - Current run is terminating due to exception: applying transform \<monai.transforms.compose.Compose object at 0x0000014188C0B160\> {#2023-03-22-115857944-32928-mainthread-error-igniteengineenginesupervisedevaluator1086---current-run-is-terminating-due-to-exception-applying-transform-monaitransformscomposecompose-object-at-0x0000014188c0b160}  
  
\[2023-03-22 11:58:57,945\] \[32928\] \[MainThread\] \[ERROR\]  
(ignite.engine.engine.SupervisedEvaluator:180) - Exception: applying  
transform \<monai.transforms.compose.Compose object at  
0x0000014188C0B160\> Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \\_apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\dictionary.py\",  
line 202, in \*\*call\*\* d\[key\] = self.converter(d\[key\], argmax,  
to\_onehot, threshold, rounding) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\array.py\",  
line 220, in \*\*call\*\* img\_t = one\_hot( File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\networks\\utils.py\",  
line 158, in one\_hot o = torch.zeros(size=sh, dtype=dtype,  
device=labels.device)  
  
### torch.cuda.OutOfMemoryError: CUDA out of memory. Tried to allocate 1.69 GiB (GPU 0; 8.00 GiB total capacity; 6.14 GiB already allocated; 0 bytes free; 6.22 GiB reserved in total by PyTorch) If reserved memory is \>\> allocated memory try setting max\_split\_size\_mb to avoid fragmentation. See documentation for Memory Management and PYTORCH\_CUDA\_ALLOC\_CONF {#torchcudaoutofmemoryerror-cuda-out-of-memory-tried-to-allocate-169-gib-gpu-0-800-gib-total-capacity-614-gib-already-allocated-0-bytes-free-622-gib-reserved-in-total-by-pytorch-if-reserved-memory-is--allocated-memory-try-setting-max\_split\_size\_mb-to-avoid-fragmentation-see-documentation-for-memory-management-and-pytorch\_cuda\_alloc\_conf}  
  
The above exception was the direct cause of the following exception:  
Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \*apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\compose.py\",  
line 174, in \*\*call\*\* input\* = apply\_transform(\*transform, input\*,  
self.map\_items, self.unpack\_items, self.log\_stats) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.post.dictionary.AsDiscreted object at  
0x0000014188BFE5B0\> The above exception was the direct cause of the  
following exception: Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1068, in \\_run\_once\_on\_dataset\_as\_gen self.state.output =  
self.\\_process\_function(self, self.state.batch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 308, in \\_iteration  
engine.fire\_event(IterationEvents.MODEL\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 449, in fire\_event return self.\\_fire\_event(event\_name) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 224, in \\_run\_postprocessing engine.state.batch\[i\],  
engine.state.output\[i\] = engine\_apply\_transform(b, o, posttrans) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\utils.py\",  
line 258, in engine\_apply\_transform transformed\_data =  
apply\_transform(transform, data) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>  
\[2023-03-22 11:58:58,024\] \[32928\] \[MainThread\] \[ERROR\]  
(ignite.engine.engine.SupervisedEvaluator:992) - Engine run is  
terminating due to exception: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>  
\[2023-03-22 11:58:58,024\] \[32928\] \[MainThread\] \[ERROR\]  
(ignite.engine.engine.SupervisedEvaluator:180) - Exception: applying  
transform \<monai.transforms.compose.Compose object at  
0x0000014188C0B160\> Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \\_apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\dictionary.py\",  
line 202, in \*\*call\*\* d\[key\] = self.converter(d\[key\], argmax,  
to\_onehot, threshold, rounding) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\array.py\",  
line 220, in \*\*call\*\* img\_t = one\_hot( File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\networks\\utils.py\",  
line 158, in one\_hot o = torch.zeros(size=sh, dtype=dtype,  
device=labels.device) torch.cuda.OutOfMemoryError: CUDA out of memory.  
Tried to allocate 1.69 GiB (GPU 0; 8.00 GiB total capacity; 6.14 GiB  
already allocated; 0 bytes free; 6.22 GiB reserved in total by PyTorch)  
If reserved memory is \>\> allocated memory try setting  
max\_split\_size\_mb to avoid fragmentation. See documentation for Memory  
Management and PYTORCH\_CUDA\_ALLOC\_CONF The above exception was the  
direct cause of the following exception: Traceback (most recent call  
last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \*apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\compose.py\",  
line 174, in \*\*call\*\* input\* = apply\_transform(\*transform, input\*,  
self.map\_items, self.unpack\_items, self.log\_stats) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.post.dictionary.AsDiscreted object at  
0x0000014188BFE5B0\> The above exception was the direct cause of the  
following exception: Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 959, in \\_internal\_run\_as\_gen epoch\_time\_taken += yield from  
self.\\_run\_once\_on\_dataset\_as\_gen() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1087, in \\_run\_once\_on\_dataset\_as\_gen self.\\_handle\_exception(e)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1068, in \\_run\_once\_on\_dataset\_as\_gen self.state.output =  
self.\\_process\_function(self, self.state.batch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 308, in \\_iteration  
engine.fire\_event(IterationEvents.MODEL\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 449, in fire\_event return self.\\_fire\_event(event\_name) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 224, in \\_run\_postprocessing engine.state.batch\[i\],  
engine.state.output\[i\] = engine\_apply\_transform(b, o, posttrans) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\utils.py\",  
line 258, in engine\_apply\_transform transformed\_data =  
apply\_transform(transform, data) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>  
\[2023-03-22 11:58:58,027\] \[32928\] \[MainThread\] \[ERROR\]  
(ignite.engine.engine.SupervisedTrainer:992) - Engine run is terminating  
due to exception: applying transform \<monai.transforms.compose.Compose  
object at 0x0000014188C0B160\> \[2023-03-22 11:58:58,027\] \[32928\]  
\[MainThread\] \[ERROR\] (ignite.engine.engine.SupervisedTrainer:180) -  
Exception: applying transform \<monai.transforms.compose.Compose object  
at 0x0000014188C0B160\> Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \\_apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\dictionary.py\",  
line 202, in \*\*call\*\* d\[key\] = self.converter(d\[key\], argmax,  
to\_onehot, threshold, rounding) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\array.py\",  
line 220, in \*\*call\*\* img\_t = one\_hot( File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\networks\\utils.py\",  
line 158, in one\_hot o = torch.zeros(size=sh, dtype=dtype,  
device=labels.device) torch.cuda.OutOfMemoryError: CUDA out of memory.  
Tried to allocate 1.69 GiB (GPU 0; 8.00 GiB total capacity; 6.14 GiB  
already allocated; 0 bytes free; 6.22 GiB reserved in total by PyTorch)  
If reserved memory is \>\> allocated memory try setting  
max\_split\_size\_mb to avoid fragmentation. See documentation for Memory  
Management and PYTORCH\_CUDA\_ALLOC\_CONF The above exception was the  
direct cause of the following exception: Traceback (most recent call  
last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \*apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\compose.py\",  
line 174, in \*\*call\*\* input\* = apply\_transform(\*transform, input\*,  
self.map\_items, self.unpack\_items, self.log\_stats) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.post.dictionary.AsDiscreted object at  
0x0000014188BFE5B0\> The above exception was the direct cause of the  
following exception: Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 965, in \\_internal\_run\_as\_gen  
self.\\_fire\_event(Events.EPOCH\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\validation\_handler.py\",  
line 76, in \*\*call\*\* self.validator.run(engine.state.epoch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 148, in run super().run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 281, in run super().run(data=self.data\_loader,  
max\_epochs=self.state.max\_epochs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 892, in run return self.\\_internal\_run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 935, in \\_internal\_run return next(self.\\_internal\_run\_generator)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 993, in \\_internal\_run\_as\_gen self.\\_handle\_exception(e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 959, in \\_internal\_run\_as\_gen epoch\_time\_taken += yield from  
self.\\_run\_once\_on\_dataset\_as\_gen() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1087, in \\_run\_once\_on\_dataset\_as\_gen self.\\_handle\_exception(e)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1068, in \\_run\_once\_on\_dataset\_as\_gen self.state.output =  
self.\\_process\_function(self, self.state.batch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 308, in \\_iteration  
engine.fire\_event(IterationEvents.MODEL\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 449, in fire\_event return self.\\_fire\_event(event\_name) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 224, in \\_run\_postprocessing engine.state.batch\[i\],  
engine.state.output\[i\] = engine\_apply\_transform(b, o, posttrans) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\utils.py\",  
line 258, in engine\_apply\_transform transformed\_data =  
apply\_transform(transform, data) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>  
Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \\_apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\dictionary.py\",  
line 202, in \*\*call\*\* d\[key\] = self.converter(d\[key\], argmax,  
to\_onehot, threshold, rounding) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\post\\array.py\",  
line 220, in \*\*call\*\* img\_t = one\_hot( File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\networks\\utils.py\",  
line 158, in one\_hot o = torch.zeros(size=sh, dtype=dtype,  
device=labels.device) torch.cuda.OutOfMemoryError: CUDA out of memory.  
Tried to allocate 1.69 GiB (GPU 0; 8.00 GiB total capacity; 6.14 GiB  
already allocated; 0 bytes free; 6.22 GiB reserved in total by PyTorch)  
If reserved memory is \>\> allocated memory try setting  
max\_split\_size\_mb to avoid fragmentation. See documentation for Memory  
Management and PYTORCH\_CUDA\_ALLOC\_CONF The above exception was the  
direct cause of the following exception: Traceback (most recent call  
last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 102, in apply\_transform return \\_apply\_transform(transform, data,  
unpack\_items) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 66, in \*apply\_transform return transform(parameters) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\compose.py\",  
line 174, in \*\*call\*\* input\* = apply\_transform(\*transform, input\*,  
self.map\_items, self.unpack\_items, self.log\_stats) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.post.dictionary.AsDiscreted object at  
0x0000014188BFE5B0\> The above exception was the direct cause of the  
following exception: Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\runpy.py\", line 197, in  
\\_run\_module\_as\_main return \\_run\_code(code, main\_globals, None, File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\runpy.py\", line 87, in  
\\_run\_code exec(code, run\_globals) File  
\"C:\\Users\\keyur\\MONAILabel\\monailabel\\interfaces\\utils\\app.py\",  
line 128, in `<module>`{=html} run\_main() File  
\"C:\\Users\\keyur\\MONAILabel\\monailabel\\interfaces\\utils\\app.py\",  
line 113, in run\_main result = a.train(request) File  
\"C:\\Users\\keyur\\MONAILabel\\monailabel\\interfaces\\app.py\", line  
422, in train result = task(request, self.datastore()) File  
\"C:\\Users\\keyur\\MONAILabel\\monailabel\\tasks\\train\\basic\_train.py\",  
line 463, in \*\*call\*\* res = self.train(0, world\_size, req, datalist)  
File  
\"C:\\Users\\keyur\\MONAILabel\\monailabel\\tasks\\train\\basic\_train.py\",  
line 552, in train context.trainer.run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\trainer.py\",  
line 53, in run super().run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 281, in run super().run(data=self.data\_loader,  
max\_epochs=self.state.max\_epochs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 892, in run return self.\\_internal\_run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 935, in \\_internal\_run return next(self.\\_internal\_run\_generator)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 993, in \\_internal\_run\_as\_gen self.\\_handle\_exception(e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 965, in \\_internal\_run\_as\_gen  
self.\\_fire\_event(Events.EPOCH\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\validation\_handler.py\",  
line 76, in \*\*call\*\* self.validator.run(engine.state.epoch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 148, in run super().run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 281, in run super().run(data=self.data\_loader,  
max\_epochs=self.state.max\_epochs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 892, in run return self.\\_internal\_run() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 935, in \\_internal\_run return next(self.\\_internal\_run\_generator)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 993, in \\_internal\_run\_as\_gen self.\\_handle\_exception(e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 959, in \\_internal\_run\_as\_gen epoch\_time\_taken += yield from  
self.\\_run\_once\_on\_dataset\_as\_gen() File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1087, in \\_run\_once\_on\_dataset\_as\_gen self.\\_handle\_exception(e)  
File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 1068, in \\_run\_once\_on\_dataset\_as\_gen self.state.output =  
self.\\_process\_function(self, self.state.batch) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\evaluator.py\",  
line 308, in \\_iteration  
engine.fire\_event(IterationEvents.MODEL\_COMPLETED) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 449, in fire\_event return self.\\_fire\_event(event\_name) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 224, in \\_run\_postprocessing engine.state.batch\[i\],  
engine.state.output\[i\] = engine\_apply\_transform(b, o, posttrans) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\engines\\utils.py\",  
line 258, in engine\_apply\_transform transformed\_data =  
apply\_transform(transform, data) File  
\"C:\\Users\\keyur.conda\\envs\\monai\\lib\\site-packages\\monai\\transforms\\transform.py\",  
line 129, in apply\_transform raise RuntimeError(f\"applying transform  
{transform}\") from e RuntimeError: applying transform  
\<monai.transforms.compose.Compose object at 0x0000014188C0B160\>  
\[2023-03-22 11:58:59,641\] \[7720\] \[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:83) - Return code: 1  
  
\*\*To Reproduce\*\* Steps to reproduce the behavior:  
  
1. Activate Monailabel server  
2. Start training from 3D Slicer  
3. Run commands \'\....\' (how you have started monailabel server) :  
 monailabel start\_server \--app apps/radiology \--studies  
 datasets/training \--conf models segmentation  
  
\*\*Expected behavior\*\* A clear and concise description of what you  
expected to happen.  
  
\*\*Screenshots\*\* If applicable, add screenshots to help explain your  
problem.  
  
\*\*Environment\*\*  
  
Ensuring you use the relevant python executable, please paste the output  
of:  
  
 python -c 'import monai; monai.config.print\_debug\_info()'  
  
 ================================  
 Printing MONAI config...  
 ================================  
 MONAI version: 1.1.0  
 Numpy version: 1.23.5  
 Pytorch version: 2.0.0+cu118  
 MONAI flags: HAS\_EXT = False, USE\_COMPILED = False, USE\_META\_DICT = False  
 MONAI rev id: a2ec3752f54bfc3b40e7952234fbeb5452ed63e3  
 MONAI \_\_file\_\_: C:\Users\keyur\.conda\envs\monai\lib\site-packages\monai\\_\_init\_\_.py  
  
 Optional dependencies:  
 Pytorch Ignite version: 0.4.10  
 Nibabel version: 5.0.1  
 scikit-image version: 0.20.0  
 Pillow version: 9.3.0  
 Tensorboard version: 2.12.0  
 gdown version: 4.6.4  
 TorchVision version: 0.15.1+cpu  
 tqdm version: 4.65.0  
 lmdb version: 1.4.0  
 psutil version: 5.9.0  
 pandas version: 1.5.3  
 einops version: 0.6.0  
 transformers version: NOT INSTALLED or UNKNOWN VERSION.  
 mlflow version: 2.2.2  
 pynrrd version: 0.4.3  
  
 For details about installing the optional dependencies, please visit:  
 https://docs.monai.io/en/latest/installation.html#installing-the-recommended-dependencies  
  
  
 ================================  
 Printing system config...  
 ================================  
 System: Windows  
 Win32 version: ('10', '10.0.22621', 'SP0', 'Multiprocessor Free')  
 Win32 edition: Core  
 Platform: Windows-10-10.0.22621-SP0  
 Processor: Intel64 Family 6 Model 186 Stepping 2, GenuineIntel  
 Machine: AMD64  
 Python version: 3.9.16  
 Process name: python.exe  
 Command: ['C:/Users/keyur/.conda/envs/monai\\python.exe', '-m', 'ipykernel\_launcher', '-f', 'C:\\Users\\keyur\\AppData\\Roaming\\jupyter\\runtime\\kernel-16ea1610-8b90-4286-bcdb-5d8bc9d19305.json']  
 Open files: [popenfile(path='C:\\Users\\keyur\\.ipython\\profile\_default\\history.sqlite', fd=-1), popenfile(path='C:\\Program Files\\WindowsApps\\Microsoft.LanguageExperiencePacken-GB\_22621.13.87.0\_neutral\_\_8wekyb3d8bbwe\\Windows\\System32\\en-GB\\2d99171d54bafb1068cad8303bddb437\\tzres.dll.mui', fd=-1), popenfile(path='C:\\Windows\\System32\\en-US\\kernel32.dll.mui', fd=-1), popenfile(path='C:\\Windows\\System32\\DriverStore\\FileRepository\\nvsmui.inf\_amd64\_1e558733305022a1\\nvcubins.bin', fd=-1), popenfile(path='C:\\Windows\\System32\\en-US\\KernelBase.dll.mui', fd=-1)]  
 Num physical CPUs: 14  
 Num logical CPUs: 20  
 Num usable CPUs: 20  
 CPU usage (%): [0.7, 0.4, 1.0, 0.1, 5.1, 1.1, 3.7, 1.2, 1.8, 0.4, 0.9, 0.4, 11.4, 14.7, 9.1, 10.1, 10.1, 10.6, 11.7, 12.5]  
 CPU freq. (MHz): 2600  
 Load avg. in last 1, 5, 15 mins (%): [13.8, 13.8, 10.0]  
 Disk usage (%): 44.1  
 Avg. sensor temp. (Celsius): UNKNOWN for given OS  
 Total physical memory (GB): 31.6  
 Available memory (GB): 18.1  
 Used memory (GB): 13.5  
  
 ================================  
 Printing GPU config...  
 ================================  
 Num GPUs: 1  
 Has CUDA: True  
 CUDA version: 11.8  
 cuDNN enabled: True  
 cuDNN version: 8700  
 Current device: 0  
 Library compiled for CUDA architectures: ['sm\_37', 'sm\_50', 'sm\_60', 'sm\_61', 'sm\_70', 'sm\_75', 'sm\_80', 'sm\_86', 'sm\_90', 'compute\_37']  
 GPU 0 Name: NVIDIA GeForce RTX 4070 Laptop GPU  
 GPU 0 Is integrated: False  
 GPU 0 Is multi GPU board: False  
 GPU 0 Multi processor count: 36  
 GPU 0 Total memory (GB): 8.0  
 GPU 0 CUDA capability (maj.min): 8.9  
  
 \*\*Additional context\*\*  
 Add any other context about the problem here.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Issues with heuristic\_planner on a multiple GPU setup

**Issue No. 1349 opened by saileshsidhwani on 2023-03-21 at 20:28:56 Type: None**

\*\*Describe the bug\*\* When using \'heuristic\_planner\' with the sample  
radiology app, starting the monailabel server errors out on my system  
with the following error:  
  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\utils\others\planner.py", line 97, in \_get\_target\_img\_size  
 width\_base\_2 = int(2 \*\* np.round(np.log2(width)))  
 ValueError: cannot convert float NaN to integer  
  
My machine has 2 different GPUs, the first one is a 2GB Quadro P600 and  
the second one is a 24GB NVIDIA TITAN RTX.  
  
Digging more into this I noticed that the `\_get\_target\_img\_size` method  
in `monailabel.utils.others.planner` class always uses the first GPU on  
the system to compute available GPU memory for heuristic purposes.  
  
 def \_get\_target\_img\_size(target\_img\_size):  
 # This should return an image according to the free gpu memory available  
 # Equation obtained from curve fitting using table:  
 # https://tinyurl.com/tableGPUMemory  
 gpu\_mem = gpu\_memory\_map()[0]  
  
This is not ideal as the first GPU on my system is only 2GB and the math  
in the above function leads to errors. For training/inference, I always  
set the cuda device to make sure the right GPU is used instead (RTX in  
my case).  
  
If in the above code, I replace what GPU to use, everything works as  
expected `gpu\_mem = gpu\_memory\_map()[0]` with  
`gpu\_mem = gpu\_memory\_map()[1]`  
  
This leads me to believe that It would be best if I can somehow tell the  
planner what GPU to use to compute the heuristic.  
  
\*\*Server logs\*\*  
  
 [2023-03-21 16:00:44,290] [23536] [MainThread] [INFO] (uvicorn.error:75) - Started server process [23536]  
 [2023-03-21 16:00:44,291] [23536] [MainThread] [INFO] (uvicorn.error:45) - Waiting for application startup.  
 [2023-03-21 16:00:44,291] [23536] [MainThread] [INFO] (monailabel.interfaces.utils.app:38) - Initializing App from: \\batfs-sb09-cifs\vmgr\sb09\ssidhwan\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\monailabel\sample-apps\radiology; studies: C:\Users\ssidhwan\Desktop\imagesTr; conf: {'models': 'segmentation\_spleen', 'heuristic\_planner': 'true'}  
 [2023-03-21 16:00:45,795] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for MONAILabelApp Found: <class 'main.MyApp'>  
 [2023-03-21 16:00:46,019] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.deepedit.DeepEdit'>  
 [2023-03-21 16:00:46,032] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.deepgrow\_2d.Deepgrow2D'>  
 [2023-03-21 16:00:46,064] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.deepgrow\_3d.Deepgrow3D'>  
 [2023-03-21 16:00:46,078] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.localization\_spine.LocalizationSpine'>  
 [2023-03-21 16:00:46,100] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.localization\_vertebra.LocalizationVertebra'>  
 [2023-03-21 16:00:46,112] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.segmentation.Segmentation'>  
 [2023-03-21 16:00:46,133] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.segmentation\_spleen.SegmentationSpleen'>  
 [2023-03-21 16:00:46,143] [23536] [MainThread] [INFO] (monailabel.utils.others.class\_utils:37) - Subclass for TaskConfig Found: <class 'lib.configs.segmentation\_vertebra.SegmentationVertebra'>  
 [2023-03-21 16:00:46,160] [23536] [MainThread] [INFO] (main:93) - +++ Adding Model: segmentation\_spleen => lib.configs.segmentation\_spleen.SegmentationSpleen  
 [2023-03-21 16:00:46,325] [23536] [MainThread] [INFO] (lib.configs.segmentation\_spleen:75) - EPISTEMIC Enabled: False; Samples: 5  
 [2023-03-21 16:00:46,330] [23536] [MainThread] [INFO] (main:96) - +++ Using Models: ['segmentation\_spleen']  
 [2023-03-21 16:00:46,335] [23536] [MainThread] [INFO] (monailabel.interfaces.app:135) - Init Datastore for: C:\Users\ssidhwan\Desktop\imagesTr  
 [2023-03-21 16:00:46,339] [23536] [MainThread] [INFO] (monailabel.datastore.local:129) - Auto Reload: True; Extensions: ['\*.nii.gz', '\*.nii', '\*.nrrd', '\*.jpg', '\*.png', '\*.tif', '\*.svs', '\*.xml']  
 [2023-03-21 16:00:46,385] [23536] [MainThread] [INFO] (monailabel.datastore.local:576) - Invalidate count: 0  
 [2023-03-21 16:00:46,399] [23536] [MainThread] [INFO] (monailabel.datastore.local:150) - Start observing external modifications on datastore (AUTO RELOAD)  
 [2023-03-21 16:00:46,406] [23536] [MainThread] [INFO] (monailabel.utils.others.planner:36) - Reading datastore metadata for heuristic planner...  
 100%|███████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████████| 10/10 [00:08<00:00, 1.12it/s]  
 [2023-03-21 16:00:55,324] [23536] [MainThread] [INFO] (monailabel.utils.others.generic:161) - Using nvidia-smi command  
 [2023-03-21 16:00:55,461] [23536] [MainThread] [INFO] (monailabel.utils.others.planner:72) - Available GPU memory: {0: 177, 1: 23853} in MB  
 [2023-03-21 16:00:55,466] [23536] [MainThread] [INFO] (monailabel.utils.others.generic:161) - Using nvidia-smi command  
 -28.484375  
 invalid value encountered in log2  
 [2023-03-21 16:00:55,696] [23536] [MainThread] [ERROR] (uvicorn.error:119) - Traceback (most recent call last):  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\starlette\routing.py", line 635, in lifespan  
 async with self.lifespan\_context(app):  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\starlette\routing.py", line 530, in \_\_aenter\_\_  
 await self.\_router.startup()  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\starlette\routing.py", line 612, in startup  
 await handler()  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\app.py", line 106, in startup\_event  
 instance = app\_instance()  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\interfaces\utils\app.py", line 51, in app\_instance  
 app = c(app\_dir=app\_dir, studies=studies, conf=conf)  
 File "\\batfs-sb09-cifs\vmgr\sb09\ssidhwan\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\monailabel\sample-apps\radiology\main.py", line 101, in \_\_init\_\_  
 super().\_\_init\_\_(  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\interfaces\app.py", line 99, in \_\_init\_\_  
 self.\_datastore: Datastore = self.init\_datastore()  
 File "\\batfs-sb09-cifs\vmgr\sb09\ssidhwan\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\monailabel\sample-apps\radiology\main.py", line 113, in init\_datastore  
 self.planner.run(datastore)  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\utils\others\planner.py", line 75, in run  
 self.spatial\_size = self.\_get\_target\_img\_size(np.mean(img\_sizes, 0, np.int64))  
 File "S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monailabel\utils\others\planner.py", line 97, in \_get\_target\_img\_size  
 width\_base\_2 = int(2 \*\* np.round(np.log2(width)))  
 ValueError: cannot convert float NaN to integer  
  
 [2023-03-21 16:00:55,697] [23536] [MainThread] [ERROR] (uvicorn.error:56) - Application startup failed. Exiting.  
  
\*\*To Reproduce\*\* Steps to reproduce the behavior:  
  
1. Go to On WIndows 10  
  
2. Install  
  
```{=html}  
<!-- -->  
```  
 $ python -m pip install --upgrade pip setuptools wheel  
 $ pip install torch==1.12.1+cu113 torchvision==0.13.1+cu113 torchaudio==0.12.1 --extra-index-url https://download.pytorch.org/whl/cu113  
 $ pip install monailabel  
  
3. Run commands  
 `monailabel start\_server --app .\monailabel\sample-apps\radiology\ --studies 'C:\Users\ssidhwan\Desktop\imagesTr\' --conf models segmentation\_spleen --conf heuristic\_planner true`  
  
\*\*Expected behavior\*\* A way to tell heuristic planner on what GPU(s) to  
use insted of always using the first GPU  
  
\*\*Screenshots\*\* If applicable, add screenshots to help explain your  
problem.  
  
\*\*Environment\*\*  
  
Ensuring you use the relevant python executable, please paste the output  
of:  
  
 python -c 'import monai; monai.config.print\_debug\_info()'  
 ================================  
 Printing MONAI config...  
 ================================  
 MONAI version: 1.1.0  
 Numpy version: 1.23.5  
 Pytorch version: 1.12.1+cu113  
 MONAI flags: HAS\_EXT = False, USE\_COMPILED = False, USE\_META\_DICT = False  
 MONAI rev id: a2ec3752f54bfc3b40e7952234fbeb5452ed63e3  
 MONAI \_\_file\_\_: S:\VirtualEnvs\Windows\Cuda11\_3\_Torch1\_12\lib\site-packages\monai\\_\_init\_\_.py  
  
 Optional dependencies:  
 Pytorch Ignite version: 0.4.10  
 Nibabel version: 5.0.1  
 scikit-image version: 0.20.0  
 Pillow version: 9.4.0  
 Tensorboard version: 2.12.0  
 gdown version: 4.6.4  
 TorchVision version: 0.13.1+cu113  
 tqdm version: 4.65.0  
 lmdb version: 1.4.0  
 psutil version: 5.9.4  
 pandas version: 1.5.3  
 einops version: 0.6.0  
 transformers version: NOT INSTALLED or UNKNOWN VERSION.  
 mlflow version: 2.2.1  
 pynrrd version: 0.4.3  
  
 For details about installing the optional dependencies, please visit:  
 https://docs.monai.io/en/latest/installation.html#installing-the-recommended-dependencies  
  
  
 ================================  
 Printing system config...  
 ================================  
 System: Windows  
 Win32 version: ('10', '10.0.19045', 'SP0', 'Multiprocessor Free')  
 Win32 edition: Enterprise  
 Platform: Windows-10-10.0.19045-SP0  
 Processor: Intel64 Family 6 Model 85 Stepping 4, GenuineIntel  
 Machine: AMD64  
 Python version: 3.9.13  
 Process name: python.exe  
 Command: ['C:\\Users\\ssidhwan\\AppData\\Local\\Programs\\Python\\Python39\\python.exe', '-c', 'import monai; monai.config.print\_debug\_info()']  
 Open files: [popenfile(path='C:\\Windows\\System32\\en-US\\kernel32.dll.mui', fd=-1), popenfile(path='C:\\Windows\\System32\\en-US\\tzres.dll.mui', fd=-1), popenfile(path='C:\\Windows\\System32\\en-US\\KernelBase.dll.mui', fd=-1)]  
 Num physical CPUs: 6  
 Num logical CPUs: 12  
 Num usable CPUs: 12  
 CPU usage (%): [47.1, 8.5, 23.4, 10.2, 24.1, 16.2, 21.3, 26.9, 19.8, 10.4, 22.5, 28.5]  
 CPU freq. (MHz): 3600  
 Load avg. in last 1, 5, 15 mins (%): [0.0, 0.0, 0.0]  
 Disk usage (%): 30.9  
 Avg. sensor temp. (Celsius): UNKNOWN for given OS  
 Total physical memory (GB): 63.7  
 Available memory (GB): 34.0  
 Used memory (GB): 29.6  
  
 ================================  
 Printing GPU config...  
 ================================  
 Num GPUs: 2  
 Has CUDA: True  
 CUDA version: 11.3  
 cuDNN enabled: True  
 cuDNN version: 8302  
 Current device: 0  
 Library compiled for CUDA architectures: ['sm\_37', 'sm\_50', 'sm\_60', 'sm\_61', 'sm\_70', 'sm\_75', 'sm\_80', 'sm\_86', 'compute\_37']  
 GPU 0 Name: NVIDIA TITAN RTX  
 GPU 0 Is integrated: False  
 GPU 0 Is multi GPU board: False  
 GPU 0 Multi processor count: 72  
 GPU 0 Total memory (GB): 24.0  
 GPU 0 CUDA capability (maj.min): 7.5  
 GPU 1 Name: Quadro P600  
 GPU 1 Is integrated: False  
 GPU 1 Is multi GPU board: False  
 GPU 1 Multi processor count: 3  
 GPU 1 Total memory (GB): 2.0  
 GPU 1 CUDA capability (maj.min): 6.1  
  
\*\*Additional context\*\* Add any other context about the problem here.  
  
 $ nvidia-smi   
 +-----------------------------------------------------------------------------+   
 | NVIDIA-SMI 516.94 Driver Version: 516.94 CUDA Version: 11.7 |   
 |-------------------------------+----------------------+----------------------+   
 | GPU Name TCC/WDDM | Bus-Id Disp.A | Volatile Uncorr. ECC |   
 | Fan Temp Perf Pwr:Usage/Cap| Memory-Usage | GPU-Util Compute M. |   
 | | | MIG M. |   
 |===============================+======================+======================|   
 | 0 Quadro P600 WDDM | 00000000:C1:00.0 On | N/A |   
 | 34% 47C P5 N/A / N/A | 1723MiB / 2048MiB | 13% Default |   
 | | | N/A |   
 +-------------------------------+----------------------+----------------------+   
 | 1 NVIDIA TITAN RTX WDDM | 00000000:E1:00.0 Off | N/A |   
 | 41% 29C P8 8W / 280W | 2069MiB / 24576MiB | 0% Default |   
 | | | N/A |   
 +-------------------------------+----------------------+----------------------+

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

XNAT integration with MONAI Label for training

**Issue No. 1342 opened by YuanSingapore on 2023-03-17 at 06:04:18 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* a few  
customers from APJ are looking for an end-to-end solution with  
MONAILabel integrated with XNAT. We have successfully set up the  
connection. However, the training can\'t be triggered from XNAT yet.  
  
\*\*Describe the solution you\'d like\*\* We would hope to get the training  
pipeline be triggered through XNAT so that the customer can demonstrate  
the end-to-end workflow.  
  
\*\*Describe alternatives you\'ve considered\*\* Currently, we triggered  
training through 3Dslicer which is not ideal for customer  
  
\*\*Additional context\*\* Add any other context or screenshots about the  
feature request here.  
  
Appreciate your support on the case.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Set up MONAILabel with XNAT for training

**Issue No. 1339 opened by YuanSingapore on 2023-03-15 at 05:38:08 Type: None**

## MONAI Integration with XNAT  
  
Hi Team,  
  
I am building a few working prototypes for our key customers \--  
hospitals and research institute \-- in APJ area, to help them set up a  
training & inference work environment for medical imaging. Their  
preference is to have XNAT and MONAI Label as for front end and servers  
respectively. Until now, we have already successfully connected XNAT and  
MONAI Label for inference. However, my customer cannot complete the  
training process due to some components are not processed yet within  
MONAI Label.  
  
Can I request the team to help to prioritise this activity so that we  
can get this pipeline running end to end?  
  
Many Thanks  
  
```{=html}  
</div>  
```

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

CUDA is not getting find by Monailabel

**Issue No. 1337 opened by keyurradia on 2023-03-13 at 22:32:47 Type: None**

\*\*Describe the bug\*\* I am using pc with gpu nvidia rtx 4070 with 12gb  
memory and is cuda enabeled. I have installed pytorch with cuda option.  
the Monailabel is still not able to find or work with cuda on the pc.  
can you please help with that.  
  
\*\*Server logs\*\* 2023-03-13 23:19:59,209 - USING:: version = False  
2023-03-13 23:19:59,209 - USING:: app =  
C:\\Users\\keyur\\apps\\radiology 2023-03-13 23:19:59,209 - USING::  
studies = C:\\Users\\keyur\\datasets\\training 2023-03-13 23:19:59,209 -  
USING:: verbose = INFO 2023-03-13 23:19:59,209 - USING:: conf =  
\[\[\'models\', \'segmentation\'\]\] 2023-03-13 23:19:59,210 - USING::  
host = 0.0.0.0 2023-03-13 23:19:59,210 - USING:: port = 8000 2023-03-13  
23:19:59,210 - USING:: uvicorn\_app = monailabel.app:app 2023-03-13  
23:19:59,210 - USING:: ssl\_keyfile = None 2023-03-13 23:19:59,210 -  
USING:: ssl\_certfile = None 2023-03-13 23:19:59,210 - USING::  
ssl\_keyfile\_password = None 2023-03-13 23:19:59,211 - USING::  
ssl\_ca\_certs = None 2023-03-13 23:19:59,211 - USING:: workers = None  
2023-03-13 23:19:59,211 - USING:: limit\_concurrency = None 2023-03-13  
23:19:59,211 - USING:: access\_log = False 2023-03-13 23:19:59,211 -  
USING:: log\_config = None 2023-03-13 23:19:59,211 - USING:: dryrun =  
False 2023-03-13 23:19:59,212 - USING:: action = start\_server 2023-03-13  
23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_API\_STR = 2023-03-13  
23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_PROJECT\_NAME = MONAILabel  
2023-03-13 23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_APP\_DIR =  
2023-03-13 23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_STUDIES =  
2023-03-13 23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_AUTH\_ENABLE = False  
2023-03-13 23:19:59,212 - ENV SETTINGS:: MONAI\_LABEL\_AUTH\_DB =  
2023-03-13 23:19:59,213 - ENV SETTINGS:: MONAI\_LABEL\_APP\_CONF = \'{}\'  
2023-03-13 23:19:59,213 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_TRAIN = True  
2023-03-13 23:19:59,213 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_STRATEGY =  
True 2023-03-13 23:19:59,213 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_SCORING  
= True 2023-03-13 23:19:59,213 - ENV SETTINGS::  
MONAI\_LABEL\_TASKS\_BATCH\_INFER = True 2023-03-13 23:19:59,213 - ENV  
SETTINGS:: MONAI\_LABEL\_DATASTORE = 2023-03-13 23:19:59,213 - ENV  
SETTINGS:: MONAI\_LABEL\_DATASTORE\_URL = 2023-03-13 23:19:59,213 - ENV  
SETTINGS:: MONAI\_LABEL\_DATASTORE\_USERNAME = 2023-03-13 23:19:59,214 -  
ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_PASSWORD = 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_API\_KEY = 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_CACHE\_PATH =  
2023-03-13 23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_PROJECT =  
2023-03-13 23:19:59,214 - ENV SETTINGS::  
MONAI\_LABEL\_DATASTORE\_ASSET\_PATH = 2023-03-13 23:19:59,214 - ENV  
SETTINGS:: MONAI\_LABEL\_DATASTORE\_DSA\_ANNOTATION\_GROUPS = 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_USERNAME = 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_PASSWORD = 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_CACHE\_PATH =  
2023-03-13 23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_QIDO\_PREFIX = None  
2023-03-13 23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_WADO\_PREFIX = None  
2023-03-13 23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_STOW\_PREFIX = None  
2023-03-13 23:19:59,214 - ENV SETTINGS::  
MONAI\_LABEL\_DICOMWEB\_FETCH\_BY\_FRAME = False 2023-03-13 23:19:59,214 -  
ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_CONVERT\_TO\_NIFTI = True 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_SEARCH\_FILTER =  
\'{\"Modality\": \"CT\"}\' 2023-03-13 23:19:59,214 - ENV SETTINGS::  
MONAI\_LABEL\_DICOMWEB\_CACHE\_EXPIRY = 180 2023-03-13 23:19:59,214 - ENV  
SETTINGS:: MONAI\_LABEL\_DICOMWEB\_PROXY\_TIMEOUT = 30.0 2023-03-13  
23:19:59,214 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_READ\_TIMEOUT = 5.0  
2023-03-13 23:19:59,214 - ENV SETTINGS::  
MONAI\_LABEL\_DATASTORE\_AUTO\_RELOAD = True 2023-03-13 23:19:59,215 - ENV  
SETTINGS:: MONAI\_LABEL\_DATASTORE\_READ\_ONLY = False 2023-03-13  
23:19:59,215 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_FILE\_EXT =  
\'\[\"\*.nii.gz\", \"\*.nii\", \"\*.nrrd\", \"\*.jpg\", \"\*.png\",  
\"\*.tif\", \"\*.svs\", \"\*.xml\"\]\' 2023-03-13 23:19:59,215 - ENV  
SETTINGS:: MONAI\_LABEL\_SERVER\_PORT = 8000 2023-03-13 23:19:59,215 - ENV  
SETTINGS:: MONAI\_LABEL\_CORS\_ORIGINS = \'\[\]\' 2023-03-13 23:19:59,215 -  
ENV SETTINGS:: MONAI\_LABEL\_SESSIONS = True 2023-03-13 23:19:59,215 - ENV  
SETTINGS:: MONAI\_LABEL\_SESSION\_PATH = 2023-03-13 23:19:59,215 - ENV  
SETTINGS:: MONAI\_LABEL\_SESSION\_EXPIRY = 3600 2023-03-13 23:19:59,215 -  
ENV SETTINGS:: MONAI\_LABEL\_INFER\_CONCURRENCY = -1 2023-03-13  
23:19:59,215 - ENV SETTINGS:: MONAI\_LABEL\_INFER\_TIMEOUT = 600 2023-03-13  
23:19:59,216 - ENV SETTINGS:: MONAI\_LABEL\_TRACKING\_ENABLED = True  
2023-03-13 23:19:59,216 - ENV SETTINGS:: MONAI\_LABEL\_TRACKING\_URI =  
2023-03-13 23:19:59,216 - ENV SETTINGS:: MONAI\_ZOO\_SOURCE = github  
2023-03-13 23:19:59,216 - ENV SETTINGS:: MONAI\_ZOO\_REPO =  
Project-MONAI/model-zoo/hosting\_storage\_v1 2023-03-13 23:19:59,219 - ENV  
SETTINGS:: MONAI\_ZOO\_AUTH\_TOKEN = 2023-03-13 23:19:59,219 - ENV  
SETTINGS:: MONAI\_LABEL\_AUTO\_UPDATE\_SCORING = True 2023-03-13  
23:19:59,219 - Allow Origins: \[\'\*\'\] \[2023-03-13 23:20:14,054\]  
\[3404\] \[MainThread\] \[INFO\] (uvicorn.error:75) - Started server  
process \[3404\] \[2023-03-13 23:20:14,056\] \[3404\] \[MainThread\]  
\[INFO\] (uvicorn.error:45) - Waiting for application startup.  
\[2023-03-13 23:20:14,058\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.interfaces.utils.app:37) - Initializing App from:  
C:\\Users\\keyur\\apps\\radiology; studies:  
C:\\Users\\keyur\\datasets\\training; conf: {\'models\':  
\'segmentation\'} \[2023-03-13 23:20:14,274\] \[3404\] \[MainThread\]  
\[INFO\] (monailabel.utils.others.class\_utils:57) - Subclass for  
MONAILabelApp Found: \<class \'main.MyApp\'\> \[2023-03-13  
23:20:14,290\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepedit.DeepEdit\'\> \[2023-03-13  
23:20:14,292\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_2d.Deepgrow2D\'\> \[2023-03-13  
23:20:14,293\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_3d.Deepgrow3D\'\> \[2023-03-13  
23:20:14,296\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.localization\_spine.LocalizationSpine\'\>  
\[2023-03-13 23:20:14,300\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.localization\_vertebra.LocalizationVertebra\'\>  
\[2023-03-13 23:20:14,324\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation.Segmentation\'\> \[2023-03-13  
23:20:14,333\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation\_spleen.SegmentationSpleen\'\>  
\[2023-03-13 23:20:14,348\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.segmentation\_vertebra.SegmentationVertebra\'\>  
\[2023-03-13 23:20:14,362\] \[3404\] \[MainThread\] \[INFO\] (main:93) -  
+++ Adding Model: segmentation =\> lib.configs.segmentation.Segmentation  
\[2023-03-13 23:20:14,451\] \[3404\] \[MainThread\] \[INFO\] (main:96) -  
+++ Using Models: \[\'segmentation\'\] \[2023-03-13 23:20:14,459\]  
\[3404\] \[MainThread\] \[INFO\] (monailabel.interfaces.app:134) - Init  
Datastore for: C:\\Users\\keyur\\datasets\\training \[2023-03-13  
23:20:14,472\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:130) - Auto Reload: True; Extensions:  
\[\'\*.nii.gz\', \'\*.nii\', \'\*.nrrd\', \'\*.jpg\', \'\*.png\', \'\*.tif\',  
\'\*.svs\', \'\*.xml\'\] \[2023-03-13 23:20:14,507\] \[3404\]  
\[MainThread\] \[INFO\] (monailabel.datastore.local:577) - Invalidate  
count: 0 \[2023-03-13 23:20:14,519\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:151) - Start observing external  
modifications on datastore (AUTO RELOAD) \[2023-03-13 23:20:14,538\]  
\[3404\] \[MainThread\] \[INFO\] (main:126) - +++ Adding Inferer::  
segmentation =\> \<lib.infers.segmentation.Segmentation object at  
0x000001F12698C2E0\> \[2023-03-13 23:20:14,539\] \[3404\] \[MainThread\]  
\[INFO\] (main:191) - {\'segmentation\':  
\<lib.infers.segmentation.Segmentation object at 0x000001F12698C2E0\>,  
\'Histogram+GraphCut\':  
\<monailabel.scribbles.infer.HistogramBasedGraphCut object at  
0x000001F12AB54580\>, \'GMM+GraphCut\':  
\<monailabel.scribbles.infer.GMMBasedGraphCut object at  
0x000001F12AB54550\>} \[2023-03-13 23:20:14,540\] \[3404\]  
\[MainThread\] \[INFO\] (main:206) - +++ Adding Trainer:: segmentation  
=\> \<lib.trainers.segmentation.Segmentation object at  
0x000001F12AB7E6A0\> \[2023-03-13 23:20:14,542\] \[3404\] \[MainThread\]  
\[INFO\] (monailabel.utils.sessions:51) - Session Path:  
C:\\Users\\keyur.cache\\monailabel\\sessions \[2023-03-13 23:20:14,543\]  
\[3404\] \[MainThread\] \[INFO\] (monailabel.utils.sessions:52) -  
Session Expiry (max): 3600 \[2023-03-13 23:20:14,543\] \[3404\]  
\[MainThread\] \[INFO\] (monailabel.interfaces.app:468) - App Init -  
completed \[2023-03-13 23:20:14,544\] \[timeloop\] \[INFO\] Starting  
Timeloop.. \[2023-03-13 23:20:14,544\] \[3404\] \[MainThread\] \[INFO\]  
(timeloop:60) - Starting Timeloop.. \[2023-03-13 23:20:14,549\]  
\[timeloop\] \[INFO\] Registered job \<function  
MONAILabelApp.on\_init\_complete.`<locals>`{=html}.run\_scheduler at  
0x000001F1269F0700\> \[2023-03-13 23:20:14,549\] \[3404\] \[MainThread\]  
\[INFO\] (timeloop:42) - Registered job \<function  
MONAILabelApp.on\_init\_complete.`<locals>`{=html}.run\_scheduler at  
0x000001F1269F0700\> \[2023-03-13 23:20:14,549\] \[timeloop\] \[INFO\]  
Timeloop now started. Jobs will run based on the interval set  
\[2023-03-13 23:20:14,549\] \[3404\] \[MainThread\] \[INFO\]  
(timeloop:63) - Timeloop now started. Jobs will run based on the  
interval set \[2023-03-13 23:20:14,550\] \[3404\] \[MainThread\]  
\[INFO\] (uvicorn.error:59) - Application startup complete. \[2023-03-13  
23:20:14,552\] \[3404\] \[MainThread\] \[INFO\] (uvicorn.error:206) -  
Uvicorn running on <http://0.0.0.0:8000> (Press CTRL+C to quit)  
\[2023-03-13 23:20:36,423\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.endpoints.activelearning:43) - Active Learning Request:  
{\'strategy\': \'random\', \'client\_id\': \'user-xyz\'} \[2023-03-13  
23:21:50,060\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.endpoints.datastore:67) - Image: 23.02.14.17; File:  
\<starlette.datastructures.UploadFile object at 0x000001F12AF3F7F0\>;  
params: {\"client\_id\": \"user-xyz\"} \[2023-03-13 23:21:50,116\]  
\[3404\] \[MainThread\] \[INFO\] (monailabel.datastore.local:439) -  
Adding Image: 23.02.14.17 =\>  
C:\\Users\\keyur\\AppData\\Local\\Temp\\tmpv0s5oyc5.nii.gz \[2023-03-13  
23:21:50,698\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.endpoints.datastore:100) - Saving Label for 23.02.14.17 for  
tag: final by admin \[2023-03-13 23:21:50,702\] \[3404\] \[MainThread\]  
\[INFO\] (monailabel.endpoints.datastore:111) - Save Label params:  
{\"label\_info\": \[{\"name\": \"liver\", \"idx\": 1}, {\"name\":  
\"venaporta\", \"idx\": 2}, {\"name\": \"livervein\", \"idx\": 3},  
{\"name\": \"venacava\", \"idx\": 4}, {\"name\": \"tumor\", \"idx\":  
5}\], \"client\_id\": \"user-xyz\"} \[2023-03-13 23:21:50,703\] \[3404\]  
\[MainThread\] \[INFO\] (monailabel.datastore.local:486) - Saving Label  
for Image: 23.02.14.17; Tag: final; Info: {\'label\_info\': \[{\'name\':  
\'liver\', \'idx\': 1}, {\'name\': \'venaporta\', \'idx\': 2},  
{\'name\': \'livervein\', \'idx\': 3}, {\'name\': \'venacava\', \'idx\':  
4}, {\'name\': \'tumor\', \'idx\': 5}\], \'client\_id\': \'user-xyz\'}  
\[2023-03-13 23:21:50,704\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:494) - Adding Label: 23.02.14.17 =\> final  
=\> C:\\Users\\keyur\\AppData\\Local\\Temp\\tmp2qgif4r\\_.nii.gz  
\[2023-03-13 23:21:50,710\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:510) - Label Info: {\'label\_info\':  
\[{\'name\': \'liver\', \'idx\': 1}, {\'name\': \'venaporta\', \'idx\':  
2}, {\'name\': \'livervein\', \'idx\': 3}, {\'name\': \'venacava\',  
\'idx\': 4}, {\'name\': \'tumor\', \'idx\': 5}\], \'client\_id\':  
\'user-xyz\', \'ts\': 1678746110, \'name\': \'23.02.14.17.nii.gz\'}  
\[2023-03-13 23:21:50,715\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.interfaces.app:492) - New label saved for: 23.02.14.17 =\>  
23.02.14.17 \[2023-03-13 23:21:50,775\] \[3404\] \[Thread-1\] \[INFO\]  
(monailabel.datastore.local:624) - Adding New Label: final5 =\>  
23.02.14.17 =\> 23.02.14.17.nii \[2023-03-13 23:21:50,780\] \[3404\]  
\[Thread-1\] \[INFO\] (monailabel.datastore.local:577) - Invalidate  
count: 1 \[2023-03-13 23:21:55,028\] \[3404\] \[MainThread\] \[INFO\]  
(monailabel.utils.async\_tasks.task:41) - Train request: {\'model\':  
\'segmentation\', \'name\': \'train\_01\', \'pretrained\': True,  
\'device\': \'cuda\', \'max\_epochs\': 50, \'early\_stop\_patience\': -1,  
\'val\_split\': 0.2, \'train\_batch\_size\': 1, \'val\_batch\_size\': 1,  
\'multi\_gpu\': True, \'gpus\': \'all\', \'dataset\':  
\'SmartCacheDataset\', \'dataloader\': \'ThreadDataLoader\',  
\'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'client\_id\': \'user-xyz\'}  
\[2023-03-13 23:21:55,039\] \[3404\] \[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:49) - Before::  
C:\\Users\\keyur.conda\\envs; \[2023-03-13 23:21:55,041\] \[3404\]  
\[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:53) - After::  
C:\\Users\\keyur.conda\\envs; \[2023-03-13 23:21:55,042\] \[3404\]  
\[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:65) - COMMAND::  
C:\\Users\\keyur.conda\\envs\\monailabel-env\\python.exe -m  
monailabel.interfaces.utils.app -m train -r  
{\"model\":\"segmentation\",\"name\":\"train\_01\",\"pretrained\":true,\"device\":\"cuda\",\"max\_epochs\":50,\"early\_stop\_patience\":-1,\"val\_split\":0.2,\"train\_batch\_size\":1,\"val\_batch\_size\":1,\"multi\_gpu\":true,\"gpus\":\"all\",\"dataset\":\"SmartCacheDataset\",\"dataloader\":\"ThreadDataLoader\",\"tracking\":\"mlflow\",\"tracking\_uri\":\"\",\"tracking\_experiment\_name\":\"\",\"client\_id\":\"user-xyz\"}  
\[2023-03-13 23:21:55,668\] \[22344\] \[MainThread\] \[INFO\]  
(\*\*main\*\*:37) - Initializing App from:  
C:\\Users\\keyur\\apps\\radiology; studies:  
C:\\Users\\keyur\\datasets\\training; conf: {\'models\':  
\'segmentation\'} \[2023-03-13 23:21:59,895\] \[22344\] \[MainThread\]  
\[INFO\] (monailabel.utils.others.class\_utils:57) - Subclass for  
MONAILabelApp Found: \<class \'main.MyApp\'\> \[2023-03-13  
23:21:59,903\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepedit.DeepEdit\'\> \[2023-03-13  
23:21:59,903\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_2d.Deepgrow2D\'\> \[2023-03-13  
23:21:59,904\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.deepgrow\_3d.Deepgrow3D\'\> \[2023-03-13  
23:21:59,905\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.localization\_spine.LocalizationSpine\'\>  
\[2023-03-13 23:21:59,905\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.localization\_vertebra.LocalizationVertebra\'\>  
\[2023-03-13 23:21:59,906\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation.Segmentation\'\> \[2023-03-13  
23:21:59,907\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class \'lib.configs.segmentation\_spleen.SegmentationSpleen\'\>  
\[2023-03-13 23:21:59,907\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.utils.others.class\_utils:57) - Subclass for TaskConfig  
Found: \<class  
\'lib.configs.segmentation\_vertebra.SegmentationVertebra\'\>  
\[2023-03-13 23:21:59,907\] \[22344\] \[MainThread\] \[INFO\]  
(main:93) - +++ Adding Model: segmentation =\>  
lib.configs.segmentation.Segmentation \[2023-03-13 23:21:59,974\]  
\[22344\] \[MainThread\] \[INFO\] (main:96) - +++ Using Models:  
\[\'segmentation\'\] \[2023-03-13 23:21:59,975\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.interfaces.app:134) - Init Datastore  
for: C:\\Users\\keyur\\datasets\\training \[2023-03-13 23:21:59,975\]  
\[22344\] \[MainThread\] \[INFO\] (monailabel.datastore.local:130) -  
Auto Reload: False; Extensions: \[\'\*.nii.gz\', \'\*.nii\', \'\*.nrrd\',  
\'\*.jpg\', \'\*.png\', \'\*.tif\', \'\*.svs\', \'\\*.xml\'\] \[2023-03-13  
23:21:59,986\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.datastore.local:577) - Invalidate count: 0 \[2023-03-13  
23:21:59,986\] \[22344\] \[MainThread\] \[INFO\] (main:126) - +++ Adding  
Inferer:: segmentation =\> \<lib.infers.segmentation.Segmentation object  
at 0x0000013C3E77D9D0\> \[2023-03-13 23:21:59,986\] \[22344\]  
\[MainThread\] \[INFO\] (main:191) - {\'segmentation\':  
\<lib.infers.segmentation.Segmentation object at 0x0000013C3E77D9D0\>,  
\'Histogram+GraphCut\':  
\<monailabel.scribbles.infer.HistogramBasedGraphCut object at  
0x0000013C3FC5FF40\>, \'GMM+GraphCut\':  
\<monailabel.scribbles.infer.GMMBasedGraphCut object at  
0x0000013C3FC5FF10\>} \[2023-03-13 23:21:59,986\] \[22344\]  
\[MainThread\] \[INFO\] (main:206) - +++ Adding Trainer:: segmentation  
=\> \<lib.trainers.segmentation.Segmentation object at  
0x0000013C3FC5FFA0\> \[2023-03-13 23:21:59,986\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.utils.sessions:51) - Session Path:  
C:\\Users\\keyur.cache\\monailabel\\sessions \[2023-03-13 23:21:59,986\]  
\[22344\] \[MainThread\] \[INFO\] (monailabel.utils.sessions:52) -  
Session Expiry (max): 3600 \[2023-03-13 23:21:59,986\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:432) - Train  
Request (input): {\'model\': \'segmentation\', \'name\': \'train\_01\',  
\'pretrained\': True, \'device\': \'cuda\', \'max\_epochs\': 50,  
\'early\_stop\_patience\': -1, \'val\_split\': 0.2, \'train\_batch\_size\':  
1, \'val\_batch\_size\': 1, \'multi\_gpu\': True, \'gpus\': \'all\',  
\'dataset\': \'SmartCacheDataset\', \'dataloader\':  
\'ThreadDataLoader\', \'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'client\_id\': \'user-xyz\',  
\'local\_rank\': 0} \[2023-03-13 23:21:59,986\] \[22344\] \[MainThread\]  
\[INFO\] (monailabel.tasks.train.basic\_train:442) -  
CUDA\_VISIBLE\_DEVICES: None \[2023-03-13 23:21:59,988\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:447) -  
Distributed/Multi GPU is limited \[2023-03-13 23:21:59,988\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:462) -  
Distributed Training = FALSE \[2023-03-13 23:21:59,988\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:489) - 0 -  
Train Request (final): {\'name\': \'train\_01\', \'pretrained\': True,  
\'device\': \'cuda\', \'max\_epochs\': 50, \'early\_stop\_patience\': -1,  
\'val\_split\': 0.2, \'train\_batch\_size\': 1, \'val\_batch\_size\': 1,  
\'multi\_gpu\': False, \'gpus\': \'all\', \'dataset\':  
\'SmartCacheDataset\', \'dataloader\': \'ThreadDataLoader\',  
\'tracking\': \'mlflow\', \'tracking\_uri\': \'\',  
\'tracking\_experiment\_name\': \'\', \'model\': \'segmentation\',  
\'client\_id\': \'user-xyz\', \'local\_rank\': 0, \'run\_id\':  
\'20230313\_232159\'} \[2023-03-13 23:21:59,988\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:622) - 0 -  
Using Device: cpu; IDX: None \[2023-03-13 23:21:59,988\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:515) -  
Run/Output Path:  
C:\\Users\\keyur\\apps\\radiology\\model\\segmentation\\train\_01  
\[2023-03-13 23:21:59,988\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:531) - Tracking: mlflow \[2023-03-13  
23:21:59,988\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:532) - Tracking URI:  
file:///C:/Users/keyur/apps/radiology/model/segmentation/train\_01/mlruns;  
\[2023-03-13 23:21:59,988\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:533) - Tracking Experiment Name:  
segmentation; Run Name: run\_20230313\_232159 \[2023-03-13 23:21:59,989\]  
\[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:410) - Total Records for Training: 7  
\[2023-03-13 23:21:59,989\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:411) - Total Records for Validation:  
2 Loading dataset: 0%\| \| 0/2 \[00:00\<?, ?it/s\] Loading dataset:  
50%\|##### \| 1/2 \[00:04\<00:04, 4.95s/it\] Loading dataset:  
100%\|##########\| 2/2 \[00:08\<00:00, 4.10s/it\] Loading dataset:  
100%\|##########\| 2/2 \[00:08\<00:00, 4.23s/it\] cache\_num is greater  
or equal than dataset length, fall back to regular  
monai.data.CacheDataset. \[2023-03-13 23:22:08,462\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:328) - 0 -  
Records for Validation: 2 \[2023-03-13 23:22:08,468\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:318) - 0 -  
Adding Validation to run every \'1\' interval \[2023-03-13  
23:22:08,471\] \[22344\] \[MainThread\] \[INFO\]  
(monailabel.tasks.train.basic\_train:710) - 0 - Load Path  
C:\\Users\\keyur\\apps\\radiology\\model\\pretrained\_segmentation.pt  
Loading dataset: 0%\| \| 0/7 \[00:00\<?, ?it/s\] Loading dataset:  
14%\|#4 \| 1/7 \[00:07\<00:47, 7.93s/it\] Loading dataset: 29%\|##8 \|  
2/7 \[00:13\<00:33, 6.69s/it\] Loading dataset: 43%\|####2 \| 3/7  
\[00:19\<00:25, 6.47s/it\] Loading dataset: 57%\|#####7 \| 4/7  
\[00:23\<00:15, 5.14s/it\] Loading dataset: 71%\|#######1 \| 5/7  
\[00:28\<00:10, 5.27s/it\] Loading dataset: 86%\|########5 \| 6/7  
\[00:35\<00:05, 5.82s/it\] Loading dataset: 100%\|##########\| 7/7  
\[00:42\<00:00, 6.14s/it\] Loading dataset: 100%\|##########\| 7/7  
\[00:42\<00:00, 6.04s/it\] \[2023-03-13 23:22:50,735\] \[22344\]  
\[MainThread\] \[INFO\] (monailabel.tasks.train.basic\_train:264) - 0 -  
Records for Training: 7 torch.cuda.amp.GradScaler is enabled, but CUDA  
is not available. Disabling. \[2023-03-13 23:22:50,750\] \[22344\]  
\[MainThread\] \[INFO\] (ignite.engine.engine.SupervisedTrainer:876) -  
Engine run resuming from iteration 0, epoch 0 until 50 epochs  
\[2023-03-13 23:22:50,766\] \[22344\] \[MainThread\] \[ERROR\]  
(ignite.engine.engine.SupervisedTrainer:992) - Engine run is terminating  
due to exception: Attempting to deserialize object on a CUDA device but  
torch.cuda.is\_available() is False. If you are running on a CPU-only  
machine, please use torch.load with map\_location=torch.device(\'cpu\')  
to map your storages to the CPU. \[2023-03-13 23:22:50,766\] \[22344\]  
\[MainThread\] \[ERROR\] (ignite.engine.engine.SupervisedTrainer:180) -  
Exception: Attempting to deserialize object on a CUDA device but  
torch.cuda.is\_available() is False. If you are running on a CPU-only  
machine, please use torch.load with map\_location=torch.device(\'cpu\')  
to map your storages to the CPU. Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 946, in \\_internal\_run\_as\_gen self.\\_fire\_event(Events.STARTED)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monai\\handlers\\checkpoint\_loader.py\",  
line 107, in \*\*call\*\* checkpoint = torch.load(self.load\_path,  
map\_location=self.map\_location) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 789, in load return \\_load(opened\_zipfile, map\_location,  
pickle\_module, \\*\\*pickle\_load\_args) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1131, in \\_load result = unpickler.load() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1101, in persistent\_load load\_tensor(dtype, nbytes, key,  
\\_maybe\_decode\_ascii(location)) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1083, in load\_tensor wrap\_storage=restore\_location(storage,  
location), File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 215, in default\_restore\_location result = fn(storage, location)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 182, in \\_cuda\_deserialize device = validate\_cuda\_device(location)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 166, in validate\_cuda\_device raise RuntimeError(\'Attempting to  
deserialize object on a CUDA \' RuntimeError: Attempting to deserialize  
object on a CUDA device but torch.cuda.is\_available() is False. If you  
are running on a CPU-only machine, please use torch.load with  
map\_location=torch.device(\'cpu\') to map your storages to the CPU.  
Traceback (most recent call last): File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\runpy.py\", line  
197, in \\_run\_module\_as\_main return \\_run\_code(code, main\_globals, None,  
File \"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\runpy.py\",  
line 87, in \\_run\_code exec(code, run\_globals) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monailabel\\interfaces\\utils\\app.py\",  
line 128, in `<module>`{=html} run\_main() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monailabel\\interfaces\\utils\\app.py\",  
line 113, in run\_main result = a.train(request) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monailabel\\interfaces\\app.py\",  
line 422, in train result = task(request, self.datastore()) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monailabel\\tasks\\train\\basic\_train.py\",  
line 463, in \*\*call\*\* res = self.train(0, world\_size, req, datalist)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monailabel\\tasks\\train\\basic\_train.py\",  
line 552, in train context.trainer.run() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monai\\engines\\trainer.py\",  
line 53, in run super().run() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monai\\engines\\workflow.py\",  
line 281, in run super().run(data=self.data\_loader,  
max\_epochs=self.state.max\_epochs) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 892, in run return self.\\_internal\_run() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 935, in \\_internal\_run return next(self.\\_internal\_run\_generator)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 993, in \\_internal\_run\_as\_gen self.\\_handle\_exception(e) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 636, in \\_handle\_exception  
self.\\_fire\_event(Events.EXCEPTION\_RAISED, e) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monai\\handlers\\stats\_handler.py\",  
line 181, in exception\_raised raise e File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 946, in \\_internal\_run\_as\_gen self.\\_fire\_event(Events.STARTED)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\ignite\\engine\\engine.py\",  
line 425, in \\_fire\_event func(\\*first, \\*(event\_args + others),  
\\*\\*kwargs) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\monai\\handlers\\checkpoint\_loader.py\",  
line 107, in \*\*call\*\* checkpoint = torch.load(self.load\_path,  
map\_location=self.map\_location) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 789, in load return \\_load(opened\_zipfile, map\_location,  
pickle\_module, \\*\\*pickle\_load\_args) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1131, in \\_load result = unpickler.load() File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1101, in persistent\_load load\_tensor(dtype, nbytes, key,  
\\_maybe\_decode\_ascii(location)) File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 1083, in load\_tensor wrap\_storage=restore\_location(storage,  
location), File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 215, in default\_restore\_location result = fn(storage, location)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 182, in \\_cuda\_deserialize device = validate\_cuda\_device(location)  
File  
\"C:\\Users\\keyur.conda\\envs\\monailabel-env\\lib\\site-packages\\torch\\serialization.py\",  
line 166, in validate\_cuda\_device raise RuntimeError(\'Attempting to  
deserialize object on a CUDA \' RuntimeError: Attempting to deserialize  
object on a CUDA device but torch.cuda.is\_available() is False. If you  
are running on a CPU-only machine, please use torch.load with  
map\_location=torch.device(\'cpu\') to map your storages to the CPU.  
\[2023-03-13 23:22:51,452\] \[3404\] \[ThreadPoolExecutor-2\_0\] \[INFO\]  
(monailabel.utils.async\_tasks.utils:83) - Return code: 1

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Cannot read properties of undefined (reading 'byteLength')

**Issue No. 1336 opened by junxiant on 2023-03-13 at 15:40:51 Type: None**

Hello,  
  
I am trying to run MonaiLabel segmentation on a CT scan, but I  
encountered this error.  
![image](https://user-images.githubusercontent.com/25401067/224734485-7daf6f29-52cc-4afa-bea1-f6fa91c0286b.png)  
  
Server logs seems fine as there are no errors here.  
![image](https://user-images.githubusercontent.com/25401067/224749162-7845b5bb-fd75-4279-a7a1-389e39f24306.png)  
  
Could this be a dicom header issue?

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

TypeError for creating model path when running batch inference with custom monai bundle

**Issue No. 1335 opened by kbressem on 2023-03-10 at 22:38:17 Type: None**

\*\*Describe the bug\*\* I am running a custom segmentation model based on  
monai bundle and want to perform batch inference to speed up  
annotations. While single inference executed from within the monailabel  
slicer app works well, with batch inference I get the following error:  
  
``` python  
TypeError: join() argument must be str, bytes, or os.PathLike object, not 'list'  
```  
  
\*\*Server logs\*\*  
  
``` bash  
myapp-monailabel-1 |  
myapp-monailabel-1 | =============  
myapp-monailabel-1 | == PyTorch ==  
myapp-monailabel-1 | =============  
myapp-monailabel-1 |  
myapp-monailabel-1 | NVIDIA Release 22.10 (build 46164382)  
myapp-monailabel-1 | PyTorch Version 1.13.0a0+d0d6b1f  
myapp-monailabel-1 |  
myapp-monailabel-1 | Container image Copyright (c) 2022, NVIDIA CORPORATION & AFFILIATES. All rights reserved.  
myapp-monailabel-1 |  
myapp-monailabel-1 | Copyright (c) 2014-2022 Facebook Inc.  
myapp-monailabel-1 | Copyright (c) 2011-2014 Idiap Research Institute (Ronan Collobert)  
myapp-monailabel-1 | Copyright (c) 2012-2014 Deepmind Technologies (Koray Kavukcuoglu)  
myapp-monailabel-1 | Copyright (c) 2011-2012 NEC Laboratories America (Koray Kavukcuoglu)  
myapp-monailabel-1 | Copyright (c) 2011-2013 NYU (Clement Farabet)  
myapp-monailabel-1 | Copyright (c) 2006-2010 NEC Laboratories America (Ronan Collobert, Leon Bottou, Iain Melvin, Jason Weston)  
myapp-monailabel-1 | Copyright (c) 2006 Idiap Research Institute (Samy Bengio)  
myapp-monailabel-1 | Copyright (c) 2001-2004 Idiap Research Institute (Ronan Collobert, Samy Bengio, Johnny Mariethoz)  
myapp-monailabel-1 | Copyright (c) 2015 Google Inc.  
myapp-monailabel-1 | Copyright (c) 2015 Yangqing Jia  
myapp-monailabel-1 | Copyright (c) 2013-2016 The Caffe contributors  
myapp-monailabel-1 | All rights reserved.  
myapp-monailabel-1 |  
myapp-monailabel-1 | Various files include modifications (c) NVIDIA CORPORATION & AFFILIATES. All rights reserved.  
myapp-monailabel-1 |  
myapp-monailabel-1 | This container image and its contents are governed by the NVIDIA Deep Learning Container License.  
myapp-monailabel-1 | By pulling and using the container, you accept the terms and conditions of this license:  
myapp-monailabel-1 | https://developer.nvidia.com/ngc/nvidia-deep-learning-container-license  
myapp-monailabel-1 |  
myapp-monailabel-1 | Using PYTHONPATH=/opt:  
myapp-monailabel-1 |  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: version = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: app = /apps/monaibundle  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: studies = /data/subset500/images  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: verbose = INFO  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: conf = [['models', 'myapp']]  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: host = 0.0.0.0  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: port = 8000  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: uvicorn\_app = monailabel.app:app  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: ssl\_keyfile = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: ssl\_certfile = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: ssl\_keyfile\_password = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: ssl\_ca\_certs = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: workers = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: limit\_concurrency = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: access\_log = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: log\_config = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: dryrun = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - USING:: action = start\_server  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_API\_STR =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_PROJECT\_NAME = MONAILabel  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_APP\_DIR =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_STUDIES =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_AUTH\_ENABLE = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_AUTH\_DB =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_APP\_CONF = '{}'  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_TRAIN = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_STRATEGY = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_SCORING = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_TASKS\_BATCH\_INFER = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_URL =  
myapp-monailabel-1 | 2023-03-10 22:17:00,188 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_USERNAME =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_PASSWORD =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_API\_KEY =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_CACHE\_PATH =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_PROJECT =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_ASSET\_PATH =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_DSA\_ANNOTATION\_GROUPS =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_USERNAME =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_PASSWORD =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_CACHE\_PATH =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_QIDO\_PREFIX = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_WADO\_PREFIX = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_STOW\_PREFIX = None  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_FETCH\_BY\_FRAME = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_CONVERT\_TO\_NIFTI = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_SEARCH\_FILTER = '{"Modality": "CT"}'  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_CACHE\_EXPIRY = 180  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_PROXY\_TIMEOUT = 30.0  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DICOMWEB\_READ\_TIMEOUT = 5.0  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_AUTO\_RELOAD = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_READ\_ONLY = False  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_DATASTORE\_FILE\_EXT = '["\*.nii.gz", "\*.nii", "\*.nrrd", "\*.jpg", "\*.png", "\*.tif", "\*.svs", "\*.xml"]'  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_SERVER\_PORT = 8000  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_CORS\_ORIGINS = '[]'  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_SESSIONS = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_SESSION\_PATH =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_SESSION\_EXPIRY = 3600  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_INFER\_CONCURRENCY = -1  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_INFER\_TIMEOUT = 600  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_TRACKING\_ENABLED = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_TRACKING\_URI =  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_ZOO\_SOURCE = github  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_ZOO\_REPO = Project-MONAI/model-zoo/hosting\_storage\_v1  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_ZOO\_AUTH\_TOKEN = ghp\_xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 - ENV SETTINGS:: MONAI\_LABEL\_AUTO\_UPDATE\_SCORING = True  
myapp-monailabel-1 | 2023-03-10 22:17:00,189 -  
myapp-monailabel-1 | [2023-03-10 22:17:00,456] [1] [MainThread] [INFO] (uvicorn.error:75) - Started server process [1]  
myapp-monailabel-1 | [2023-03-10 22:17:00,456] [1] [MainThread] [INFO] (uvicorn.error:45) - Waiting for application startup.  
myapp-monailabel-1 | [2023-03-10 22:17:00,456] [1] [MainThread] [INFO] (monailabel.interfaces.utils.app:37) - Initializing App from: /apps/monaibundle; studies: /data/subset500/images; conf: {'models': 'myapp'}  
myapp-monailabel-1 | [2023-03-10 22:17:00,486] [1] [MainThread] [INFO] (monailabel.utils.others.class\_utils:57) - Subclass for MONAILabelApp Found: <class 'main.MyApp'>  
myapp-monailabel-1 | [2023-03-10 22:17:03,238] [1] [MainThread] [INFO] (monailabel.utils.others.generic:301) - +++ Adding Bundle from Local: myapp => /apps/monaibundle/model/myapp  
myapp-monailabel-1 | [2023-03-10 22:17:03,239] [1] [MainThread] [INFO] (monailabel.utils.others.generic:313) - +++ Using Bundle Models: ['myapp']  
myapp-monailabel-1 | [2023-03-10 22:17:03,239] [1] [MainThread] [INFO] (monailabel.interfaces.app:134) - Init Datastore for: /data/subset500/images  
myapp-monailabel-1 | [2023-03-10 22:17:03,240] [1] [MainThread] [INFO] (monailabel.datastore.local:130) - Auto Reload: True; Extensions: ['\*.nii.gz', '\*.nii', '\*.nrrd', '\*.jpg', '\*.png', '\*.tif', '\*.svs', '\*.xml']  
myapp-monailabel-1 | [2023-03-10 22:17:03,304] [1] [MainThread] [INFO] (monailabel.datastore.local:577) - Invalidate count: 0  
myapp-monailabel-1 | [2023-03-10 22:17:03,304] [1] [MainThread] [INFO] (monailabel.datastore.local:151) - Start observing external modifications on datastore (AUTO RELOAD)  
myapp-monailabel-1 | [2023-03-10 22:17:03,495] [1] [MainThread] [INFO] (main:64) - +++ Adding Inferer:: myapp => <monailabel.tasks.infer.bundle.BundleInferTask object at 0x7f27d71e6d90>  
myapp-monailabel-1 | [2023-03-10 22:17:03,515] [1] [MainThread] [INFO] (main:78) - +++ Adding Trainer:: myapp => <monailabel.tasks.train.bundle.BundleTrainTask object at 0x7f27ced2dd90>  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (main:88) - Active Learning Strategies:: ['random', 'first']  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (monailabel.utils.sessions:51) - Session Path: /root/.cache/monailabel/sessions  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (monailabel.utils.sessions:52) - Session Expiry (max): 3600  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (monailabel.interfaces.app:468) - App Init - completed  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (timeloop:60) - Starting Timeloop..  
myapp-monailabel-1 | Allow Origins: ['\*']  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [timeloop] [INFO] Starting Timeloop..  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [timeloop] [INFO] Registered job <function MONAILabelApp.on\_init\_complete.<locals>.run\_scheduler at 0x7f27d6e39c10>  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [timeloop] [INFO] Timeloop now started. Jobs will run based on the interval set  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (timeloop:42) - Registered job <function MONAILabelApp.on\_init\_complete.<locals>.run\_scheduler at 0x7f27d6e39c10>  
myapp-monailabel-1 | [2023-03-10 22:17:03,516] [1] [MainThread] [INFO] (timeloop:63) - Timeloop now started. Jobs will run based on the interval set  
myapp-monailabel-1 | [2023-03-10 22:17:03,517] [1] [MainThread] [INFO] (uvicorn.error:59) - Application startup complete.  
myapp-monailabel-1 | [2023-03-10 22:17:03,517] [1] [MainThread] [INFO] (uvicorn.error:206) - Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)  
myapp-monailabel-1 | [2023-03-10 22:17:12,199] [1] [MainThread] [INFO] (monailabel.utils.async\_tasks.task:41) - Batch\_infer request: {'model': 'myapp', 'images': <BatchInferImageType.IMAGES\_ALL: 'all'>, 'device': 'cuda', 'multi\_gpu': True, 'gpus': 'all', 'logging': 'WARNING', 'save\_label': True, 'label\_tag': 'original', 'max\_workers': 1, 'max\_batch\_size': 0, 'bundle\_path': '/apps/monaibundle/model/myapp'}  
myapp-monailabel-1 | [2023-03-10 22:17:12,200] [1] [ThreadPoolExecutor-2\_0] [INFO] (monailabel.utils.async\_tasks.utils:49) - Before:: /opt:  
myapp-monailabel-1 | [2023-03-10 22:17:12,202] [1] [ThreadPoolExecutor-2\_0] [INFO] (monailabel.utils.async\_tasks.utils:53) - After:: /opt::/apps/monaibundle/model/myapp  
myapp-monailabel-1 | [2023-03-10 22:17:12,203] [1] [ThreadPoolExecutor-2\_0] [INFO] (monailabel.utils.async\_tasks.utils:65) - COMMAND:: /opt/conda/bin/python -m monailabel.interfaces.utils.app -m batch\_infer -r {"model":"myapp","images":"all","device":"cuda","multi\_gpu":true,"gpus":"all","logging":"WARNING","save\_label":true,"label\_tag":"original","max\_workers":1,"max\_batch\_size":0,"bundle\_path":"/apps/monaibundle/model/myapp"}  
myapp-monailabel-1 | [2023-03-10 22:17:12,333] [116] [MainThread] [INFO] (\_\_main\_\_:37) - Initializing App from: /apps/monaibundle; studies: /data/subset500/images; conf: {'models': 'myapp'}  
myapp-monailabel-1 | [2023-03-10 22:17:15,459] [116] [MainThread] [INFO] (monailabel.utils.others.class\_utils:57) - Subclass for MONAILabelApp Found: <class 'main.MyApp'>  
myapp-monailabel-1 | [2023-03-10 22:17:18,059] [116] [MainThread] [INFO] (monailabel.utils.others.generic:301) - +++ Adding Bundle from Local: myapp => /apps/monaibundle/model/myapp  
myapp-monailabel-1 | [2023-03-10 22:17:18,059] [116] [MainThread] [INFO] (monailabel.utils.others.generic:313) - +++ Using Bundle Models: ['myapp']  
myapp-monailabel-1 | [2023-03-10 22:17:18,059] [116] [MainThread] [INFO] (monailabel.interfaces.app:134) - Init Datastore for: /data/subset500/images  
myapp-monailabel-1 | [2023-03-10 22:17:18,059] [116] [MainThread] [INFO] (monailabel.datastore.local:130) - Auto Reload: True; Extensions: ['\*.nii.gz', '\*.nii', '\*.nrrd', '\*.jpg', '\*.png', '\*.tif', '\*.svs', '\*.xml']  
myapp-monailabel-1 | [2023-03-10 22:17:18,100] [116] [MainThread] [INFO] (monailabel.datastore.local:577) - Invalidate count: 0  
myapp-monailabel-1 | [2023-03-10 22:17:18,100] [116] [MainThread] [INFO] (monailabel.datastore.local:151) - Start observing external modifications on datastore (AUTO RELOAD)  
myapp-monailabel-1 | [2023-03-10 22:17:18,197] [116] [MainThread] [INFO] (main:64) - +++ Adding Inferer:: myapp => <monailabel.tasks.infer.bundle.BundleInferTask object at 0x7fd0081145b0>  
myapp-monailabel-1 | [2023-03-10 22:17:18,217] [116] [MainThread] [INFO] (main:78) - +++ Adding Trainer:: myapp => <monailabel.tasks.train.bundle.BundleTrainTask object at 0x7fcec42411c0>  
myapp-monailabel-1 | [2023-03-10 22:17:18,217] [116] [MainThread] [INFO] (main:88) - Active Learning Strategies:: ['random', 'first']  
myapp-monailabel-1 | [2023-03-10 22:17:18,217] [116] [MainThread] [INFO] (monailabel.utils.sessions:51) - Session Path: /root/.cache/monailabel/sessions  
myapp-monailabel-1 | [2023-03-10 22:17:18,217] [116] [MainThread] [INFO] (monailabel.utils.sessions:52) - Session Expiry (max): 3600  
myapp-monailabel-1 | [2023-03-10 22:17:18,217] [116] [MainThread] [INFO] (monailabel.interfaces.tasks.batch\_infer:59) - Total number of images for batch inference: 493; Max Batch Size: 0; Label Tag: original  
myapp-monailabel-1 | [2023-03-10 22:17:18,299] [116] [MainThread] [INFO] (monailabel.interfaces.tasks.batch\_infer:132) - Running inference for image id 1.2.826.0.1.3680043.10.474.1147302139452080037190733417686299226  
myapp-monailabel-1 | Traceback (most recent call last):  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/runpy.py", line 194, in \_run\_module\_as\_main  
myapp-monailabel-1 | return \_run\_code(code, main\_globals, None,  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/runpy.py", line 87, in \_run\_code  
myapp-monailabel-1 | exec(code, run\_globals)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/utils/app.py", line 128, in <module>  
myapp-monailabel-1 | run\_main()  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/utils/app.py", line 117, in run\_main  
myapp-monailabel-1 | result = a.batch\_infer(request)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/app.py", line 344, in batch\_infer  
myapp-monailabel-1 | return self.\_batch\_infer(request, datastore if datastore else self.datastore(), self.infer)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/tasks/batch\_infer.py", line 116, in \_\_call\_\_  
myapp-monailabel-1 | res = run\_infer\_task(t, datastore, infer)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/tasks/batch\_infer.py", line 133, in run\_infer\_task  
myapp-monailabel-1 | r = infer(req, datastore)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/interfaces/app.py", line 306, in infer  
myapp-monailabel-1 | result\_file\_name, result\_json = task(request)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/site-packages/monailabel/tasks/infer/basic\_infer.py", line 272, in \_\_call\_\_  
myapp-monailabel-1 | os.path.join(os.path.dirname(self.path[0]), req.get("model\_filename", "model.pt"))  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/posixpath.py", line 90, in join  
myapp-monailabel-1 | genericpath.\_check\_arg\_types('join', a, \*p)  
myapp-monailabel-1 | File "/opt/conda/lib/python3.8/genericpath.py", line 152, in \_check\_arg\_types  
myapp-monailabel-1 | raise TypeError(f'{funcname}() argument must be str, bytes, or '  
myapp-monailabel-1 | TypeError: join() argument must be str, bytes, or os.PathLike object, not 'list'  
```  
  
\*\*Expected behavior\*\* A running batch inference, where the model creates  
segmentations for each image in the dataset and stores them in the  
`labels/original` folder.  
  
\*\*Environment\*\* I am using the latest docker image of MONAILabel. The  
Docker file is just:  
  
``` Dockerfile  
FROM projectmonai/monailabel:latest  
```  
  
The app is initialized with docker compose  
  
``` yaml  
version: "3.1"  
services:  
 monailabel:  
 build: .  
 command: >  
 monailabel start\_server  
 --app /apps/monaibundle  
 --studies /data/subset500/images  
 --conf models myapp  
 ports:  
 - 9147:8000  
 restart: unless-stopped  
 volumes:  
 - ${WORKSPACE}/${LOG\_DIR}:/app/radiology/logs/  
 - ${WORKSPACE}/data/:/data  
 - ${WORKSPACE}/apps:/apps  
 - ${WORKSPACE}/bundle:/apps/monaibundle/model/myapp/  
 environment:  
 - MONAI\_ZOO\_AUTH\_TOKEN=${MONAI\_ZOO\_AUTH\_TOKEN}  
 ipc: host  
 ulimits:  
 memlock:  
 soft: -1  
 hard: -1  
 stack:  
 soft: 67108864  
 hard: 67108864  
 deploy:  
 resources:  
 reservations:  
 devices:  
 - driver: nvidia  
 count: 1  
 capabilities: [gpu]  
```

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

New model deepgrow\_v2: with multi-head inference

**Issue No. 1328 opened by AHarouni on 2023-02-24 at 19:08:05 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\*  
  
Need to be able to train a unet like model (segresnet,etc) as it exist  
today Plus in addition add a second head to produce 2 outputs. forground  
and background which would work as deepgrow mode  
  
This network would be trained in 2 modes:  
  
- Main network mode. train segresnet as is with normal 10 labels head  
- Deep grow mode. we need to freeze all or segresnet layer. only allow  
 to train the new second head taking from the encoding / decoding and  
 have a new FC layer with 2 outputs  
  
This feature request is related to the following bugs:  
  
- [Finetune new model by freezing all layers except  
 FC](https://github.com/Project-MONAI/MONAILabel/issues/1298)  
- [Deepedit has low resolution and non responsive to new  
 clicks](https://github.com/Project-MONAI/MONAILabel/issues/1299)  
  
\*\*Describe the solution you\'d like\*\* Provide new network architecture  
with 2 heads. This would allow us to train the same core encoding and  
decoding layer or the network  
  
Allow for 2 modes of training:  
  
- train all label  
- train deepgrow Allow 2 modes to run infer:  
- Run infer as normal segmentation  
- Run deep grow  
  
\*\*Describe alternatives you\'ve considered\*\*  
  
 checkptPath = "/rootpath/train\_01/train\_model.pt"  
 checkpoint = torch.load(checkptPath)  
 model\_state\_dict = checkpoint.get(self.model\_state\_dict, checkpoint)  
 self.network.load\_state\_dict(model\_state\_dict, strict=False)

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Add Freeze option for deep grow

**Issue No. 1327 opened by AHarouni on 2023-02-24 at 17:28:41 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* I am  
working on a segmentation problem or multiple organs. I have a  
segmentation model that does ok segmenting about 70% of the structures.  
I have trained another model deep grow to help me annotate. I start by  
running the segmentation model inference then use the load original  
labels.  
  
Now I switch to deepgrow to so I can add on the missing areas say in the  
liver. when I click a couple of points then update, deepgrow clears all  
the liver labels and runs inference which is much worse than what I had.  
  
\*\*Describe the solution you\'d like\*\* Please add a freeze option to  
deepgrow to append to the existing label. that is when clicked the new  
foreground label is added to what already exist in slicer.  
  
I know that would not work when adding background points since you would  
want the model to remove extra overflowing labels. However, I never face  
this issue as it is simpler to remove extra labels by a brush  
  
\*\*Describe alternatives you\'ve considered\*\* Tried to use deep grow.  
However, The model has low resolution and doesn\'t seem to be responsive  
with new clicks

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Blackduck reports NPM and GRADLE Failure

**Issue No. 1320 opened by YanxuanLiu on 2023-02-21 at 08:36:40 Type:**

\*\*Describe the bug\*\* When we trigger blossom-ci, it will run  
vulnerability scan first. It reports NPM and GRADLE failure.  
<https://github.com/Project-MONAI/MONAILabel/actions/runs/4230364922/jobs/7347611921#step:4:162>  
  
I suggest to skip npm and gradle detector to ensure the blossom-ci runs  
well.

**Additional Information:**

* Milestones: None
* Assignees: YanxuanLiu

No comments at the moment!

Reviewer label for OHIF and Xnat

**Issue No. 1301 opened by AHarouni on 2023-02-13 at 18:09:15 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* Label  
reviewer is a great plugin. can we have this feature in the OHIF plugin.  
Once we have it in OHIF may be we could have it in Xnat too.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Deepedit has low resolution and non responsive to new clicks

**Issue No. 1299 opened by AHarouni on 2023-02-13 at 17:42:50 Type: None**

\*\*Describe the bug\*\* Testing deep edit model to annotate new volumes  
gives a great initial start. However the resolution is very low to our  
usecase. We need high res 1x1x1 or even more in the future. I think the  
reason is that deepedit resize the image. So for our large volumes of  
chest or full body CT the resolution is lost.  
  
Another issue is the model doesn\'t seem to respond to my clicks. May be  
it is relegated to the low resolution issue above or something with the  
model itself  
  
Side note: Any plans to redesign deep edit to take only 3 channels as  
deepgrow? I am not sure how we can use deepedit with more labels as the  
new 104 total segmentor CT data. Also fine tuning is basically  
impossible with this architecture.  
  
\*\*Tied solutions\*\* Retraining by replacing the resize with random crop,  
solved the resolution issue. However, I am unable to resolve the clicks  
being unresponsive  
  
 def train\_pre\_transforms(self, context: Context):  
 return [  
 LoadImaged(keys=("image", "label"), reader="ITKReader"),  
 EnsureChannelFirstd(keys=("image", "label")),  
 NormalizeLabelsInDatasetd(keys="label", label\_names=self.\_labels),  
 Orientationd(keys=["image", "label"], axcodes="RAS"),  
 # This transform may not work well for MR images  
 Spacingd(keys=("image", "label"), pixdim=self.target\_spacing, mode=("bilinear", "nearest")),  
 ScaleIntensityRanged(keys="image", a\_min=-175, a\_max=250, b\_min=0.0, b\_max=1.0, clip=True),  
 RandFlipd(keys=("image", "label"), spatial\_axis=[0], prob=0.10),  
 RandFlipd(keys=("image", "label"), spatial\_axis=[1], prob=0.10),  
 RandFlipd(keys=("image", "label"), spatial\_axis=[2], prob=0.10),  
 RandRotate90d(keys=("image", "label"), prob=0.10, max\_k=3),  
 RandShiftIntensityd(keys="image", offsets=0.10, prob=0.50),  
 # Resized(keys=("image", "label"), spatial\_size=self.spatial\_size, mode=("area", "nearest")),  
 RandCropByPosNegLabeld(keys=("image", "label"), label\_key="label",  
 pos=1, neg=1, num\_samples=self.num\_samples,spatial\_size=self.spatial\_size,),  
 # Transforms for click simulation  
 FindAllValidSlicesMissingLabelsd(keys="label", sids="sids"),  
 AddInitialSeedPointMissingLabelsd(keys="label", guidance="guidance", sids="sids"),  
 AddGuidanceSignalDeepEditd(keys="image", guidance="guidance", number\_intensity\_ch=self.number\_intensity\_ch),  
 #  
 ToTensord(keys=("image", "label")),  
 SelectItemsd(keys=("image", "label", "guidance", "label\_names")),  
 ]

**Additional Information:**

* Milestones: None
* Assignees: diazandr3s

No comments at the moment!

Finetune new model by freezing all layers except FC

**Issue No. 1298 opened by AHarouni on 2023-02-13 at 17:21:56 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* No I  
think an essential usecase is to be able to fine tune a model. Currently  
the code can load a model then continue training all the layers. I am  
looking for how can I freeze all layers except the last FC layer and  
just train a new FC with less number of classes.  
  
\*\*Describe the solution you\'d like\*\* a simple way to:  
  
1. pass in the model name/ checkpoint.  
2. Specify the FC layer name to keep training while freezing the rest  
 of the layers  
  
\*\*Describe alternatives you\'ve considered\*\* I wrote function below to  
copy weights and freeze layers for `segresnet`. it keeps the last layer  
named `conv\_0.conv\_0`. I load this model in my init of my app. However,  
training doesn\'t converge so I think something is missing  
  
 def pruneModelFCLayer(dst\_model, src\_model, checkptPath):  
 checkpoint = torch.load(checkptPath)  
 src\_model\_state\_dict = checkpoint.get("model", checkpoint)  
 src\_model.load\_state\_dict(src\_model\_state\_dict , strict=False)  
  
 new\_model\_state\_dic, updated\_keys , unchanged\_keys = copy\_model\_state( dst\_model , src\_model  
 , exclude\_vars="conv\_0.conv\_0", inplace=False)  
 print(f"unchanged keys {unchanged\_keys}")  
 src\_model\_state\_dict['conv\_final.2.conv.weight'][j, ...]  
  
 dst\_model.load\_state\_dict(new\_model\_state\_dic) # , strict=load\_strict)  
  
 ### stop gradients for the pretrained weights  
 for x in dst\_model.named\_parameters():  
 if x[0] in updated\_keys:  
 x[1].requires\_grad = False  
  
 params = generate\_param\_groups(network=dst\_model,layer\_matches=[lambda x: x[0] in updated\_keys],  
 match\_types=["filter"],lr\_values=[1e-4],include\_others=False)  
  
 return dst\_model ,params

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Create new datastore "localJson" to read from json file the location of images and labels

**Issue No. 1292 opened by AHarouni on 2023-02-10 at 19:33:52 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* Monai  
core can easily train using a json config with images and labels. Monai  
label on the other hand need to move files in a certain structure. It is  
very hard even impossible to move data on servers managed by admins.  
  
\*\*Describe the solution you\'d like\*\* It would be great to be able to  
load from a json which points to images and labels. Moreover you should  
be able to give it a directory to create the normal datastoer sturcture  
for any labels to be added to unlabeled images, allow to add images to  
this directory. In short keep all existing functionality but be able to  
load from json as a start. This will allow to easily modify the json  
(add or remove images) so user can experiment with different data sizes  
  
\*\*Describe alternatives you\'ve considered\*\* I have an initial solution  
here. I copied the local.py / inherited from it so it would be easier to  
diff with local. I broke some software engineering principals by  
overwriting some private functions. I am sure you can come up with an  
easier way to do it or restructure the code to a parent class and 2  
children.  
  
to get this to work, I start the server with flags below to point to the  
`dataset.json` and the data root  
  
 --studies /data/Challenges/Totalsegmentator\_dataset/monailabel \  
 --conf dataJsonPath /data/Challenges/Totalsegmentator\_dataset/monailabel2/dataset.json \  
 --conf dataJsonRoot /data/Challenges/Totalsegmentator\_dataset/monailabel/  
  
then a minor change to `radiology/main.py` to check on these flags to  
load the localjson datastore instead of the local datastore  
  
 def init\_datastore(self) -> Datastore:  
 dataJsonPath = self.conf.get("dataJsonPath", "")  
 if dataJsonPath:  
 from monailabel.datastore.localJson import LocalJsonDatastore  
 from monailabel.config import settings  
 dataJsonRoot = self.conf.get("dataJsonRoot", "")  
 if not dataJsonRoot:  
 dataJsonRoot = os.path.dirname(dataJsonPath)  
 print(f" ------------------------------ got dataJsonPath as {dataJsonPath}")  
 datastore = LocalJsonDatastore(  
 self.studies,  
 extensions=settings.MONAI\_LABEL\_DATASTORE\_FILE\_EXT,  
 auto\_reload=settings.MONAI\_LABEL\_DATASTORE\_AUTO\_RELOAD,  
 dataJsonRoot=dataJsonRoot, dataJsonPath=dataJsonPath  
 )  
  
 # datastore = self.init\_datastore\_json(dataJsonPath)  
 else:  
 datastore = super().init\_datastore()  
  
 if self.heuristic\_planner:  
 self.planner.run(datastore)  
 return datastore  
  
below is my initial implementation to `LocalJsonDatastore`  
  
 # Copyright (c) MONAI Consortium  
 # Licensed under the Apache License, Version 2.0 (the "License");  
 # you may not use this file except in compliance with the License.  
 # You may obtain a copy of the License at  
 # http://www.apache.org/licenses/LICENSE-2.0  
 # Unless required by applicable law or agreed to in writing, software  
 # distributed under the License is distributed on an "AS IS" BASIS,  
 # WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.  
 # See the License for the specific language governing permissions and  
 # limitations under the License.  
  
 import copy  
 import fnmatch  
 import io  
 import json  
 import logging  
 import os  
 import pathlib  
 import shutil  
 import time  
 from typing import Any, Dict, List, Tuple  
  
 from filelock import FileLock  
 from pydantic import BaseModel  
 from watchdog.events import PatternMatchingEventHandler  
 from watchdog.observers import Observer  
  
 from monailabel.interfaces.datastore import Datastore, DefaultLabelTag  
 from monailabel.interfaces.exception import ImageNotFoundException, LabelNotFoundException  
 from monailabel.utils.others.generic import file\_checksum, file\_ext, remove\_file  
 from monai.data import load\_decathlon\_properties, load\_decathlon\_datalist  
  
 logger = logging.getLogger(\_\_name\_\_)  
  
  
 class DataModel2(BaseModel):  
 ext: str = ""  
 path: str = ""  
 info: Dict[str, Any] = {}  
  
  
 class ImageLabelModel2(BaseModel):  
 image: DataModel2  
 labels: Dict[str, DataModel2] = {} # tag => label  
  
 def tags(self):  
 return self.labels.keys()  
 def label\_path(self,tag): # aeh added  
 # return path from json if tag == final  
 if tag==DefaultLabelTag.FINAL:  
 lbPath = self.labels[tag].path  
 elif tag==DefaultLabelTag.ORIGINAL:  
 lbPath = self.labels[tag].path  
 return lbPath  
  
  
 class LocalDatastoreModel2(BaseModel):  
 name: str  
 description: str  
 images\_dir: str = ""  
 labels\_dir: str = "labels"  
 objects: Dict[str, ImageLabelModel2] = {}  
  
 # will be ignored while saving...  
 base\_path: str = ""  
  
 def tags(self):  
 tags = set()  
 for v in self.objects.values():  
 tags.update(v.tags())  
 return tags  
  
 def filter\_by\_tag(self, tag: str):  
 return {k: v for k, v in self.objects.items() if v.labels.get(tag)}  
  
 def label(self, id: str, tag: str):  
 obj = self.objects.get(id)  
 return obj.labels.get(tag) if obj else None  
  
 def image\_path(self):  
 return os.path.join(self.base\_path, self.images\_dir) if self.base\_path else self.images\_dir  
  
 def label\_path(self, tag):  
 path = os.path.join(self.labels\_dir, tag) if tag else self.labels\_dir  
 return os.path.join(self.base\_path, path) if self.base\_path else path  
  
 def labels\_path(self):  
 path = self.labels\_dir  
 return {tag: os.path.join(path, tag) if self.base\_path else path for tag in self.tags()}  
  
 from monailabel.datastore.local import LocalDatastore  
 class LocalJsonDatastore(LocalDatastore): #Datastore LocalDatastore  
 """  
 Class to represent a datastore local to the MONAI-Label Server  
  
 Attributes  
 ----------  
 `name: str`  
 The name of the datastore  
  
 `description: str`  
 The description of the datastore  
 """  
  
 def \_\_init\_\_(  
 self,  
 datastore\_path: str,  
 dataJsonRoot, dataJsonPath, dataJsonSection: str = "training",  
 images\_dir: str = ".",  
 labels\_dir: str = "labels",  
 datastore\_config: str = "datastore\_v2.json",  
 extensions=("\*.nii.gz", "\*.nii"),  
 auto\_reload=False,  
 ):  
 """  
 Creates a `LocalDataset` object  
  
 Parameters:  
  
 `datastore\_path: str`  
 a string to the directory tree of the desired dataset  
  
 `datastore\_config: str`  
 optional file name of the dataset configuration file (by default `dataset.json`)  
 """  
  
 #self.\_dataJsonPath = "/data/Challenges/medicaldecathlon/rawUnzipped/Task09\_Spleen/dataset\_0.json"  
 #self.\_dataRoot = "/data/Challenges/medicaldecathlon/rawUnzipped/Task09\_Spleen/"  
 # self.\_dataJsonSection = "training"  
 #self.\_dataJsonSection = "test"  
  
 self.\_dataJsonPath = dataJsonPath  
 self.\_dataRoot =dataJsonRoot  
 self.\_dataJsonSection = dataJsonSection  
  
  
 self.\_datastore\_path = datastore\_path  
 print(f" --- {datastore\_path=}")  
  
  
 self.\_datastore\_config\_path = os.path.join(datastore\_path, datastore\_config)  
 self.\_extensions = [extensions] if isinstance(extensions, str) else extensions  
 self.\_ignore\_event\_count = 0  
 self.\_ignore\_event\_config = False  
 self.\_config\_ts = 0  
 self.\_auto\_reload = auto\_reload  
  
 logging.getLogger("filelock").setLevel(logging.ERROR)  
  
 logger.info(f"Auto Reload: {auto\_reload}; Extensions: {self.\_extensions}")  
  
 os.makedirs(self.\_datastore\_path, exist\_ok=True)  
  
 self.\_lock\_file = os.path.join(datastore\_path, ".lock")  
 self.\_datastore: LocalDatastoreModel2 = LocalDatastoreModel2(  
 name="new-dataset", description="New Dataset", images\_dir=images\_dir, labels\_dir=labels\_dir  
 )  
 self.\_datastore.base\_path = self.\_datastore\_path  
 self.\_init\_from\_datastore\_file(throw\_exception=True)  
  
 os.makedirs(self.\_datastore.image\_path(), exist\_ok=True)  
 os.makedirs(self.\_datastore.label\_path(None), exist\_ok=True)  
 os.makedirs(self.\_datastore.label\_path(DefaultLabelTag.FINAL), exist\_ok=True)  
 os.makedirs(self.\_datastore.label\_path(DefaultLabelTag.ORIGINAL), exist\_ok=True)  
  
 # reconcile the loaded datastore file with any existing files in the path  
 self.\_reconcile\_datastore()  
  
 if auto\_reload:  
 logger.info("Start observing external modifications on datastore (AUTO RELOAD)")  
 # Image Dir  
 include\_patterns = [f"{self.\_datastore.image\_path()}{os.path.sep}{ext}" for ext in [\*extensions]]  
  
 # Label Dir(s)  
 label\_dirs = self.\_datastore.labels\_path()  
 label\_dirs[DefaultLabelTag.FINAL] = self.\_datastore.label\_path(DefaultLabelTag.FINAL)  
 label\_dirs[DefaultLabelTag.ORIGINAL] = self.\_datastore.label\_path(DefaultLabelTag.ORIGINAL)  
 for label\_dir in label\_dirs.values():  
 include\_patterns.extend(f"{label\_dir}{os.path.sep}{ext}" for ext in [\*extensions])  
  
 # Config  
 include\_patterns.append(self.\_datastore\_config\_path)  
  
 self.\_handler = PatternMatchingEventHandler(patterns=include\_patterns)  
 self.\_handler.on\_created = self.\_on\_any\_event  
 self.\_handler.on\_deleted = self.\_on\_any\_event  
 self.\_handler.on\_modified = self.\_on\_modify\_event  
  
 try:  
 self.\_ignore\_event\_count = 0  
 self.\_ignore\_event\_config = False  
 self.\_observer = Observer()  
 self.\_observer.schedule(self.\_handler, recursive=True, path=self.\_datastore\_path)  
 self.\_observer.start()  
 except OSError as e:  
 logger.error(  
 "Failed to start File watcher. "  
 "Local datastore will not update if images and labels are moved from datastore location."  
 )  
 logger.error(str(e))  
  
 # def name(self) -> str:  
 # """  
 # Dataset name (if one is assigned)  
 #  
 # Returns:  
 # name (str): Dataset name as string  
 # """  
 # return self.\_datastore.name  
 #  
 # def set\_name(self, name: str):  
 # """  
 # Sets the dataset name in a standardized format (lowercase, no spaces).  
 #  
 # Parameters:  
 # name (str): Desired dataset name  
 # """  
 # self.\_datastore.name = name  
 # self.\_update\_datastore\_file()  
 #  
 # def description(self) -> str:  
 # """  
 # Gets the description field for the dataset  
 #  
 # :return description: str  
 # """  
 # return self.\_datastore.description  
 #  
 # def set\_description(self, description: str):  
 # """  
 # Set a description for the dataset  
 #  
 # :param description: str  
 # """  
 # self.\_datastore.description = description  
 # self.\_update\_datastore\_file()  
 #  
 # def \_to\_id(self, file: str) -> Tuple[str, str]:  
 # ext = file\_ext(file)  
 # extensions = [e.replace("\*", "") for e in self.\_extensions]  
 # for e in extensions:  
 # if file.endswith(e):  
 # ext = e  
 # id = file.replace(ext, "")  
 # return id, ext  
 #  
 # def \_filename(self, id: str, ext: str) -> str:  
 # return id + ext  
 #  
 # def \_to\_bytes(self, file):  
 # return io.BytesIO(pathlib.Path(file).read\_bytes())  
 #  
 def datalist(self, full\_path=True) -> List[Dict[str, Any]]:  
 """  
 Return a dictionary of image and label pairs corresponding to the 'image' and 'label'  
 keys respectively  
  
 :return: the {'label': image, 'label': label} pairs for training  
 """  
  
 tag = DefaultLabelTag.FINAL  
 image\_path = self.\_datastore.image\_path()  
 label\_path = self.\_datastore.label\_path(tag)  
  
 ds = []  
 for k, v in self.\_datastore.filter\_by\_tag(tag).items():  
 ds.append(  
 {  
 # "image": os.path.realpath(os.path.join(image\_path, self.\_filename(k, v.image.ext))),  
 "image": os.path.realpath(v.image.path),  
 # "label": os.path.realpath(os.path.join(label\_path, self.\_filename(k, v.labels[tag].ext))),  
 "label": os.path.realpath(v.labels[tag].path),  
 "meta": {"image": v.image.info, "label": v.labels[tag].info},  
 }  
 )  
  
 if not full\_path:  
 ds = json.loads(json.dumps(ds).replace(f"{self.\_datastore\_path.rstrip(os.pathsep)}{os.pathsep}", ""))  
 return ds  
 #  
 # def get\_image(self, image\_id: str) -> Any:  
 # """  
 # Retrieve image object based on image id  
 #  
 # :param image\_id: the desired image's id  
 # :return: return the "image"  
 # """  
 # uri = self.get\_image\_uri(image\_id)  
 # return self.\_to\_bytes(uri) if uri else None  
  
 def get\_image\_uri(self, image\_id: str) -> str:  
 """  
 Retrieve image uri based on image id  
  
 :param image\_id: the desired image's id  
 :return: return the image uri  
 """  
 obj = self.\_datastore.objects.get(image\_id)  
 return str(os.path.realpath(obj.image.path)) if obj else ""  
  
 def get\_image\_info(self, image\_id: str) -> Dict[str, Any]:  
 """  
 Get the image information for the given image id  
  
 :param image\_id: the desired image id  
 :return: image info as a list of dictionaries Dict[str, Any]  
 """  
 obj = self.\_datastore.objects.get(image\_id)  
 info = copy.deepcopy(obj.image.info) if obj else {}  
 if obj:  
 name = self.\_filename(image\_id, obj.image.ext)  
 # path = os.path.realpath(os.path.join(self.\_datastore.image\_path(), name))  
 path = os.path.realpath(obj.image.path)  
 info["path"] = path  
 logger.info(f"----------------------------aeh org label --- in get\_image\_info ------ {info}")  
 return info  
 ############ TODO Need to add this code / function on load from json file  
 # def \_\_call\_\_(self, request, datastore: Datastore):  
 # loader = LoadImage(image\_only=True)  
 # result = {}  
 # for image\_id in datastore.list\_images():  
 # for tag in self.tags:  
 # label\_id: str = datastore.get\_label\_by\_image\_id(image\_id, tag)  
 # if label\_id:  
 # label = loader(datastore.get\_label\_uri(label\_id, tag))  
 # if isinstance(label, torch.Tensor):  
 # label = label.numpy()  
 # slices = [sid for sid in range(label.shape[0]) if np.sum(label[sid] > 0)]  
 # info = {"sum": int(np.sum(label)), "slices": len(slices)}  
 # logger.debug(f"{label\_id} => {info}")  
 #  
 # datastore.update\_label\_info(label\_id, tag, info)  
 # result[label\_id] = info  
 # return result  
 #  
 def get\_label(self, label\_id: str, label\_tag: str) -> Any:  
 """  
 Retrieve image object based on label id  
  
 :param label\_id: the desired label's id  
 :param label\_tag: the matching label's tag  
 :return: return the "label"  
 """  
 uri = self.get\_label\_uri(label\_id, label\_tag)  
 return self.\_to\_bytes(uri) if uri else None  
  
 # def get\_label\_uri(self, label\_id: str, label\_tag: str) -> str:  
 # """  
 # Retrieve label uri based on image id  
 #  
 # :param label\_id: the desired label's id  
 # :param label\_tag: the matching label's tag  
 # :return: return the label uri  
 # """  
 # label = self.\_datastore.label(label\_id, label\_tag)  
 # name = self.\_filename(label\_id, label.ext) if label else ""  
 # return str(os.path.realpath(os.path.join(self.\_datastore.label\_path(label\_tag), name))) if label else ""  
 #  
 # def get\_labels\_by\_image\_id(self, image\_id: str) -> Dict[str, str]:  
 # """  
 # Retrieve all label ids for the given image id  
 #  
 # :param image\_id: the desired image's id  
 # :return: label ids mapped to the appropriate `LabelTag` as Dict[LabelTag, str]  
 # """  
 # obj = self.\_datastore.objects.get(image\_id)  
 # return {tag: image\_id for tag in obj.labels} if obj else {}  
 #  
 # def get\_label\_by\_image\_id(self, image\_id: str, tag: str) -> str:  
 # """  
 # Retrieve label id for the given image id and tag  
 #  
 # :param image\_id: the desired image's id  
 # :param tag: matching tag name  
 # :return: label id  
 # """  
 # return self.get\_labels\_by\_image\_id(image\_id).get(tag, "")  
 #  
 # def get\_label\_info(self, label\_id: str, label\_tag: str) -> Dict[str, Any]:  
 # """  
 # Get the label information for the given label id  
 #  
 # :param label\_id: the desired label id  
 # :param label\_tag: the matching label tag  
 # :return: label info as a list of dictionaries Dict[str, Any]  
 # """  
 # label = self.\_datastore.label(label\_id, label\_tag)  
 # info: Dict[str, Any] = label.info if label else {}  
 # return info  
 #  
 # def get\_labeled\_images(self) -> List[str]:  
 # """  
 # Get all images that have a corresponding label  
 #  
 # :return: list of image ids List[str]  
 # """  
 # return [k for k, v in self.\_datastore.objects.items() if v.labels.get(DefaultLabelTag.FINAL)]  
 #  
 # def get\_unlabeled\_images(self) -> List[str]:  
 # """  
 # Get all images that have no corresponding label  
 #  
 # :return: list of image ids List[str]  
 # """  
 # return [k for k, v in self.\_datastore.objects.items() if not v.labels.get(DefaultLabelTag.FINAL)]  
 #  
 # def list\_images(self) -> List[str]:  
 # """  
 # Return list of image ids available in the datastore  
 #  
 # :return: list of image ids List[str]  
 # """  
 # return list(self.\_datastore.objects.keys())  
 #  
 # def \_on\_any\_event(self, event):  
 # if self.\_ignore\_event\_count:  
 # logger.debug(f"Ignoring event by count: {self.\_ignore\_event\_count} => {event}")  
 # self.\_ignore\_event\_count = max(self.\_ignore\_event\_count - 1, 0)  
 # return  
 #  
 # logger.debug(f"Event: {event}")  
 # self.refresh()  
 #  
 # def \_on\_modify\_event(self, event):  
 # # handle modify events only for config path; rest ignored  
 # if event.src\_path != self.\_datastore\_config\_path:  
 # return  
 #  
 # if self.\_ignore\_event\_config:  
 # self.\_ignore\_event\_config = False  
 # return  
 #  
 # self.\_init\_from\_datastore\_file()  
 #  
 # def refresh(self):  
 # """  
 # Refresh the datastore based on the state of the files on disk  
 # """  
 # print(f"----- in refresh")  
 # self.\_reconcile\_datastore()  
 #  
 def add\_image(self, image\_id: str, image\_filename: str, image\_info: Dict[str, Any]) -> str:  
 raise NotImplemented  
 # id, image\_ext = self.\_to\_id(os.path.basename(image\_filename))  
 # if not image\_id:  
 # image\_id = id  
 #  
 # logger.info(f"Adding Image: {image\_id} => {image\_filename}")  
 # name = self.\_filename(image\_id, image\_ext)  
 # dest = os.path.realpath(os.path.join(self.\_datastore.image\_path(), name))  
 #  
 # with FileLock(self.\_lock\_file):  
 # logger.debug("Acquired the lock!")  
 # shutil.copy(image\_filename, dest)  
 #  
 # image\_info = image\_info if image\_info else {}  
 # image\_info["ts"] = int(time.time())  
 # # image\_info["checksum"] = file\_checksum(dest)  
 # image\_info["name"] = name  
 #  
 # self.\_datastore.objects[image\_id] = ImageLabelModel2(image=DataModel2(info=image\_info, ext=image\_ext,path=dest))  
 # self.\_update\_datastore\_file(lock=False)  
 # logger.debug("Released the lock!")  
 # return image\_id  
 #  
 # def remove\_image(self, image\_id: str) -> None:  
 # logger.info(f"Removing Image: {image\_id}")  
 #  
 # obj = self.\_datastore.objects.get(image\_id)  
 # if not obj:  
 # raise ImageNotFoundException(f"Image {image\_id} not found")  
 #  
 # # Remove all labels  
 # tags = list(obj.labels.keys())  
 # for tag in tags:  
 # self.remove\_label(image\_id, tag)  
 #  
 # # Remove Image  
 # name = self.\_filename(image\_id, obj.image.ext)  
 # remove\_file(os.path.realpath(os.path.join(self.\_datastore.image\_path(), name)))  
 #  
 # if not self.\_auto\_reload:  
 # self.refresh()  
 #  
 def save\_label(self, image\_id: str, label\_filename: str, label\_tag: str, label\_info: Dict[str, Any]) -> str:  
 """  
 Save a label for the given image id and return the newly saved label's id  
  
 :param image\_id: the image id for the label  
 :param label\_filename: the path to the label file  
 :param label\_tag: the tag for the label  
 :param label\_info: additional info for the label  
 :return: the label id for the given label filename  
 """  
 logger.info(f"Saving Label for Image: {image\_id}; Tag: {label\_tag}; Info: {label\_info}")  
 obj = self.\_datastore.objects.get(image\_id)  
 if not obj:  
 raise ImageNotFoundException(f"Image {image\_id} not found")  
  
 \_, label\_ext = self.\_to\_id(os.path.basename(label\_filename))  
 label\_id = image\_id  
  
 logger.info(f"Adding Label: {image\_id} => {label\_tag} => {label\_filename}")  
 label\_path = self.\_datastore.label\_path(label\_tag)  
 name = self.\_filename(image\_id, label\_ext)  
 dest = os.path.join(label\_path, name)  
  
 with FileLock(self.\_lock\_file):  
 logger.debug("Acquired the lock!")  
 os.makedirs(label\_path, exist\_ok=True)  
 shutil.copy(label\_filename, dest)  
  
 label\_info = label\_info if label\_info else {}  
 label\_info["ts"] = int(time.time())  
 label\_info["checksum"] = file\_checksum(dest)  
 label\_info["name"] = name  
  
 obj.labels[label\_tag] = DataModel2(info=label\_info, ext=label\_ext, path=dest)  
 logger.info(f"Label Info: {label\_info}")  
 self.\_update\_datastore\_file(lock=False)  
 logger.debug("Release the lock!")  
 return label\_id  
  
 # def remove\_label(self, label\_id: str, label\_tag: str) -> None:  
 # logger.info(f"Removing label: {label\_id} => {label\_tag}")  
 # remove\_file(self.get\_label\_uri(label\_id, label\_tag))  
 #  
 # if not self.\_auto\_reload:  
 # self.refresh()  
 #  
 # def update\_image\_info(self, image\_id: str, info: Dict[str, Any]) -> None:  
 # """  
 # Update (or create a new) info tag for the desired image  
 #  
 # :param image\_id: the id of the image we want to add/update info  
 # :param info: a dictionary of custom image information Dict[str, Any]  
 # """  
 # obj = self.\_datastore.objects.get(image\_id)  
 # if not obj:  
 # raise ImageNotFoundException(f"Image {image\_id} not found")  
 #  
 # obj.image.info.update(info)  
 # self.\_update\_datastore\_file()  
 #  
 # def update\_label\_info(self, label\_id: str, label\_tag: str, info: Dict[str, Any]) -> None:  
 # """  
 # Update (or create a new) info tag for the desired label  
 #  
 # :param label\_id: the id of the label we want to add/update info  
 # :param label\_tag: the matching label tag  
 # :param info: a dictionary of custom label information Dict[str, Any]  
 # """  
 # label = self.\_datastore.label(label\_id, label\_tag)  
 # if not label:  
 # raise LabelNotFoundException(f"Label: {label\_id} Tag: {label\_tag} not found")  
 #  
 # label.info.update(info)  
 # self.\_update\_datastore\_file()  
 #  
 def \_list\_files(self, path, patterns):  
 ## don't want to call this should call \_list\_json\_files instead  
 raise NotImplemented  
 # files = os.listdir(path)  
 #  
 # filtered = dict()  
 # for pattern in patterns:  
 # matching = fnmatch.filter(files, pattern)  
 # for file in matching:  
 # filtered[os.path.basename(file)] = file  
 # return filtered  
  
 def \_list\_json\_files(self, jsonKey, LabelTag=""):  
 logger.info(f" ------------------------ in \_list\_json\_files {jsonKey=}")  
  
 filtered = dict()  
 logger.info(f"------------loading {self.\_dataJsonPath=} -------- {self.\_dataJsonSection=} ------- {self.\_dataRoot=}")  
 dsLst =load\_decathlon\_datalist(self.\_dataJsonPath , True, self.\_dataJsonSection,base\_dir= self.\_dataRoot )  
 for imgNLabel in dsLst:  
 if jsonKey in imgNLabel:  
 imgPath=imgNLabel[jsonKey]  
 # labelPath=imgNLabel["label"]  
 filtered[os.path.basename(imgPath)] = imgPath  
 else:  
 print(f"-->>>>>>>>>>>>>>>>>>>>> tag {jsonKey} is not is json dict")  
  
 return filtered  
  
 # def \_reconcile\_datastore(self):  
 # logger.debug("reconcile datastore...")  
 # invalidate = 0  
 # invalidate += self.\_remove\_non\_existing()  
 # invalidate += self.\_add\_non\_existing\_images()  
 #  
 # labels\_dir = self.\_datastore.label\_path(None)  
 # logger.debug(f"Labels Dir {labels\_dir}")  
 #  
 # tags = [f for f in os.listdir(labels\_dir) if os.path.isdir(os.path.join(labels\_dir, f))]  
 # logger.debug(f"Label Tags: {tags}")  
 # for tag in tags:  
 # invalidate += self.\_add\_non\_existing\_labels(tag)  
 #  
 # invalidate += self.\_remove\_non\_existing()  
 #  
 # logger.info(f"Invalidate count: {invalidate}")  
 # if invalidate:  
 # logger.debug("Save datastore file to disk")  
 # self.\_update\_datastore\_file()  
 # else:  
 # logger.debug("No changes needed to flush to disk")  
 #  
 def \_add\_non\_existing\_images(self) -> int:  
 invalidate = 0  
 self.\_init\_from\_datastore\_file()  
  
 # local\_images = self.\_list\_files(self.\_datastore.image\_path(), self.\_extensions,jsonKey="image")  
 local\_images = self.\_list\_json\_files(jsonKey="image")  
  
 image\_ids = list(self.\_datastore.objects.keys())  
 # for image\_file in local\_images:  
 for image\_file, image\_path in local\_images.items():  
 image\_id, image\_ext = self.\_to\_id(image\_file)  
 if image\_id not in image\_ids:  
 logger.info(f"Adding New Image: {image\_id} => {image\_file} ==>{image\_path}")  
  
 name = self.\_filename(image\_id, image\_ext)  
 image\_info = {  
 "ts": int(time.time()),  
 # "checksum": file\_checksum(os.path.join(self.\_datastore.image\_path(), name)),  
 "name": name,  
 }  
  
 invalidate += 1  
 self.\_datastore.objects[image\_id] = ImageLabelModel2(image=DataModel2(info=image\_info, ext=image\_ext,path=image\_path))  
  
 return invalidate  
  
 def \_add\_non\_existing\_labels(self, tag) -> int:  
 invalidate = 0  
 self.\_init\_from\_datastore\_file()  
  
 if tag == DefaultLabelTag.ORIGINAL:  
 ### TODO AEH hack can do 1 or the other  
 #### either main behavoure: orignal loads the inference  
 #### or local the orignal from the json file  
 local\_labels = self.\_list\_json\_files(jsonKey="orignal", LabelTag=tag)  
  
 logger.info(f"--------------- aeh\_org {local\_labels=}")  
 # local\_labels = self.\_list\_files(self.\_datastore.label\_path(tag), self.\_extensions)  
 else:  
 local\_labels = self.\_list\_json\_files(jsonKey="label", LabelTag=tag)  
  
 image\_ids = list(self.\_datastore.objects.keys())  
 # for label\_file in local\_labels:  
 for label\_file, label\_path in local\_labels.items():  
 label\_id, label\_ext = self.\_to\_id(label\_file)  
  
 obj = self.\_datastore.objects.get(label\_id)  
 if not obj or label\_id not in image\_ids:  
 logger.warning(f"IGNORE:: No matching image exists for '{label\_id}' to add [{label\_file}]")  
 continue  
  
 if not obj.labels.get(tag):  
 logger.info(f"Adding New Label: {tag} => {label\_id} => {label\_file}")  
  
 name = self.\_filename(label\_id, label\_ext)  
 label\_info = {  
 "ts": int(time.time()),  
 # "checksum": file\_checksum(os.path.join(self.\_datastore.label\_path(tag), name)),  
 "checksum": file\_checksum(label\_path),  
 "name": name,  
 }  
  
 self.\_datastore.objects[label\_id].labels[tag] = DataModel2(info=label\_info, ext=label\_ext,path=label\_path)  
 invalidate += 1  
  
 return invalidate  
  
 def \_remove\_non\_existing(self) -> int:  
 invalidate = 0  
 self.\_init\_from\_datastore\_file()  
  
 objects: Dict[str, ImageLabelModel2] = {}  
 for image\_id, obj in self.\_datastore.objects.items():  
 name = self.\_filename(image\_id, obj.image.ext)  
 # print(f" aeh {image\_id=} , {type(obj)=} details {obj=}")  
 # if not os.path.exists(os.path.realpath(os.path.join(self.\_datastore.image\_path(), name))):  
 if not os.path.exists(os.path.realpath(obj.image.path)):  
 logger.info(f"Removing non existing Image Id: {image\_id} with path {obj.image.path}")  
 invalidate += 1  
 else:  
 labels: Dict[str, DataModel2] = {}  
 for tag, label in obj.labels.items():  
 name = self.\_filename(image\_id, label.ext)  
 if tag == DefaultLabelTag.ORIGINAL:  
 lbPath=os.path.join(self.\_datastore.label\_path(tag), name)  
 logger.info(f"-----------aeh\_org {lbPath=}")  
 else:  
 lbPath= label.path  
 if not os.path.exists(os.path.realpath(lbPath)):  
 logger.info(f"Removing non existing Label Id: '{image\_id}' for '{tag}'")  
 invalidate += 1  
 else:  
 labels[tag] = label  
 obj.labels.clear()  
 obj.labels.update(labels)  
 objects[image\_id] = obj  
  
 self.\_datastore.objects.clear()  
 self.\_datastore.objects.update(objects)  
 return invalidate  
  
 def \_init\_from\_datastore\_file(self, throw\_exception=False):  
 try:  
 with FileLock(self.\_lock\_file):  
 logger.debug("Acquired the lock!")  
 if os.path.exists(self.\_datastore\_config\_path):  
 ts = os.stat(self.\_datastore\_config\_path).st\_mtime  
 if self.\_config\_ts != ts:  
 logger.debug(f"Reload Datastore; old ts: {self.\_config\_ts}; new ts: {ts}")  
 self.\_datastore = LocalDatastoreModel2.parse\_file(self.\_datastore\_config\_path)  
 self.\_datastore.base\_path = self.\_datastore\_path  
 self.\_config\_ts = ts  
 logger.debug("Release the Lock...")  
 except ValueError as e:  
 logger.error(f"+++ Failed to load datastore => {e}")  
 if throw\_exception:  
 raise e  
  
 # def \_update\_datastore\_file(self, lock=True):  
 # def \_write\_to\_file():  
 # logger.debug("+++ Datastore is updated...")  
 # self.\_ignore\_event\_config = True  
 # with open(self.\_datastore\_config\_path, "w") as f:  
 # f.write(json.dumps(self.\_datastore.dict(exclude={"base\_path"}), indent=2, default=str))  
 # self.\_config\_ts = os.stat(self.\_datastore\_config\_path).st\_mtime  
 #  
 # if lock:  
 # with FileLock(self.\_lock\_file):  
 # logger.debug("Acquired the Lock...")  
 # \_write\_to\_file()  
 # logger.debug("Released the Lock...")  
 # else:  
 # \_write\_to\_file()  
 #  
 # def status(self) -> Dict[str, Any]:  
 # tags: dict = {}  
 # for obj in self.\_datastore.objects.values():  
 # for tag, \_ in obj.labels.items():  
 # tags[tag] = tags.get(tag, 0) + 1  
 #  
 # return {  
 # "total": len(self.list\_images()),  
 # "completed": len(self.get\_labeled\_images()),  
 # "label\_tags": tags,  
 # }  
 #  
 # def json(self):  
 # return self.\_datastore.dict(exclude={"base\_path"})

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Xnat datastore: Push back dicom seg to xnat

**Issue No. 1291 opened by AHarouni on 2023-02-10 at 18:41:01 Type: None**

\*\*Is your feature request related to a problem? Please describe.\*\* Xnat  
datastore is great. Works great. Missing feature is to implement the  
save method in the data store so that Admin can run batch inference on  
segmenation models to trigger inference, convert nifti to dicom seg then  
push it back to xnat  
  
\*\*Describe the solution you\'d like\*\* I have implemented these methods  
in the xnat.py by adding functions below  
  
 def \_\_convert\_Nifti\_2\_dcmSeg(self,series\_dir,niiSegPath,modelName,labelNames)->str:  
 label\_info =[]  
 for i,lb in enumerate(labelNames):  
 label\_info.append(  
 {"modelName": modelName, "name": str(i+1)+'\_'+lb , "description": "lb"+str(i+1)+'\_'+ lb}  
 )  
 dcmSegFile=nifti\_to\_dicom\_seg(series\_dir=series\_dir, label=niiSegPath,label\_info=label\_info)  
 logging.info(f" converted nifit to dicom seg --- at {dcmSegFile}")  
 return dcmSegFile  
 ##############################################################################  
 def \_\_getDcmSegUploadurl(self,image\_id,aiaaModelName):  
 project, subject, experiment, scan = self.\_id\_to\_fields(image\_id)  
 assessionID= experiment  
 aiaaDcmSegUploadurlRoot = "/xapi/roi/projects/" + project + '/sessions/' + assessionID + '/collections/'  
 url = "{}/xapi/roi/projects/{}/sessions/{}/collections/Pat{}\_S{}\_{}".format(  
 self.api\_url,  
 quote\_plus(project),  
 quote\_plus(assessionID),  
 quote\_plus(subject),  
 quote\_plus(scan),  
 quote\_plus(aiaaModelName),  
 )  
 return url  
 ##############################################################################  
 def save\_label(self, image\_id: str, label\_filename: str, label\_tag: str, label\_info: Dict[str, Any]) -> str:  
 aiaaModelName = label\_info.get('model', "NoModel") # waiting on how to get the model name  
 labelNames=label\_info.get('params',{}).get('label\_names',{})  
  
 #### convert nii to dcm seg and upload to Xnat  
 series\_dir=self.\_download\_image(image\_id)  
 tmpDcmSegpath = self.\_\_convert\_Nifti\_2\_dcmSeg(series\_dir,label\_filename, aiaaModelName, labelNames)  
 data = open(tmpDcmSegpath, "rb")  
 url=self.\_\_getDcmSegUploadurl(image\_id,aiaaModelName)  
 self.\_request\_put(url,data)  
 logger.info(f"------------- in xnat upload with {url=} and tmp file {tmpDcmSegpath=}")  
  
 return image\_id  
 ##############################################################################  
 def \_request\_put(self, url,data):  
 logger.info(f" in upload dcm seg {url}")  
 headers = {'Content-Type': "application/octet-stream"}  
 params = {"overwrite": "true", "type": "SEG"}  
 response = self.xnat\_session.put(url, params=params, data=data, headers =headers, allow\_redirects=True) # ,verify=False,  
 if response.status\_code != 200: # failed call  
 logger.error(f" post call error status\_code= {response.status\_code} text ={response.text}")  
 else:  
 logger.info(f" put completed {response.text}")  
  
 return response  
  
Also minor modification to the nifti\_to\_dicom\_seg function in  
monailabel/datastore/utils/convert.py  
  
 def nifti\_to\_dicom\_seg(series\_dir, label, label\_info, file\_ext="\*", use\_itk=True):  
 start = time.time()  
  
 label\_np, meta\_dict = LoadImage()(label)  
 unique\_labels = np.unique(label\_np.flatten()).astype(np.int)  
 unique\_labels = unique\_labels[unique\_labels != 0]  
  
 ### add model name and labels  
 info = label\_info[0] if label\_info and 0 < len(label\_info) else {}  
 modelName = info.get("modelName", "AIName")  
  
 .....  
  
 template = {  
 "ContentCreatorName": "Reader1",  
 "ClinicalTrialSeriesID": "Session1",  
 "ClinicalTrialTimePointID": "1",  
 "SeriesDescription": modelName , <<<---- add model name   
 "SeriesNumber": "300",  
 "InstanceNumber": "1",  
 "segmentAttributes": [segment\_attributes],  
 "ContentLabel": "SEGMENTATION",  
 "ContentDescription": "MONAI Label - Image segmentation",  
 "ClinicalTrialCoordinatingCenterName": "MONAI",  
 "BodyPartExamined": "",  
 }

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Plugin Quality (User Experience) for 3D Slicer, QuPath, OHIF

**Issue No. 1278 opened by SachidanandAlle on 2023-02-06 at 15:43:34 Type:**

Collect and evaluate user experience for all plugins

**Additional Information:**

* Milestones: Discoverability: (many features, not clear how to use), documentation; tutorials; demo
* Assignees: SachidanandAlle, pdogra89

No comments at the moment!

Restored transform doesn't restore orientation

**Issue No. 1266 opened by AHarouni on 2023-01-27 at 23:11:06 Type:**

\*\*Describe the bug\*\* I use Restored transform to invert the space  
resolution, however it fails to restore the orientation changed with any  
of the `Orientationd(keys="image", axcodes="RAS"),`  
  
\*\*To Reproduce\*\* add this transform to spleen model in the pretranforms  
`Orientationd(keys="image", axcodes="RAS"),`  
  
and in the post transform `Restored(keys="pred", ref\_image="image"),`  
  
you would see that the segmentation is wrong

**Additional Information:**

* Milestones: Bugs & Fixes & Misc for 0.7
* Assignees: None

No comments at the moment!

unable to use inverse\_transforms in infers

**Issue No. 1265 opened by AHarouni on 2023-01-27 at 23:07:03 Type:**

Can\'t add any code to inverse\_transforms gives error  
  
 class MySegmentation(BasicInferTask):   
 ....   
 def inverse\_transforms(self, data=None) -> Union[None, Sequence[Callable]]:  
 return [  
 RestoreOrientationd(keys="pred", ref\_image="image")  
 ]  
  
gives error  
  
 File "/MONAILabel/monailabel/tasks/infer/basic\_infer.py", line 306, in \_\_call\_\_  
 data = self.run\_invert\_transforms(data, pre\_transforms, self.inverse\_transforms(data))  
 File "/MONAILabel/monailabel/tasks/infer/basic\_infer.py", line 397, in run\_invert\_transforms  
 transforms = [pre\_transforms[n if isinstance(n, str) else n.\_\_name\_\_] for n in names]  
 File "/MONAILabel/monailabel/tasks/infer/basic\_infer.py", line 397, in <listcomp>  
 transforms = [pre\_transforms[n if isinstance(n, str) else n.\_\_name\_\_] for n in names]  
 AttributeError: 'Orientationd' object has no attribute '\_\_name\_\_'  
  
the only way I can get it to work is to create a fake class and add  
attribute \*\*name\*\* as  
  
 def inverse\_transforms(self, data=None) -> Union[None, Sequence[Callable]]:  
 class CodeHack:  
 pass  
 codehack = CodeHack() # Create an empty employee record  
 codehack.\_\_name\_\_ = 7  
  
 return [codehack]  
  
this would invert the transform number 7 without any errors.

**Additional Information:**

* Milestones: Bugs & Fixes & Misc for 0.7
* Assignees: None

No comments at the moment!

Evaluate options to minimize MONAILabel start on windows and ubuntu

**Issue No. 1263 opened by SachidanandAlle on 2023-01-24 at 00:07:43 Type:**

scripts, ui etc.. find and evaluate possible solutions in the industry

**Additional Information:**

* Milestones: UI Installer/Setup package for MONAI Label (windows, ubuntu) [P1]
* Assignees: None

No comments at the moment!

Showcase OHIF + MONAILabel with user auth support

**Issue No. 1261 opened by SachidanandAlle on 2023-01-24 at 00:04:34 Type:**

e2e integration with user support

**Additional Information:**

* Milestones: Client (3DSlicer, OHIF, QuPath etc..) user authenticated [P1]
* Assignees: None

No comments at the moment!

Pytorch, Ignite, Monai -> Monai Label Tutorial

**Issue No. 1195 opened by danieltudosiu on 2022-12-09 at 12:20:47 Type: og**

\*\*Is your feature request related to a problem? Please describe.\*\* Due  
to the complexities required by the creation of interfaces for the  
backend and frontend to interact the \"information\" flow within MONAI  
Label is heavily offuscated and hard to comprehend for development.  
  
\*\*Describe the solution you\'d like\*\* Create the following four  
animations/figures:  
  
- An UML-like figure of MONAI Label that is specific for the backend  
 to complement the existing REST API one.  
- Three transition animations that exemplify how you move from X to  
 MONAI Label, where X is:  
 - Pure Pytorch  
 - Ignite  
 - MONAI  
  
The last point is inspired by the [PyTorch Lighting  
animation](https://www.pytorchlightning.ai/#:~:text=optimizer.zero\_grad())  
which conveys a lot of information in a very succinct way.  
  
\*\*Additional context\*\* This came to be after I had explained MONAI Label  
to \@MichelaA and a while ago Ignite to \@diazandr3s. This would  
facilitate a faster and better understanding of MONAI Label and  
hopefully as a side-effect less issue.  
  
Furthermore, this falls in line with the style of tutorials that are  
available in MONAI Core and it would be highly useful due to the fact  
that MONAI Label borrows design from PyTorch Lighting ultimately merging  
two very different APIs together which leads to a steep learning curve.

**Additional Information:**

* Milestones: None
* Assignees: danieltudosiu

No comments at the moment!

0.5 threshold used in multi-label deepedit

**Issue No. 1181 opened by danieltudosiu on 2022-12-05 at 18:28:39 Type: og**

\*\*Describe the bug\*\* In the codebase  
([here](https://github.com/Project-MONAI/MONAI/blob/dev/monai/apps/deepedit/transforms.py#L456),  
[here](https://github.com/Project-MONAI/MONAI/blob/dev/monai/apps/deepedit/transforms.py#L540)  
and  
[here](https://github.com/Project-MONAI/MONAI/blob/dev/monai/apps/deepedit/transforms.py#L770))  
the threshold of 0.5 is used disregarding the number of labels.  
  
\*\*Expected behavior\*\* The threshold should be 1/(len(labels)-1). Such  
that for a 3-class problem (besides background) the threshold is 0.33.  
  
At the moment we are asking the models to be extra sure of the labels  
before considering them right. This might be useful but is not  
explicitly implemented and is worse and worse the more classes there  
are.

**Additional Information:**

* Milestones: None
* Assignees: danieltudosiu

No comments at the moment!

Decouple Image Signal and Guidance Signal

**Issue No. 1173 opened by danieltudosiu on 2022-12-04 at 19:11:11 Type: og**

\*\*Is your feature request related to a problem? Please describe.\*\* At  
the moment there is no straightforward way to decouple the Image Signal  
and Guidance Signal from the DeepEdit (and I assume the rest of the  
library).  
  
The problem stems from the usage of the base SupervisedTrainer and  
bypassing its limitations via a new iteration method that internally  
calls the default iteration function.  
  
\*\*Describe the solution you\'d like\*\* I think DeepEdit should  
incorporate the  
[Interaction](https://github.com/Project-MONAI/MONAILabel/blob/f44be201f9800a6db279a3f26684573db3827ee0/monailabel/deepedit/interaction.py#L23)  
logic into a new Trainer that allows the same flexibility with added  
ones such as specialised prepare\_batch for both the internal loop as the  
external loop.  
  
Furthermore, there is no default Key for the Signal similar to  
[GanKeys](https://github.com/Project-MONAI/MONAI/blob/4deb76571e5b702d134317359db57d902fcec6a4/monai/utils/enums.py#L311)  
or  
[AdversarialKeys](https://github.com/Project-MONAI/GenerativeModels/blob/main/generative/utils/enums.py#L25).  
  
Lastly, currently, DeepEdit development is hindered by using the default  
prepare\_batch that does not take into account the existence of Guidance  
Signal and its usage in specialised blocks such as:  
  
- [Bi-Directional Seed Attention  
 Module](https://ieeexplore.ieee.org/document/9178957).  
- [Hint Fusion Block/Interaction Guided Gating  
 Network](https://link.springer.com/chapter/10.1007/978-3-031-16437-8\_65)  
- [Inception-Atrous-Collation](https://ieeexplore.ieee.org/document/9892573)  
- Cross-Attention in Transformer based networks (Swin U-NETR, U-NETR)  
  
\*\*Additional context\*\* Currently trying to improve the click ingestion  
of MONAI Label together with Self-Supervised Learning.  
  
Edit: Added more examples of recent techniques requiring the decoupling  
of signals. Edit 2: Added details about the lack of keys.

**Additional Information:**

* Milestones: None
* Assignees: danieltudosiu

No comments at the moment!

RuntimeError when "brats\_mri\_segmentation\_v0.2.1" from monaibundle is used.

**Issue No. 1051 opened by PranayBolloju on 2022-10-07 at 12:52:05 Type: None**

\*\*Describe the bug\*\* MONAI Label server is giving the following error  
when \"brats\_mri\_segmentation\_v0.2.1\" is used for brain tumor  
segmentation.  
  
\*\*RuntimeError: Given groups=1, weight of size \[16, 4, 3, 3, 3\],  
expected input\[1, 240, 240, 240, 160\] to have 4 channels, but got 240  
channels instead\*\*  
  
\*\*To Reproduce\*\* Steps to reproduce the behavior:  
  
1. pip install monailabel  
2. monailabel apps \--download \--name monaibundle \--output apps  
3. monailabel datasets \--download \--name Task01\_BrainTumour \--output  
 datasets  
4. monailabel start\_server \--app apps/monaibundle \--studies  
 datasets/Task01\_BrainTumour/imagesTr \--conf models  
 brats\_mri\_segmentation\_v0.2.1  
5. Run the model in 3D slicer with any image from the dataset.  
  
\*\*Expected behavior\*\* Segmentation should be displayed in 3D slicer.  
  
\*\*Screenshots\*\*  
![image](https://user-images.githubusercontent.com/98535359/194555949-b930809d-3f43-4e08-a3c5-781fff39a85b.png)  
![image](https://user-images.githubusercontent.com/98535359/194556418-9365d26d-24e5-4e88-a8ae-cd9080261dc3.png)  
  
\*\*Environment\*\*  
  
Ensuring you use the relevant python executable, please paste the output  
of:  
  
 python -c 'import monai; monai.config.print\_debug\_info()'  
  
# ================================ Printing MONAI config\... {#-printing-monai-config}  
  
MONAI version: 1.0.0 Numpy version: 1.22.4 Pytorch version: 1.12.1+cpu  
MONAI flags: HAS\_EXT = False, USE\_COMPILED = False, USE\_META\_DICT =  
False MONAI rev id: 170093375ce29267e45681fcec09dfa856e1d7e7 MONAI  
\*\*file\*\*:  
C:\\Users\\Admin\\AppData\\Local\\Programs\\Python\\Python39\\lib\\site-packages\\monai\\_\*init\*\\_.py  
  
Optional dependencies: Pytorch Ignite version: 0.4.10 Nibabel version:  
4.0.2 scikit-image version: 0.19.3 Pillow version: 9.2.0 Tensorboard  
version: 2.10.0 gdown version: 4.5.1 TorchVision version: 0.13.1+cpu  
tqdm version: 4.64.0 lmdb version: 1.3.0 psutil version: 5.9.1 pandas  
version: 1.4.3 einops version: 0.4.1 transformers version: NOT INSTALLED  
or UNKNOWN VERSION. mlflow version: NOT INSTALLED or UNKNOWN VERSION.  
pynrrd version: 0.4.3

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Video Annotation with MONAI Label - CVAT support

**Issue No. 1002 opened by tangy5 on 2022-09-19 at 03:19:18 Type: og**

Current endoscopy app annotation works with frame, the scoring method  
recommends 2D images for task. This issue discusses the integration of  
video annotation with current tooltracking and deid models.  
  
Endoscopy APP Video Annotation logic:  
  
1. MONAI Label starts studies with endoscopy videos, avi, mp4 format.  
2. The MONAI Label server solves video frames as input and predict  
 scores for each frame.  
3. Video-based scorning module calculates the metric.  
4. MONAI Label server push suggested videos (avi, mp4) to CVAT based on  
 metric.  
5. Users annotate video frames on CVAT. Mark complete until all frames  
 in video are annotated.  
6. MONAI Label server periodically query if there are finished tasks.  
7. MONAI Label server train/fine-tune model with new annotated video  
 frames.  
8. MONAI Label server predict scores for video frames, and loop  
 processing for active learning Step 2 -- Step 8  
  
Development steps:  
  
- ☒  
 1. Prepare endoscopy sample videos, and dataset.  
- ☒  
 2. Create video reader and image loader with videos for  
 tooltracking, deid function modules.  
- ☒  
 3. Create a scoring method based on videos, instead of 2D image.  
- ☐  
 4. Push/transfer suggested video from MONAI label to CVAT  
- ☒  
 5. MONAI Label query module on periodically check whether videos  
 are annotated.  
- ☒  
 6. Datastore communicating video and video labels between MONAI  
 Label and CVAT.  
- ☒  
 7. Check whether active learning loops compatible with videos.  
  
Video annotation and active learning workflow based on prior frame  
integration.  
![method](https://user-images.githubusercontent.com/58751975/190944420-4c20399f-d2c3-4eff-a876-4821166c5890.jpg)  
  
\@SachidanandAlle \@Nic-Ma Correct me if there are any thoughts. CVAT  
can only send frame requests to functions. We need to process the video  
loader at model data loader, and a video-based scoring method.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Develop/Integrate C++ Client library

**Issue No. 910 opened by SachidanandAlle on 2022-08-04 at 16:42:44 Type: og**

It could be from PAIR or can be new all together. Objective is to  
develop C++ client that can be used in tools like PAIR, MITKWorkbench,  
ITK Snap etc.. to invoke MONAILabel APIs

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Evaulate C++ Client developed by PAIR

**Issue No. 909 opened by SachidanandAlle on 2022-08-04 at 16:40:50 Type: og**

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Caching of volumes to GPU during training of deepedit (radiology app)

**Issue No. 874 opened by nvahmadi on 2022-07-14 at 15:41:09 Type: og**

Currently, it is not possible to cache data to GPU during training of  
deepedit, to accelerate training as described in the [Fast Training  
Tutorial](https://github.com/Project-MONAI/tutorials/blob/main/acceleration/fast\_training\_tutorial.ipynb)  
from MONAI-Core.  
  
This idea was already mentioned in PR #485, which already implemented  
other acceleration techniques (e.g. DiceCE Loss, Novograd optimizer,  
ThreadDataLoader).  
  
I tried putting the two transforms `ToTensord()` and `ToDeviced()`  
before the first Randomized transform, but that throws an error that a  
torch tensor cannot be cast to a numpy tensor (the error is thrown in  
the transform `AddInitialSeedPointMissingLabelsd()`).  
  
Looking into the deepedit training transforms, the reason for the above  
error is probably the computation of a chamfer distance function using  
scipy\'s `distance\_transform\_cdt`. I saw in the MONAI-Core discussion  
[#1332](https://github.com/Project-MONAI/MONAI/issues/1332) that  
\@tvercaut notified us about their recent work  
[FastGeodis](https://github.com/masadcv/FastGeodis), which allows for  
fast computation of Euclidean/Geodesic distance functions based on cuda  
(torch-compatible!).  
  
It would be great to revisit the idea of PR #485 and offer caching of  
images to GPU during training. My simple attempt above is not  
sufficient: apart from having to make  
`AddInitialSeedPointMissingLabelsd()` torch-based, the Multi-GPU  
scenario requires distributed caching across GPUs - I am not sure where  
in the MONAI-Label code this would go.

**Additional Information:**

* Milestones: None
* Assignees: diazandr3s, tangy5

No comments at the moment!

Add monitoring metrics based on prometheus

**Issue No. 803 opened by SachidanandAlle on 2022-05-24 at 19:14:54 Type: og**

<https://prometheus.io/>  
  
X axis  
  
- Track latency per request  
- Track latency per model  
  
Y axis  
  
- Capture latencies for pre/post, infer, write  
- Capture latencies for each transform

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Feature request: Support multiple annotations on the same image instance

**Issue No. 787 opened by jlvahldiek on 2022-05-16 at 14:29:18 Type: cement**

To create high-quality datasets, it is necessary to have the same image  
annotated multiple times by different people (e.g. double or triple  
segmentations). At the moment there is no interface for this in  
MONAILabel.  
  
My team could also contribute to this, but I am not sure what is the  
best way to implement this feature. Certainly some changes need to be  
made to the datastore, filter functions and plugins. Also some sort of  
user management is then necessary.  
  
Currently, for multiple annotations, we use a workaround with multiple  
instances of MONAILabel sharing the same app. Each instance is then used  
by a different annotator. Later, the data must then be merged in a  
cumbersome way.  
  
I would prefer that this functionality be integrated into MONAILabel in  
a bundled way. I am relatively sure that other teams will need this  
feature as well (@LilianaCaldeira, \@dtruhn, \@kbressem).

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

OHIF support for multi-label scribbles

**Issue No. 756 opened by masadcv on 2022-04-24 at 23:27:49 Type: og**

Recent updates added multi-label scribbles support to monai label and  
slicer. To have feature parity with OHIF, similar updates are needed to  
make scribbles work with recent changes.  
  
This involves (and is not limited to) the following steps:  
  
- Add support in OHIF to work on specific labels within multi-label  
 setting  
- Add support in OHIF to communicate segmentation labels metadata  
 (similar to how its done currently in 3D Slicer plugin)  
- Properly update/populate segmentation labels in OHIF, given output  
 from a scribbles flow

**Additional Information:**

* Milestones: None
* Assignees: masadcv

No comments at the moment!

Use standard terminology to define content of a segment

**Issue No. 724 opened by lassoan on 2022-04-04 at 19:03:46 Type: cement**

Specifying meaning of segment by label values or names is very limited  
and fragile. It is practically impossible to use consistent label values  
when the same data sets are used for multiple projects, usage of label  
names are too error-prone (due to spelling mistakes or differences in  
spelling), and meaning of labels names are often not well defined.  
  
I would recommend identifying segments using a universally applicable  
standard terminology.  
  
Since DICOM has to be supported anyway and DICOM terminology  
specification is so generic yet fairly minimal, it could be a good  
choice in MONAILabel (and in general in MONAI).  
  
A short summary of how DICOM specifies a segment:  
  
You need to specify a `category` and a `type` at a minimum. For example  
a tumor is defined as `Morphologically Altered Structure/Mass`. Or,  
liver is defined as `Tissue/Liver`. You can have a few more optional  
values, the complete list is this:  
  
 Name Required Examples  
 -------------------------- ---------- ---------------  
 category Y Tissue  
 type Y Mass, Liver  
 type modifier N Left,Right  
 anatomic region N Kidney, Liver  
 anatomic region modifier N Left, Right  
  
These values are not just stored as user-defined strings, because then  
you would need to define the exact meaning of that name, translations to  
different languages, etc., which would be a lot of work. Instead,  
existing standard terminologies (collection of terms with well-defined  
meaning) can be leveraged, such as  
[`SNOMED CT`](https://www.nlm.nih.gov/healthit/snomedct/snomed\_overview.html)  
for clinical terms, or  
[`TA2`](https://en.wikipedia.org/wiki/Terminologia\_Anatomica)  
terminologia anatomica for anatomical terms. So the idea is to just use  
these existing terms by specifying the name of the terminology and the  
code value according to that terminology. The end result is that you  
store each value as 3 strings:  
  
- `coding scheme designator`: defines the terminology, for example  
 `SCT` (SNOMED CT)  
- `code value`: code value in the chosen terminology, for example  
 `4147007`  
- `code meaning`: this is optional, for convenience only, a  
 displayable human-readable string that can be displayed, for example  
 `Mass`  
  
Slicer already uses this approach, user can choose from standard terms  
by a few clicks, and the chosen term is saved in to the segmentation  
file.  
  
<https://github.com/Slicer/Slicer/releases/download/docs-resources/terminology\_selector\_advanced.png>

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Out of source default saving

**Issue No. 614 opened by rijobro on 2022-02-04 at 17:55:33 Type: cement**

Apologies for all of the different issues.  
  
I would love to see the default save locations be outside of both the  
`--studies` and `--app` folders. If I use the same app for two different  
runs (with different data, for example), then using the default  
parameters, they will overwrite each other (each will save the  
`train\_stats.json` in the same location).  
  
I feel it would be cleaner if the user were obliged to supply an  
`--output` argument and store all input in there. This would keep both  
the input data and app folders clean, allowing multiple runs with  
minimal changes. This would also solve this problem:  
<https://github.com/Project-MONAI/MONAILabel/pull/610>.

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

MONAI Label and OHIF 3

**Issue No. 603 opened by patricio-astudillo on 2022-02-03 at 13:06:24 Type: cement**

Hi \@diazandr3s, \@dannyrb and \@SachidanandAlle,  
  
I\'m from the healthcare department of Robovision and we have a huge  
interest in using MONAI Label together with OHIF. Is there a plan to  
increase the functionalities of OHIF (with regards to the 3D  
capabilities e.g. segmentation)? And combine MONAI Label with the latest  
version (currently 3) of OHIF?  
  
Thanks in advance, Patricio

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Multi Label support for DeepEdit in OHIF

**Issue No. 574 opened by SachidanandAlle on 2022-01-03 at 22:13:46 Type: og**

Multi Label support for DeepEdit in OHIF

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Resource Scheduler API

**Issue No. 446 opened by aihsani on 2021-10-07 at 13:32:25 Type: og**

\*\*Is your feature request related to a problem? Please describe.\*\*  
Currently MONAI Label Server assumes it has virtually unlimited space  
when running training, batch inference, and on-demand inference tasks  
(caused by user clicks). In cases where all or some of the above tasks  
do not fit in the system, the most recent task will fail with an out of  
memory error and not run, causing the user to manually guess if the  
system is able to run the task in the future when resources are freed  
up.  
  
\*\*Describe the solution you\'d like\*\* It is desirable to have an API  
that allows both MONAI label server developers and MONAI Label App  
developers to priritize and schedule a task when one is required to run.  
Due to the nature of the training, batch-inference, and on-demand  
inference tasks it is required that preemption be implemented as, for  
instance, an on-demand inference always takes precedence over any other  
currently running tasks. Saving process state when preempting is out of  
the scope of this feature request.  
  
\*\*Describe alternatives you\'ve considered\*\* We are considering PyDCGM  
as the library that will enable the resource scheduler to keep track of  
the running tasks, their memory utilization in the GPU, GPU attachment,  
etc.

**Additional Information:**

* Milestones: None
* Assignees: aihsani

No comments at the moment!

Multi-Label Testing for Active Learning Strategies

**Issue No. 442 opened by finalelement on 2021-10-05 at 13:42:12 Type: og**

Multi-label datasets where there are 2 or more than 2 labels associated  
in an annotation volume. Active learning needs to be tested. The current  
AL strategy of entropy would need to be tested for multi-label.

**Additional Information:**

* Milestones: None
* Assignees: finalelement

No comments at the moment!

Not include trained image/label pair in validation

**Issue No. 440 opened by SachidanandAlle on 2021-10-05 at 09:24:59 Type: og**

Currently all submitted labels are used for training at some stage..  
instead isolate validation image+label pair

**Additional Information:**

* Milestones: None
* Assignees: None

No comments at the moment!

Multiple image modality Support

**Issue No. 241 opened by finalelement on 2021-07-26 at 15:34:07 Type: og**

This Milestone needs further clarification? Objectives? Deliverables?

**Additional Information:**

* Milestones: None
* Assignees: finalelement, diazandr3s

No comments at the moment!