Supplementary Material – Continual Segment: Towards a Single, Unified and Non-forgetting Continual Segmentation Model of 143 Whole-body Organs in CT Scans

Contents

A. Dataset Details

We describe the dataset details (one public and three private multi-organ datasets) used in our experiment here. For the public dataset TotalSegmentator [7], it consists of 1204 CT scans of different body parts with total 103 labeled anatomical structures (26 major organs, 59 bone instances, 10 muscles, and 8 vessels). Note that the face label is removed as it is an artificially created label for patient de-identification purpose after blurring the facial area. The detailed organ 103 organ instance list can be found in the link https://github.com/wasserth/TotalSegmentator. For the three in-house multi-organ datasets, they are head & neck organ dataset (denoted as HNOrgan), chest organ dataset (denoted as ChestOrgan) and dedicated esophageal cancer dataset (denoted as EsoOrgan). In HNOrgan, each of the 447 head and neck CT scans has 13 head and neck organs labeled: brainstem, eye (left and right), lens (left and right), optic nerve (left and right), optic chiasm, parotid (left and right), spinal cord, temporomandibular joint (TMJoint, left and right). ChestOrgan contains 292 chest CT scans with 31 chest anatomical structures annotated including major organs, muscles, arteries and veins. The detailed list is as follow: esophagus, sternum, thyroid left, thyroid right, trachea, bronchus left, bronchus right, anterior cervical muscle, scalenus muscle, scalenus anterior muscle, sternocleidomastoid muscle, ascending aorta, descending aorta, aorta arch, common carotid artery left, common carotid artery right, pulmonary artery, subclavian artery left, subclavian artery right, vertebral artery left, vertebral artery right, azygos vein, brachiocephalic vein left, brachiocephalic vein right, internal jugular vein left, internal jugular vein right, pulmonary vein, subclavian vein left, subclavian vein right, inferior vena cava, superior vena cava. There are four organs in ChestOrgan that are overlapped with organs in TotalSegmentator (esophagus, pulmonary artery, superior vena cava, trachea). The EsoOrgan collects 640 diagnostic CT scans of advanced esophageal cancer patient where only the esophagus is labeled. By combining all datasets, we have total 103+27+13 organ classes from 2583 unique patients. For each of these four datasets, 20% are randomly chosen as the testing set, while the rest is used as training + validation. Detailed training/validation dataset split in the decoder optimization module can be found in the Implementation Details section of this supplementary material.

In addition, for the purpose of training and validating our abnormality detection module, we further collect CT scans from 304 esophageal (private) and 625 lung cancer (public with tumor labels) patients where the 3D tumor masks are delineated at the pixel level. These combine as the lung/esophageal tumor classes from additional 929 patients.

B. Implementation Details

The default nnUNet backbone in 3D full resolution setting is adopted in our work, including a 6-block encoding path and a 5-block decoding path. Each encoding block consists of the following consecutive operations: a convolution, an instance normalization, a Leaky ReLU unit, followed by max-pooling operator.

The total training epoch for the baseline TotalSegmentator is 8000 with 250 iterations per epoch, and the training epoch for each of the in-house datasets (served as performance upper bound) is 1000 with 250 iterations per epoch. The batch size is 2. The optimizer is stochastic gradient descent with a Polynomial learning rate policy. The initial learning rate is 0.01 with a Nesterov momentum of 0.99. Default "moreDA" data augmentation is adopted, e.g., horizontal flipping, random rotation in the x-y plane with ± 10 degrees, intensity scaling with a ratio between $[0.75, 1.25]$,

adding Gaussian noise with zero mean and $[0, 0.1]$ variance. The total average training time is 2.5 GPU days per 1000 epochs. For model inference, the average running time for the proposed framework, before the decoding path optimization, is approximately 15 minutes per patient. After the decoding path optimization, the average inference time is less than 5 minutes per patient. All models are developed using PyTorch and trained on one NVIDIA A100 GPU.

Decoder Optimization – NAS Setting. For NAS, we divide each dataset into 1) 60% for network training, 2) 30% for NAS training, and 3) 10% for validation and ablation evaluation. To train the learnable weight for selecting the architecture of each decoding block, we first fix the convolution kernel selection weight to $\frac{1}{C_t}$ for 40% of the total epochs, where C_t denotes the number of classes of the t^{th} task. Then we alternatively update the convolution kernel selection weight and the decoder parameters for the rest epochs. The initial learning rate is set to 0.01 for all tasks. The learning rate is decayed following the Polynomial learning rate policy. After NAS training is complete, we follow the same 'moreDA' data augmentation scheme and retrain the searched decoding path from scratch using the re-divided dataset in a 4:1 'training-validation' ratio. The searched decoding blocks for each task are shown in Table B.1.

Decoder Optimization – Pruning Setting. We perform a block-wise teacher-student knowledge distillation (KD), aiming to further compress the decoder by replacing the searched convolutional kernels with the projection kernels. The mean-square error loss is used to match the feature map outputs of the teacher block to the student block. To ease the optimization difficulty, we first distill the deeper decoding blocks (lower resolution), then move to shallower blocks. Once the KD training of the deeper block is saturated, we freeze the deeper student blocks and move to the shallower ones. When conducting this block-wise KD, the shallower convolutional block needs to choose if to receive output feature maps from the deeper teacher blocks or those from the student block. Under this setting, if the feature map difference from previous teacher and student block is large, it would affect the subsequent feature response in the next shallower block causing the degenerated segmentation. To conquer this, we employ a simple yet effective approach: Once the distillation of the student block is finished, we use a smaller learning rate (e.g., $0.1\times$) and finetune the shallower teacher block using the deeper student block's output feature maps. We monitor the before-and-after performance drop. If the drop is less than 1% in terms of Dice score, we keep the deeper student block, otherwise, we restore the teacher block. To speed up the training process, block-wise side supervision is also used during training. The pruned decoder for each task is demonstrated in Table B.1.

Body-part and Anomaly-aware Decoder Merging. To generate the final 143-class organ segmentation output, we need to combine and merge the predictions from all decoders. There are two major issues in this step. First, since each training dataset/task often covers a specific body part, the task-specific decoder might generate false positives in body parts that are not covered in this specific dataset (because that decoder never sees other anatomic regions). We propose a straightforward yet effective solution to reduce these false positives due to the body part coverage: for each decoder, we pre-compute the body part coverage rate using all the data in this dataset/task. In this way, for a specific decoder, voxels outside the covered body parts would have a lower/zero weight, and we can use this weight to significantly decrease the confidence score of the false positive segmentations out of the covered body parts. Specifically, we first generate the body part map using an automated body part regression algorithm [8]. Then, by overlapping the bounding box of all labeled organs to the body part map, we compute the volume-wise overlapping ratio between the bounding box and the body part map. Then, a body part distribution map is generated for a patient, e.g., 80% in the chest, and 20% in the abdomen. This calculation is repeated for all patients in the dataset, and finally averaged to get a pre-computed body part distribution map $\hat{Y}^t_\beta$. The detailed body part distributions for the three in-house datasets are illustrated in Table B.2.

The second issue is that some decoders might not see the patients with abnormalities (e.g., tumors). Hence, the predictions may have false segmentation in the anomaly region. To resolve this problem, we first supplement the framework with an anomaly segmentation head. In our work, we use the esophageal and lung tumor dataset to train this head as an illustration. More abnormality datasets can be utilized, such as DeepLesion [9]. Then, we exploit the tumor predictions $\hat{Y}^\epsilon$ to generate an anomaly weighting map. The averaged tumor size p^ϵ is pre-calculated using the annotated tumor mask and used as the standard deviation of a Gaussian filter of zero-mean to further smooth the tumor prediction $\hat{Y}^\epsilon$. Here, we assume that, at location j, the prediction $\hat{Y}^t(j)$ of the t^{th} decoder is less confident if $\hat{Y}^\epsilon(j)$ is of high value.

To combine the predictions, we perform a voxel-wise selection by choosing the most confident prediction from all decoding heads, considering both the body part distribution map and the anomaly map. As the entropy function produces the highest value when the input closes to 0.5 (the most uncertain prediction), we could find the most confident prediction when the input closes to 0 or 1. Eq. (3) in the main text is used to combine the body part map $\hat{Y}^t_\beta$ and anomaly distribution map $\hat{Y}^\epsilon$. When there is no tumor prediction $\hat{Y}^\epsilon(j) = 0$ and the organ prediction is within the decoder's body part distribution range $\hat{Y}^t_\beta(j) = 1$, the output score is considered as confident and sets $M^t(j) = 0$.

Table B.1. The detailed auto-searched and pruned decoding architecture based on nnUNet. Note that decoder block 5 refers to the deepest decoding block.

		Decoder Block 5	Decoder Block 4	Decoder Block 3	Decoder Block 2	Decoder Block 1
TotalSeg	NAS	P3D	P3D	2D	P3D	2D
	Pruned	P3D	P3D	2D	Projection	2D
ChestOrgan	NAS	3D	P3D	P3D	2D	2D
	Pruned	Projection	P3D	P3D	2D	2D
HNOrgan	NAS	3D	P3D	3D	P3D	P3D
	Pruned	Projection	P3d	3D	P3D	P3D
EsoOrgan	NAS	P3D	2D	Projection	2D	Projection
	Pruned	Projection	Projection	Projection	Projection	Projection

Table B.2. The body part distributions of the in-house datasets.

	Head	Chest	Abdomen	Pelvic
ChestOrgan	8.2%	89.4%	2.4%	0%
HNOrgan	96.5%	3.5%	0%	0%
EsoOrgan	0%	96.3%	3.7%	0%

On the other hand, the other states are considered uncertain and set $M^t(j) = 0.5$. The confidence map is generated using Eq. (4) of the main text. For a voxel at location j, using Eq. (5), we collect the confidence values from all tasks whose foreground prediction is not in the background. The final output class $\hat{\mathbf{Y}}(j)$ is determined using the prediction $\hat{Y}^t(j)$, of which with the smallest $H^t(j)$.

C. Additional Ablation Results and Analysis

In our main text, we have briefly demonstrated and discussed the final segmentation performance and forgetting curve of our method and other comparison methods at each continual learning step in Table 1 and Fig 4 (Section 4.3). Here, we show the detailed numeric results of Fig 4 in Fig C.1 and Table C.1 and provide more in-depth discussion on results achieved by our model and other comparison methods.

First, the mean DSC and the forgetting rate in each dataset/step of two CSS orders are detailed in Table C.1. The mean DSC and average forgetting rate over all 143 organs at the last step is also shown. Besides the observations discussed in the main text, several additional findings can be noticed. First, the slight decrease of DSC of our method in the sub-figures of Fig 4 (mean DSC over all learned organs at the current step) is not due to our model forgetting, but simply because the achieved DSC values are lower in new datasets/steps. E.g., the upper bound mean DSC in ChestOrgan dataset is only 78.45% (as compared to 93.24% in TotalSegmentator dataset). As shown the last three rows of two CSS order in Table C.1, our method completely avoids forgetting of old knowledge when continually learning new dataset/task because of our proposed framework (frozen general encoder, light-weighted decoders, and body-part and anomaly-aware merging). In contrast, other distillation-based CSS methods all experienced severe forgetting with more than 50% forgetting rate at the last step.

Second, regarding the step-wise results of comparison methods. It is observed that methods based on the output-level knowledge distillation and MiB losses (MiB [1], ILT [5] and LISMO [4]) suffer from catastrophic forgetting after the last step (overall mean DSC $< 25\%$ and forgetting rate $> 64\%$). In contrast, although PLOP [3] also has a large forgetting rate (about 50%), the overall performance is noticeably better as compared to the other three methods. For instance, the overall mean DSC is 39.01% for PLOP in CSS order A, which is at least 3 times higher than that for MiB, ILT and LISMO. The increased ability to keep old knowledge in PLOP might come from the applied entropy-based pseudo-labeling and the knowledge distillation on intermediate features in both encoder and decoder.

Third, regarding two continual segmentation orders, the main difference is the learning order of ChestOrgan and HNOrgan: order A first learns ChestOrgan in step-2, then HNOrgan in step-3, while order B exchanges the dataset in step-2 and 3. It is observed in Table C.1 and Fig C.1 that the comparison methods forgetting rate at learning step-2 for the TotalSegmentator dataset is higher in order B than that in order A. E.g., TotalSegmentator DSC of MiB [1] at step-2 is decreased from 93.24% $\rightarrow$ 21.96% in order B vs. from 93.24% $\rightarrow$ 45.80% in order A. Notice that order B learns the HNOrgan right after TotalSegmentator at step-2, and HNOrgan contains CT images only focusing in the head and neck region, where TotalSegmentator has organs mostly labeled in the chest, abdomen, and pelvic regions. As a result, MiB can no longer see the chest, abdomen, and pelvic regions at step-2, which causes catastrophic forgetting in these body parts resulting in a significant performance drop. Instead, order A learns the ChestOrgan at step-2, and ChestOrgan covers all the chest and neck regions as well as most parts of the abdomen. Hence, MiB is still able to rehearse some old knowledge over these overlapping body parts so that the forgetting rate is reduced as compared to that in order B. A similar trend can be found in the forgetting curves of ChestOrgan and HNOrgan. These findings show that for the multi-organ continual segmentation, the forgetting rate of other comparison methods is closely related to the overlapping range of body parts in each dataset/step. In contrast, our proposed

Table C.1. Mean DSC (%, ↑) and forgetting rate (%, ↓) of our method and other comparison methods in each dataset/step of two continual segmentation orders. The last column 'All Learned Classes' lists the mean DSC and average forgetting rate [2] over all learned organs/classes at each step. The DSC in TotalSegmentator at step 1 of all comparison methods is the upper bound 93.24%, while DSC for our method is slightly lower with 92.98% at step 1 due to the decoder compression/pruning. (Sub-figures in Fig. 4 from left to right are corresponding to the numeric results in column 'All Learned Classes', 'TotalSeg', 'ChestOrgan' and 'HNOrgan' of this table.)

Methods	Step	TotalSeg (103)		ChestOrgan (31)		HNOrgan (13)		EsoOrgan (1)		All Learned Classes	
		DSC	Forget	DSC	Forget	DSC	Forget	DSC	Forget	DSC	Avg. Forget
upper bound		93.24	—	78.45	—	84.35	—	87.15	—	89.02	—
Order A: TotalSeg → ChestOrgan → HNOrgan → EsoOrgan											
MiB [1]	2	45.80	50.87	78.40	—					50.56	50.87
	3	11.68	87.48	25.66	67.27	84.22	—			20.86	77.37
	4	7.65	91.80	19.24	75.46	6.37	92.43	86.92	—	8.51	86.56
ILT† [5]	2	48.50	47.98	77.78	—					54.40	47.98
	3	13.68	85.33	28.74	63.04	84.21	—			23.08	74.19
	4	10.87	88.34	27.87	64.17	6.39	92.42	85.75	—	11.99	81.64
PLOP [3]	2	59.13	36.59	76.52	—					62.40	36.59
	3	39.46	57.68	49.19	35.72	83.17	—			45.10	46.70
	4	37.30	59.99	51.74	32.38	25.38	69.48	82.90	—	39.01	53.95
LISMO [4]	2	52.57	43.62	78.48	—					57.74	43.62
	3	13.59	85.42	29.05	62.99	84.36	—			22.86	74.21
	4	10.82	88.40	28.24	64.02	6.30	92.54	87.12	—	12.11	81.65
Ours	2	92.98	0.00	78.26	—					88.27	0.00
	3	92.98	0.00	78.26	0.00	83.97	—			87.88	0.00
	4	92.98	0.00	78.26	0.00	83.97	0.00	86.94	—	87.87	0.00
Order B: TotalSeg → HNOrgan → ChestOrgan → EsoOrgan											
MiB [1]	2	21.96	76.45			84.49	—			29.42	76.45
	3	10.72	88.50	78.46	—	6.38	92.45			23.85	90.48
	4	10.35	88.90	65.63	16.35	6.29	92.55	86.79	—	20.20	65.94
ILT† [5]	2	21.62	76.81			84.25	—			29.09	76.81
	3	14.10	84.88	78.02	—	8.48	89.93			26.13	87.41
	4	13.12	85.93	67.28	13.76	6.18	92.66	85.52	—	22.44	64.12
PLOP [3]	2	45.11	51.62			83.59	—			49.70	50.87
	3	31.90	65.79	76.13	—	17.56	78.99			38.99	72.39
	4	30.82	66.95	70.18	7.81	15.77	81.13	83.41	—	36.63	51.96
LISMO [4]	2	24.36	73.88			84.35	—			31.51	73.88
	3	15.08	83.83	78.47	—	7.85	90.69			26.84	87.26
	4	14.04	84.94	67.19	14.37	6.15	92.71	86.87	—	23.09	64.01
Ours	2	92.98	0.00			83.97	—			91.27	0.00
	3	92.98	0.00	78.26	—	83.97	0.00			87.88	0.00
	4	92.98	0.00	78.26	0.00	83.97	0.00	86.94	—	87.87	0.00

architectural-based method is learning-order and body-part invariant, which facilitates the model deployment in clinical practice.

Last, we evaluate the impact of Alternative General Encoders. We recommended starting with TotalSegentator as it covers most body parts for comprehensive feature extraction. Alternatively, other datasets can be used as the starting dataset to train General Encoder. We trained the General Encoder using the ChestOrgan dataset. A tolerable performance drop (<1% Dice) is observed in the CSS **Order A** final results. The assumed reason is that the torso region includes diverse anatomies and covers most of the body parts, and hence exhibits similar performance as the one trained using the TotalSegmentator dataset. Yet, when using the HNOrgan dataset to train the General Encoder, we notice a markedly 3% Dice drop in the final results.

D. Detailed Results of Individual Organs

We provide detailed organ segmentation results of our final model as well as the upper bound nnUNet performance trained and evaluated on TotalSegmentator, ChestOrgan, HNOrgan datasets (shown in Table D.2, D.3, D.4). The final performance on the EsoOrgan dataet has been reported

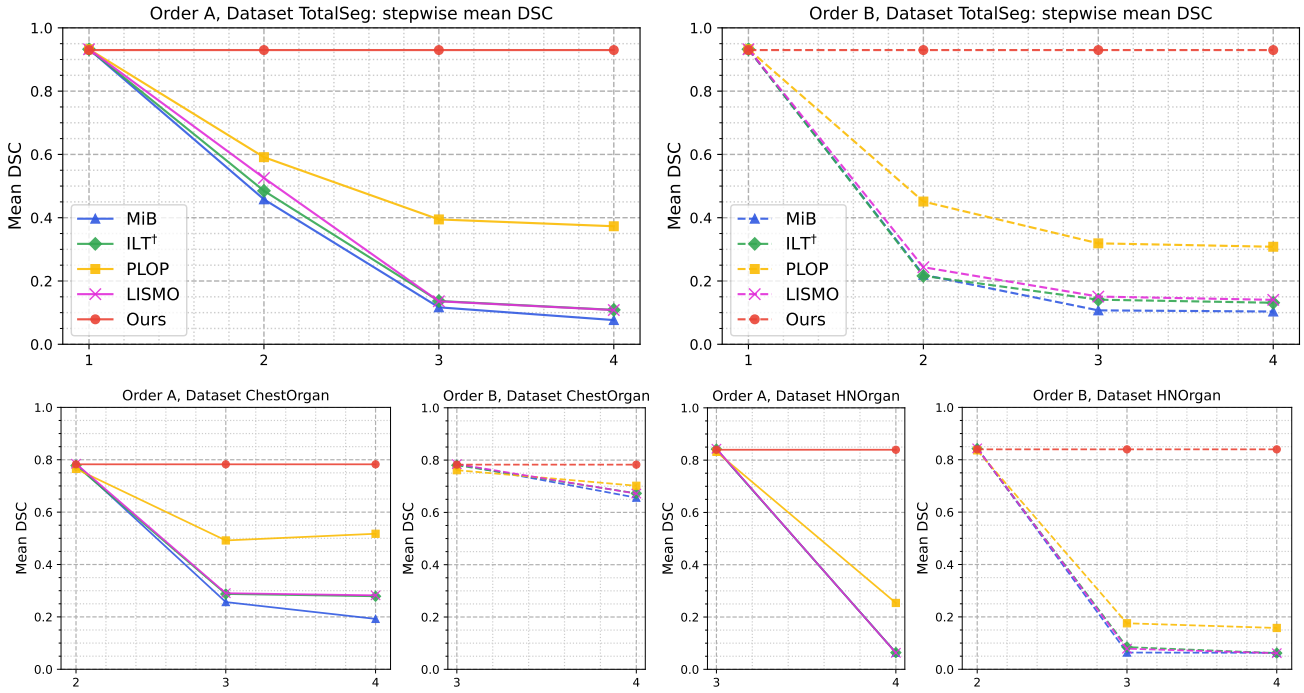

Figure C.1. The mean DSCs over each dataset at each step of **Order A** (solid line) and **Order B** (dashed line).

Table D.1. TotalSegmentator [7] label list of each organ group.

TotalSeg Organ Group	TotalSeg Organ Labels
Main Chest Organs	13—17, 42—48
Cardiovascular Vessels	7, 8, 9, 49, 51, 52, 53, 54
Excretory Organs	2, 3, 55, 57, 104
Main Abdomen Organs	1, 4, 5, 6, 10, 11, 12, 56
Head	50 ('face'-93 is exluded)
Vertebraes	18—41
Ribs	58—81
Other Bones	82—92
Muscles	94—103

Table D.2. Mean DSC (%, ↑), HD95 (mm, ↓) and ASD (mm, ↓) of 8 anatomical organ groups in TotalSeg (total 103 full body organs) of upper bound model and our final model.

TotalSeg Organ Group	Upper Bound			Ours		
	DSC	HD95	ASD	DSC	HD95	ASD
Chest Main Organ	96.66	1.71	0.35	95.62	2.54	0.43
Cardiovascular Vessels	91.75	2.33	0.55	91.97	2.96	0.74
Excretory System	93.28	4.24	1.22	93.53	4.45	1.28
Abdomen Main Organ	89.44	3.45	0.80	91.05	3.96	0.88
Head	94.51	2.44	0.62	94.61	3.02	0.69
Vertebraes	92.94	2.01	0.48	92.65	2.92	0.68
Ribs	91.54	4.24	1.06	91.49	5.15	1.20
Other Bones	95.02	6.94	2.08	93.14	7.85	2.27
Muscles	96.09	2.17	0.38	95.92	2.98	0.54

in the Table 1 of the main text. For organs in TotalSegmentator, due to the large amount of organs, we choose to group the 103 whole body organs into eight anatomical groups (Table D.1) and report the average scores of each group (see Table D.2). As shown in those tables, our final model performs closely to the upper bound accuracy when training separate nnUNet models on each dataset. There are no organs experiencing a large performance drop. The overall slightly drop in DSC and increasing in HD95/ASD is because of the decoder pruning process.

E. Reimplementation of Comparison Methods

For all comparison methods, we start with the same pretrained nnUNet model on TotalSegmentator dataset, which has been trained using 3D nnUNet setting for 8000 epochs, with 250 iterations per epoch and initial learning rate 0.01. After that, the model is finetuned sequentially on continual segmentation tasks (ChestOrgan, HNOrgan and EsoOrgan), where each dataset are finetuned for 500 epochs, with 250 iterations per epoch and initial learning rate 0.005. All the other nnUNet settings, such as data augmentation, remain the same as our implementation. Moverover, since our segmentation datasets/tasks are 3D CT scans (different from the previous continual segmentation works in natural images), adjustments to these comparison methods are required (extending 2D methods to 3D), as well as transferring their implementations to the nnUNet framework. We describe the detailed re-implementation of previous methods, especially our modifications, in the following subsections.

MiB. MiB [1] proposes two marginal losses, or unbiased losses to solve the background shift issue in continual seg-

Table D.3. Mean and standard deviation of DSC (%, ↑), HD95 (mm, ↓) and ASD (mm, ↓) of each ChestOrgan organ (total 31 chest organs) of upper bound model and our final model. _L and _R refer to the left and right.

ChestOrgan	Upper Bound			Ours		
	DSC	HD95	ASD	DSC	HD95	ASD
Esophagus	85.41±4.12	4.93±3.32	0.68±0.42	85.22±4.30	5.78±2.81	0.67±0.34
Sternum	90.30±3.27	5.07±5.00	1.16±1.07	90.26±3.64	5.83±5.33	1.18±2.07
Thyroid_L	84.13±4.71	2.87±1.36	0.66±0.36	84.10±3.98	3.54±1.23	0.73±0.33
Thyroid_R	82.79±6.28	3.39±2.34	0.79±0.41	82.75±5.32	4.18±1.89	0.88±0.38
Trachea	93.74±2.19	4.74±3.45	1.09±0.88	93.68±1.83	5.45±2.76	1.03±0.69
Bronchus_L	86.53±4.39	4.84±2.55	0.48±0.33	86.34±5.15	5.45±2.29	0.44±0.29
Bronchus_R	75.86±13.79	9.18±7.51	2.35±2.87	75.88±13.41	10.42±7.67	2.41±2.70
Anterior cervical muscle	69.23±8.05	6.31±5.67	1.34±1.22	69.02±6.79	7.08±4.54	1.39±1.20
Scalenus muscle	74.26±4.42	5.83±3.79	0.83±0.38	74.24±5.18	6.98±3.86	0.79±0.38
Scalenus anterior muscle	77.89±6.34	4.37±3.46	0.95±1.03	77.82±6.89	5.04±3.17	0.83±0.83
sternocleidomastoid muscle	82.17±3.92	4.01±2.89	1.16±1.11	82.10±3.35	4.89±3.23	1.06±0.81
common carotid artery_L	78.09±7.65	9.33±17.74	2.17±4.33	77.97±8.21	9.98±16.47	2.28±4.33
common carotid artery_R	73.92±11.08	12.43±16.29	2.84±4.72	73.77±10.63	13.37±17.83	2.90±4.81
Pulmonary artery	90.11±3.64	6.64±3.09	1.28±0.67	89.93±5.46	7.11±3.97	1.13±0.77
Subclavian artery_L	71.79±9.26	20.26±20.29	2.37±3.40	71.78±9.50	23.15±22.32	2.28±3.46
Subclavian artery_R	80.03±5.87	11.93±13.24	2.04±2.11	79.97±6.07	12.14±13.72	2.19±2.90
Vertebral artery_L	47.72±19.66	20.25±21.22	6.65±11.83	46.62±20.32	21.27±20.94	6.51±11.73
Vertebral artery_R	45.28±18.38	19.90±21.07	6.33±11.35	42.66±17.08	20.81±17.43	6.40±11.12
Ascending aorta	93.08±2.54	5.41±2.40	1.39±0.77	93.05±2.17	6.25±2.50	1.34±0.97
Descending aorta	97.13±1.75	3.21±2.21	0.72±0.37	97.05±0.94	3.73±1.98	0.67±0.35
Aorta arch	92.12±9.08	4.23±2.72	1.25±1.42	92.11±7.53	4.93±2.80	1.27±1.42
Azygos vein	73.29±11.53	8.99±14.07	1.68±3.58	73.25±10.02	9.79±14.16	1.65±3.53
brachiocephalic vein_L	85.80±5.57	3.27±2.35	0.35±0.29	85.73±5.00	4.13±2.76	0.35±0.26
brachiocephalic vein_R	85.83±5.07	4.71±1.88	0.87±0.54	85.82±5.70	5.30±2.37	0.89±0.57
internal jugular vein_L	74.66±14.57	12.77±16.31	2.88±4.51	74.63±14.41	14.19±14.63	2.90±3.81
internal jugular vein_R	78.28±8.73	12.03±16.86	2.95±3.68	78.24±8.19	15.86±13.63	3.02±3.52
Pulmonary vein	70.62±8.24	7.04±2.80	1.53±0.67	70.57±7.83	7.81±2.91	1.49±0.76
Subclavian vein_L	63.32±14.72	9.64±6.69	1.99±1.90	63.32±15.21	10.14±7.88	2.00±1.71
Subclavian vein_R	60.59±12.21	11.62±8.13	2.75±2.35	60.60±12.92	13.89±10.63	2.83±2.20
Inferior vena cava	82.51±6.23	7.95±5.58	1.69±1.13	82.45±5.82	9.01±5.56	1.71±1.25
Superior vena cava	85.38±3.83	5.85±3.55	1.36±0.88	85.29±3.85	6.60±3.45	1.33±0.75

Table D.4. Mean and standard deviation of DSC (%, ↑), HD95 (mm, ↓) and ASD (mm, ↓) of each HNOrgan organ (total 13 head-neck organs) of upper bound model and our final model. _L and _R refer to the left and right.

HNOrgan	Upper Bound			Ours		
	DSC	HD95	ASD	DSC	HD95	ASD
BrainStem	91.42±2.47	2.93±1.15	0.76±0.32	90.89±2.38	2.65±1.37	0.74±0.36
Eye_L	92.32±1.73	1.57±0.60	0.36±0.11	92.25±1.69	1.50±0.48	0.33±0.12
Eye_R	91.99±1.49	1.68±0.54	0.40±0.10	91.95±1.19	1.63±0.92	0.49±0.21
Lens_L	81.49±10.64	1.64±0.90	0.53±0.47	80.33±8.81	1.55±0.89	0.47±0.50
Lens_R	84.16±8.35	1.39±0.70	0.40±0.30	82.46±6.33	1.23±1.12	0.36±0.36
Optic Chiasm	67.04±13.34	3.73±1.66	1.03±0.65	66.60±13.29	3.59±1.91	0.96±0.71
Optic Nerve_L	74.34±6.77	3.05±2.75	0.56±0.27	74.72±6.93	2.96±3.27	0.61±0.51
Optic Nerve_R	75.15±6.47	2.64±0.80	0.51±0.30	73.64±7.07	2.49±1.18	0.56±0.35
Parotid_L	91.32±3.01	3.13±1.72	0.75±0.33	91.09±3.08	2.90±2.10	0.76±0.32
Parotid_R	90.93±2.87	3.22±1.94	0.86±0.46	90.94±2.97	3.03±2.10	0.86±0.67
TMJ_L	81.55±9.42	2.10±1.06	0.56±0.47	82.14±9.33	1.87±1.19	0.52±0.65
TMJ_R	84.81±8.57	1.85±1.06	0.43±0.39	84.70±9.28	1.69±0.93	0.42±0.50
Spinal cord	90.01±2.35	2.03±0.61	0.65±0.22	89.93±2.19	1.83±0.81	0.64±0.31

mentation in their original paper: unbiased cross-entropy (UNCE) loss, which merges the probabilities of old classes to the background label, and unbiased knowledge distillation (UNKD) loss, which merges the probabilities of all new classes (belonging unseen classes of the old model) to the background label. Notice that, the original unbi-

ased loss assumes that new classes from the current dataset are completely disjoint with all the old classes, however, this assumption is not holding in our datasets. E.g., TotalSegmentator and ChestOrgan contain four overlapping organs: inferior vena cava, trachea, esophagus and pulmonary artery. Therefore, in order to re-implement MiB losses in the nnUNet framework and make them compatibility with our datasets, we slightly modifies and generalizes both unbiased losses to handle overlapping labels in the continual learning setting. The modified UNCE loss is as follows:

$$\ell_{ce}^{\theta^t}(x,y) = -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}} \log \tilde{q}_x^t(i,y_i) \tag{1}$$

where:

$$\tilde{q}_x^t(i,c) = \begin{cases} q_x^t(i,c) & \text{if } c \neq b \\ \sum_{k\in\mathcal{Y}^{t-1}\setminus\mathcal{C}_p^t} q_x^t(i,k) & \text{if } c = b \end{cases} \tag{2}$$

Here, same notations referred to the original paper is used, except $\mathcal{C}_p^t = \mathcal{Y}^{t-1} \cup \mathcal{C}^t - b$, which indicates the overlapping classes (excluding background label) between current dataset $\mathcal{C}^t$ and all the previous classes $\mathcal{Y}^{t-1}$ at the learning step t. When calculating UNCE loss, we merge all the old labels to the background except the overlapping classes.

Similarly, we adapt UNKD loss as:

$$\ell_{kd}^{\theta^t}(x,y) = -\frac{1}{|\mathcal{I}|}\sum_{i\in\mathcal{I}}\sum_{c\in\mathcal{Y}^{t-1}\setminus\mathcal{C}_p^t} q_x^{t-1}(i,c)\log \hat{q}_x^t(i,y_i) \tag{3}$$

where:

$$\hat{q}_x^t(i,c) = \begin{cases} q_x^t(i,c) & \text{if } c \neq b \\ \sum_{k\in\mathcal{C}^t} q_x^t(i,k) & \text{if } c = b \end{cases} \tag{4}$$

In the above formula, overlapping organs from the old class set are excluded so that the knowledge distillation works on the real old classes that cannot be learned from the current dataset.

Using two modified losses, we always train the model with the latest labels and ignore the previously learned overlapping labels when overlapping organs occur. Thus, overlapping labels are trained directly using the cross-entropy loss and merged to the background in the knowledge distillation loss. In addition, we use the same hyperparameters as the MiB setting: the weight of UNKD loss are set as 10 with balanced classifier initialization strategy.

ILT. ILT [5] originally first proposes the continual semantic segmentation (CSS) protocol and provides a naive solution using an output-level knowledge distillation on the old classes ($\mathcal{L}'_D$) and a feature-level knowledge distillation on the intermediate features from encoder ($\mathcal{L}''_D$). This method leads to inferior performance and experienced severe forgetting as compared to MiB and other CSS methods on multiple natural image benchmarks [3, 6]. In order to improve ILT performance on our datasets/tasks, we modifies the original ILT setting and losses as follows: (1) ILT uses a frozen encoder setting (E_F) together with $\mathcal{L}'_D$, which is similar to our general encoder method, therefore, we re-implement ILT using this frozen encoder setting, as mentioned in the main paper; (2) since original ILT losses do not alleviate the background shift issue and have a large bias towards new classes (experiencing severe forgetting even with the frozen encoder), we additionally apply the MiB loss (Eq.2,4) to reinforce the decoder to preserve more old knowledge. In short, our re-implemented ILT can be treated as a frozen encoder version of MiB ($E_F + \mathcal{L}_{\text{mib}}$). Although leading to an improved performance as compared to the original ILT, this frozen encoder ILT version still has obvious knowledge forgetting as shown in Table 1 of the main text. This indicates that the frozen encoder with unbiased output-level knowledge distillation is not sufficient to preserve the old knowledge in CSS. In contrast, our proposed framework (general encoder + light-weighted decoder) can performance at the accuracy for the first time with real non-forgetting in CSS.

PLOP. PLOP [3] is originally implemented for 2D images, especially its multi-scale local distillation loss based on local POD. Local POD is a multi-scale feature pooling strategy consisting of computing width and height-pooled slices on multi-scale regions, which aims to better retain both global and local spatial knowledge from the old model. However, since our data are all 3D CT scans with an additional depth dimension, we specifically extend the local POD to higher dimensions when re-implementing the method. Two pooling strategies can be adopted for the 3D cases: (1) pooling 3D feature map along each single dimension and extracting three 2D projections along each axis:

$$\Phi(\mathbf{x}) = \left(\frac{1}{H}\sum_{h=1}^{H} \mathbf{x}\left[h,:,:,:\right] \middle\| \frac{1}{W}\sum_{w=1}^{W} \mathbf{x}\left[:,w,:,:\right] \middle\| \frac{1}{D}\sum_{d=1}^{D} \mathbf{x}\left[:,:,d,:\right] \right) \in \mathcal{R}^{(WD+HD+HW)\times C} \tag{5}$$

where notations follow the original PLOP paper. This pooling method can preserve enough spatial information meanwhile providing some level of plasticity to the model. (2) Pooling 3D feature map on two dimensions and only extract 1D projection along the remaining axis:

$$\Phi(\mathbf{x}) = \left(\frac{1}{HW} \sum_{h=1}^{H} \sum_{w=1}^{W} \mathbf{x}\left[h, w, :, :\right] \middle\| \frac{1}{WD} \sum_{w=1}^{W} \sum_{d=1}^{D} \mathbf{x}\left[:, w, d, :\right] \middle\| \frac{1}{HD} \sum_{h=1}^{H} \sum_{d=1}^{D} \mathbf{x}\left[h, :, d, :\right] \right) \in \mathcal{R}^{(H+W+D)\times C} \tag{6}$$

This pooling strategy has similar feature shape, however, when pooling on two axes together, most of the spatial information are lost and POD loss cannot retain the old knowledge. After comparing the performance using two strategies, we select the former one, which better handles the trade-off between model rigidity and plasticity.

For hyperparameters, the original paper uses the pod weighting factor of 0.01, which is too large for the 3D pooling case. Because the L2 norm of 3D pooled features is more than 10 times larger than that of 2D pooled features. In our experiments, we set this pod factor to 0.001. Other hyperparameters are consistent with those used in the original paper.

LISMO. The original LISMO [4] is designed based on nnUNet framework, so we are able to directly re-implement this method. We would like to mention several important differences between our datasets and those used in LISMO. In LISMO [4], it has a slightly improved result than MiB when segmenting five large abdominal organs in their experiment (using 3D low resolution of nnUNet). Under this setup, all five abdominal organs could be seen in each 3D training patch most of the time, which could frequently reinforce and rehearsal the model's ability on unseen organs in the current dataset through their memory module and prototype matching loss. However, this is not the case in our experiments, since many old organs are no longer able to observe in the new dataset due to non-overlapping body parts. E.g. abdominal organs cannot appear in the HNOrgan dataset. Moreover, the high resolution nnUNet version is used to meet the high segmentation accuracy required in practice and there are over 100 target organs spreading among the whole body range, so our patch size is impossible to cover most organs within each patch. Under this situation, the prototype matching loss is not able to compute on non-existing organs and the contrastive loss is not sufficient to constraint the feature distributions of these organs, which results in severe forgetting for the unobserved organs in our experiment.

References

[1] Fabio Cermelli, Massimiliano Mancini, Samuel Rota Bulo, Elisa Ricci, and Barbara Caputo. Modeling the background for incremental learning in semantic segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9233–9242, 2020. 3, 4, 5

[2] Arslan Chaudhry, Puneet K Dokania, Thalaiyasingam Ajanthan, and Philip HS Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 532–547, 2018. 4

[3] Arthur Douillard, Yifu Chen, Arnaud Dapogny, and Matthieu Cord. Plop: Learning without forgetting for continual semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4040–4050, 2021. 3, 4, 7

[4] Pengbo Liu, Xia Wang, Mengsi Fan, Hongli Pan, Minmin Yin, Xiaohong Zhu, et al. Learning incrementally to segment multiple organs in a CT image. In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 714–724. Springer, 2022. 3, 4, 8

[5] Umberto Michieli and Pietro Zanuttigh. Incremental learning techniques for semantic segmentation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 3, 4, 7

[6] Umberto Michieli and Pietro Zanuttigh. Continual semantic segmentation via repulsion-attraction of sparse and disentangled latent representations. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1114–1124, 2021. 7

[7] Jakob Wasserthal, Manfred Meyer, Hanns-Christian Breit, Joshy Cyriac, Shan Yang, and Martin Segeroth. Totalsegmentator: robust segmentation of 104 anatomical structures in ct images. *arXiv preprint arXiv:2208.05868*, 2022. 1, 5

[8] Ke Yan, Le Lu, and Ronald M. Summers. Unsupervised body part regression via spatially self-ordering convolutional neural networks. In *IEEE ISBI*, pages 1022–2025, 2018. 2

[9] Ke Yan, Xiaosong Wang, Le Lu, and Ronald M Summers. Deeplesion: automated mining of large-scale lesion annotations and universal lesion detection with deep learning. *Journal of medical imaging*, 5(3):036501, 2018. 2