# A Data-Adaptive Loss Function for Incomplete Data and Incremental Learning in Semantic Image Segmentation

Minh H. Vu, Gabriella Norman, Tufve Nyholm, and Tommy Löfstedt

***Abstract*— In the last years, deep learning has dramatically improved the performances in a variety of medical image analysis applications. Among different types of deep learning models, convolutional neural networks have been among the most successful and they have been used in many applications in medical imaging. Training deep convolutional neural networks often requires large amounts of image data to generalize well to new unseen images. It is often time-consuming and expensive to collect large amounts of data in the medical image domain due to expensive imaging systems, and the need for experts to manually make ground truth annotations. A potential problem arises if new structures are added when a decision support system is already deployed and in use. Since the field of radiation therapy is constantly developing, the new structures would also have to be covered by the decision support system. In the present work, we propose a novel loss function to solve multiple problems: imbalanced datasets, partially-labeled data, and incremental learning. The proposed loss function adapts to the available data in order to utilize all available data, even when some have missing annotations. We demonstrate that the proposed loss function also works well in an incremental learning setting, where an existing model is easily adapted to semi-automatically incorporate delineations of new organs when they appear. Experiments on a large in-house dataset show that the proposed method performs on par with baseline models, while greatly reducing the training time and eliminating the hassle of maintaining multiple models in practice.**

***Index Terms*— Medical imaging, CT, missing data, incremental learning, and semantic image segmentation.**

Manuscript received November 4, 2021; accepted December 23, 2021. Date of publication December 29, 2021; date of current version June 1, 2022. This work was supported in part by the Swedish Research Council under Grant 2018-05973; in part by Cancer Research Fund in Northern Sweden, Karin and Krister Olsson, Umeå University, The Västerbotten regional county, and Vinnova, the Swedish innovation agency; and in part by the Cancer Research Foundation in Northern Sweden under Grant AMP 20-1027 and Grant LP 18-2182. *(Minh H. Vu and Gabriella Norman contributed equally to this work.) (Corresponding author: Tommy Löfstedt.)*

Minh H. Vu, Gabriella Norman, and Tufve Nyholm are with the Department of Radiation Sciences, Radiation Physics, Umeå University, 901 87 Umeå, Sweden (e-mail: minh.vu@umu.se; normangabriella@gmail.com; tufve.nyholm@umu.se).

Tommy Löfstedt is with the Department of Computing Science, Umeå University, 901 87 Umeå, Sweden (e-mail: tommy@cs.umu.se).

This article has supplementary downloadable material available at https://doi.org/10.1109/TMI.2021.3139161, provided by the authors.

Digital Object Identifier 10.1109/TMI.2021.3139161

## I. Introduction

CANCER is the second leading cause of death globally and accounted for an estimated 10 million deaths in 2020 [1]. In Sweden, more than 60 000 patients are diagnosed each year, and about half of them undergo radiation therapy [2]. Before the radiation therapy can begin, an oncologist manually marks or delineates, the regions in the body that should be treated (target volumes) and the regions that are particularly important to avoid. The delineations are then used to generate a dose plan that gives a sufficient dose of radiation to the target areas, and a small dose (as small as possible) to sensitive organs, called organs-at-risks (OARs). For that goal to be achieved, the delineations must be correct. Hence, delineation is an essential part of the treatment planning process, but a time-consuming and monotonic manual task for radiation oncologists. Therefore, decision support systems that automate the delineation process would be beneficial in order to reduce the amount of time spent on the challenging task of manually delineating target volumes and organs [3], [4]. In addition, the automatic delineation would also make it possible to delineate more organs at risk. This would in time lead to a better understanding of the relation to dose to certain volumes and side effects of the treatment.

Recently, deep learning (DL) methods, and in particular deep Convolutional Neural Networks (CNNs) have led to breakthroughs in multiple areas of medical imaging. A common application among those is automatic segmentation of organs and structures [5], which—if used to automate all or parts of the delineation stage—would reduce the time spent by radiation oncologists manually delineating images. However, deep neural networks, such as deep CNNs, require large amounts of data; but data is usually challenging and expensive to collect in the medical image domain due to expensive imaging systems, and the requirement to have experts manually annotate ground truth targets or labels [6].

Moreover, since the field of radiation therapy is improving and developing, new organs are sometimes proposed to be added as OARs [7], [8] and therefore new data would be required in order to provide decision support for those newly added OARs as well. Two examples of when the clinical practice can change are from *e.g.* Lee *et al.* [7] who proposed to include the left anterior descending coronary artery region as

an OAR when treating breast cancer and from Roach *et al.* [8] who proposed to include the penile bulb for prostate cancer.

These circumstances can cause projects to spend extensive resources on creating curated datasets, spend numerous hours developing and training deep neural networks, or other machine learning (ML) models, to provide decision support, only to have to redo it when a new OAR has to be included. To add a new OAR to a decision support system requires adding new data or annotations to the dataset and then retraining the ML models from their initial configurations using all the data. A desirable approach would instead be to use the new data created during treatment planning and utilize incremental learning [9], *i.e.* to modify an existing model to semi-automatically incorporate delineations of new organs when they appear.

Another problem related to newly added OARs is that data created during clinical practice is likely incomplete, *e.g.* different patients could have different OARs delineated, and several delineations may therefore be missing for most, if not all patients. As a result, a fully labeled dataset can not be obtained. In other words, the dataset will be partially labeled.

We propose a novel loss function that adapts to the available data, allowing incomplete or missing sets of annotations or labels, and allowing new annotations or labels to be added without retraining, as a solution to the problem of missing delineations. The task investigated in this work was thus automatic segmentation of target volumes and OARs, from now on collectively denoted as *structures*. The proposed loss function then allows the delineations to be inconsistent over the patients in the dataset, to clarify it allows: (i) some patients to only provide information regarding some delineations, (ii) information regarding each delineation to be provided by all or by some patients. Since the proposed loss function allows for incomplete data to begin with, the hypothesis was that adding new OARs during or after having already trained a model would be possible without losing the performance on the already available OARs.

The proposed loss function is used in conjunction with deep CNN models, and works as follows: Available delineations are predicted by the model and contribute to the loss, while unavailable delineations are still predicted by the network, but are excluded from the loss since there are no ground truth delineations to compare with. In the end, the network will be able to predict all considered delineations. See Section II-A for a detailed description of the proposed loss function.

A similar idea was proposed by Liu *et al.* [10], but for predicting brain disease prognosis. Their model predicted 16 clinical scores for magnetic resonance imaging (MRI) data but was trained on data where parts did not have ground truths for all the 16 scores. A label was used to include or exclude the scores from the loss function depending on if they existed or not [10]. Liu *et al.* worked in the regression setting, while this work was independently developed with the classification task in mind (segmentation is classification of each individual pixel); further, the possibility of adding new clinical scores was never investigated by Liu *et al.*, neither was a comparison to semi-supervised learning (SSL) methods.

SSL is an approach to ML that utilizes both labeled data (supervised) as well as unlabeled data (unsupervised) during training. A common contemporary approach in multi-organ segmentation when training on incomplete data is to use SSL [6], [11]. For instance, Zhou *et al.* [11] used 210 labeled cases and 100 unlabeled cases of computed tomography (CT) scans to train and validate an SSL segmentation model. Their model outperformed a fully supervised model by more than a 4 % percentage points increase in terms of the Sørensen-Dice coefficient (DSC). Their SSL model had a *pseudo-labelling model* that was trained on the labeled data and then used to generate *pseudo-labels* for the unlabeled data. It then also had a *semi-supervised model* that was trained on both the labeled and the pseudo-labeled data.

A related SSL approach can be applied to the incomplete data problem in our setting, where patients only have ground truth delineations for some of the OARs. Specifically, one pseudo-labeling model for each delineation would be needed to fill the gaps in the data before training a semi-supervised model to predict all delineations. However, this approach does not allow adding new OARs after the semi-supervised model has been trained. In the case when new OARs are introduced, a pseudo-labeling model would have to be trained for the new OARs to fill the gaps in the historical data, and then the semi-supervised model would have to be retrained from its initial configuration on the labeled and pseudo-labeled data.

Another method, that can be applied in the incomplete data problem, is the Prior-aware Neural Network (PaNN) proposed by Zhou *et al.* [12]. They proposed a partial supervision method with the following steps: First, train a network on a fully-labeled dataset, where all structures have been annotated. Second, estimate a prior label distribution for the masks, based on the fully-labeled data. Third, estimate pseudo-labels for the partially-labeled data. Fourth, update two dual variables that are used to estimate the Kullback-Leibler divergence between the prior label distributions and the predictions. Fifth, updated the network using the fully-labeled data, existing partially-labeled data and computed pseudo-masks. Repeat the third through fifth steps until convergence. This approach aims to (i) minimize the categorical cross-entropy on the fully-labeled data and the partially-labeled data, and (ii) minimize the Kullback-Leibler divergence between the network outputs and the prior label distribution.

In the present work, our main contribution is to introduce the proposed data-adaptive loss function. First, we explored the properties of the data-adaptive loss function by comparing it to individual (single) models for each structure, to the SSL approach [11], and to the PaNN [12] in the segmentation task. Second, we looked at how well a model trained using the data-adaptive loss function can adapt to new OARs being introduced by mimicking the circumstances in a clinical setting, where new OARs are added to an already available decision support system.

The paper is structured as follows. We introduce the methods in Section II, and describe the experiments in Section III. We then present the experimental results and a discussion

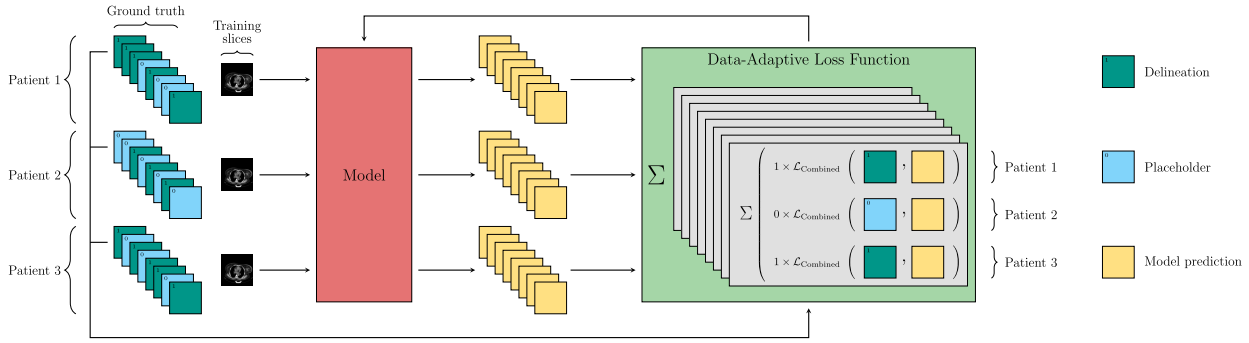


Fig. 1. Illustration of the proposed method including a two-dimensional CNN model together with the proposed data-adaptive loss function in the segmentation task. The model is presented with ground truth delineations (dark green color) and placeholders (light blue color) during training. The placeholders are used here to indicate an image where the ground truth delineations were missing, and hence had weight zero (*i.e.*, had $w_{i,j,k} = 0$).

of them in Section IV and Section V, respectively. Finally, we conclude the paper in Section VI.

## II. Methods

### A. Data-Adaptive Loss Function

The basis for the proposed loss function is a convex combination of the soft DSC loss and the binary cross-entropy (CE) loss for a single structure. The combination of DSC and CE have been successful when training segmentation networks, especially if the structures are unbalanced in size, meaning that there is a (large) disparity in the number of pixels between different segmentation maps [13], [14]. The combined loss function was thus

$$\mathcal{L}_{\text{Combined}}(I_k, \hat{I}_k) = \alpha \cdot \mathcal{L}_{\text{DSC}}(I_k, \hat{I}_k) + (1-\alpha) \cdot \mathcal{L}_{\text{CE}}(I_k, \hat{I}_k), \quad (1)$$

where $\mathcal{L}_{\text{DSC}}$ denotes the soft DSC loss [15], $\mathcal{L}_{\text{CE}}$ denotes the CE loss [16], and $\alpha \in [0, 1]$ is a parameter that determines the trade-off between the two losses. The value of $\alpha$ was found as part of the hyper-parameter search (see Section II-E).

We will denote the ground truth delineations (that are annotated by a radiation oncologist) as *delineations*, the network output predictions (by the single networks and the network using the data-adaptive loss function) as *masks*, and the predictions by the pseudo-labeling models in the SSL (see Section II-C) as *pseudo-masks*.

The function $\mathcal{L}_{\text{DSC}}$ in (1) denotes the soft DSC loss, which is defined as [15], [17]–[20]

$$\mathcal{L}_{\text{DSC}}(I, \hat{I}) = \frac{-2 \sum_l I_l \hat{I}_l + \epsilon}{\sum_l I_l + \sum_l \hat{I}_l + \epsilon}, \quad (2)$$

where the $I$ is the ground truth delineation, the $\hat{I}$ is a predicted mask, the sum is over the pixels in the delineations and masks, and $\epsilon = 1 \cdot 10^{-5}$ is a small constant added to avoid division by zero and to make correctly predicted empty masks have a high DSC score.

Further, the function $\mathcal{L}_{\text{CE}}$ in (1) denotes the CE loss, which is defined as

$$\mathcal{L}_{\text{CE}}(I, \hat{I}) = -\sum_l I_l \cdot \log(\hat{I}_l). \quad (3)$$

Since the patients may have a different set of structures delineated, a data-adaptive loss function has to be able to adapt to the available masks. We propose to only incorporate the masks for which the ground truth delineations exist and normalize the loss appropriately in order to maintain the scale of the loss across different available ground truths. The data-adaptive loss function thus only accounts for the delineations in a given slice that actually exists. Hence, the model would only be updated with regards to the available delineations in any given step.

Let $w_{i,j,k} \in \{0, 1\}$ denote whether or not structure $k = 1, \ldots, K$, with $K = 8$ in this work, is the total number of structures, exists for slice $j$ in patient $i = 1, \ldots, n$. This flag is thus used to indicate whether or not the $k$-th mask was included for a given patient slice, and the contribution of the masks to the loss (using (1)) is an average over the available masks. Similarly, the contribution to the loss of a mini-batch is normalized by the number of a particular ground truth mask that were available for each patient slice in the mini-batch rather than the number of slices in the mini-batch. Hence, only masks that are available will induce a loss for a given patient slice. The proposed data-adaptive loss function for a mini-batch of patient slices is thus,

$$\mathcal{L}(I, \hat{I}) = \frac{\sum_{(i,j)\in\mathcal{I}} \sum_{k=1}^{K} w_{i,j,k} \cdot \mathcal{L}_{\text{Combined}}(I_{i,j,k}, \hat{I}_{i,j,k})}{\sum_{(i,j)\in\mathcal{I}} \sum_{k=1}^{K} w_{i,j,k}}, \quad (4)$$

where $\mathcal{I}$ is a set of patient and slice indices, $(i, j)$, in a mini-batch of delineations, $I$, and masks, $\hat{I}$, and where $I_{i,j}$ contains all delineations for the $j$-th slice of patient $i$, $\hat{I}_{i,j}$ contains all masks for the $j$-th slice of patient $i$, and $I_{i,j,k}$ and $\hat{I}_{i,j,k}$ are the $k$-th delineations and masks for slice $j$ of patient $i$, respectively. Fig. 1 illustrates a 2D convolutional network together with the proposed loss function.

In order to tackle imbalanced datasets (see Section III-A), we also introduce two weighted data-adaptive loss functions that put appropriate emphasis on minority structures. First, the voxel-based weighted loss is defined as,

$$\mathcal{L}_{\mu}(I, \hat{I}) = \frac{\sum_{k=1}^{K} \mu_k \sum_{(i,j)\in\mathcal{I}} w_{i,j,k} \cdot \mathcal{L}_{\text{Combined}}(I_{i,j,k}, \hat{I}_{i,j,k})}{\sum_{k=1}^{K} \mu_k \sum_{(i,j)\in\mathcal{I}} w_{i,j,k}}, \quad (5)$$

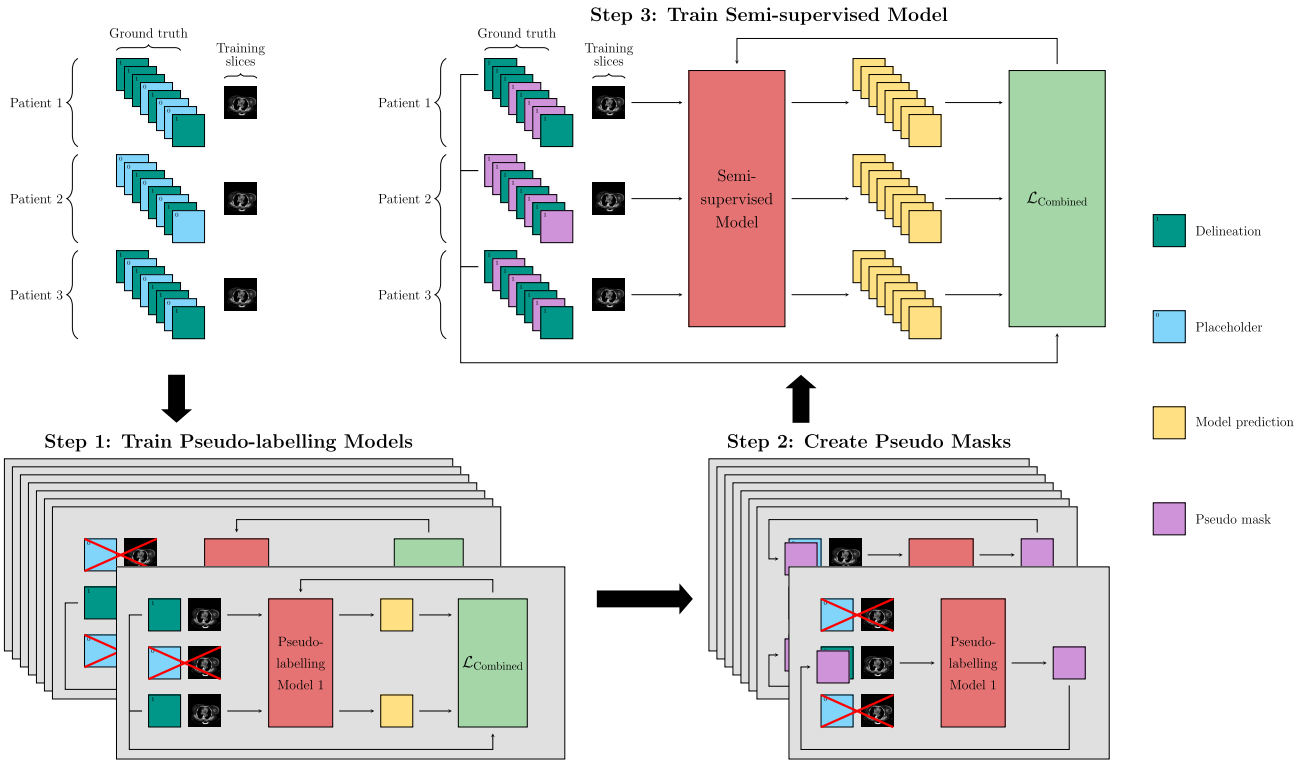


Fig. 2. An illustration of the SSL proposed by Zhou *et al.* [11] that was used in this work (see Section II-C). The SSL model has a *pseudo-labelling model* that is trained on the labeled data and then used to generate pseudo-masks for the unlabelled data. It then also has a *semi-supervised model* that is trained on both the delineations and the pseudo-masks.

where the $\mu_k \in (0, 1]$ are weights for each structure, for $k = 1, \ldots, K$, computed as exponentially weighted averages of the empirical frequencies of each structure. We used bias correction after each update to the exponentially weighted averages.

Second, the slice-based weighted loss is defined as,

$$\mathcal{L}_\gamma(I, \hat{I}) = \frac{\sum_{k=1}^{K} \gamma_k \sum_{(i,j)\in\mathcal{I}} w_{i,j,k} \cdot \mathcal{L}_{\text{Combined}}(I_{i,j,k}, \hat{I}_{i,j,k})}{\sum_{k=1}^{K} \gamma_k \sum_{(i,j)\in\mathcal{I}} w_{i,j,k}}, \tag{6}$$

where $\gamma_k \in (0, 1]$ are weights for each structure, for $k = 1, \ldots, K$, computed as exponentially weighted averages of the positive slices for each structure. Similar to the $\mu_k$, we also utilized bias correction after each update to the exponentially weighted averages.

## B. Baseline Models

To evaluate the proposed data-adaptive loss functions, we first trained eight baseline models, one for each of the eight structures: left breast, right breast, left breast with lymph nodes, right breast with lymph nodes, left lung, right lung, heart, and spinal cord (see Section III-A). Each baseline model had a modified U-Net architecture, based on Ronneberger *et al.* [21], that was altered in the depth, number of filters, and additionally had spatial dropout layers. We used the combined loss function in (1). We further used Bayesian optimization to determine the hyper-parameters for each model (see Section II-E and Table I).

## C. Semi-Supervised Model

The SSL approach used here was inspired by that of Zhou *et al.* [11]. The approach works as follows: A set of *pseudo-labeling models*, one for each structure, was trained on the available delineations. In the present work, these pseudo-labeling models were the same as the baseline models. The pseudo-labeling models were then used to predict the missing delineations, giving a full set of labels containing both the available delineations and the predicted pseudo-masks in the cases when the delineations were missing. Finally, a *semi-supervised model* was trained on all the data, *i.e.* on both the ground truth delineations and on the pseudo-masks. Fig. 2 illustrates the SSL approach that was used in this work.

## D. Prior-Aware Neural Network

The PaNN was proposed by Zhou *et al.* [12] to optimize a stochastic primal-dual gradient algorithm. In that work, the PaNN final loss was defined as

$$\mathcal{L}_{\text{prior}}(I', \hat{I}') = \mathcal{L}_{\text{F}}(I_{\text{F}}, \hat{I}_{\text{F}}) + \lambda_1 \mathcal{L}_{\text{P}}(I', \hat{I}') + \lambda_2 \mathcal{L}_{\text{C}}(I', \hat{I}'), \tag{7}$$

where $I_{\text{F}}$ and $\hat{I}_{\text{F}}$ are the ground truth delineation and predicted mask on the fully-labeled data, respectively; while $I'$ and $\hat{I}'$ are these on the union of the fully-labeled data, the existing partially-labeled data, and the computed pseudo-masks from the partially-labeled data. The $\mathcal{L}_{\text{P}}$ is the categorical cross-entropy loss. Finally, $\mathcal{L}_{\text{C}}$ is the prior-aware loss, the Kullback-Leibler divergence between the network outputs and the prior label distribution, estimated from the fully-labeled data.

TABLE I
THE SEARCH SPACE FOR THE HYPER-PARAMETER SEARCH. SGD DENOTES MINI-BATCH STOCHASTIC GRADIENT DESCENT WITH MOMENTUM AND RMS DENOTES THE RMSPROP OPTIMIZER. THIS TABLE ALSO CONTAINS THE FOUND VALUES FOR EVALUATED METHODS

| Hyper-parameter | Possible values | Left breast | Right breast | Left breast w. ax | Right breast w. ax | Left lung | Right lung | Heart | Spinal cord | Semi super-vised | PaNN | Pro-posed |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Network depth | {3, 4, 5} | 5 | 5 | 3 | 4 | 3 | 5 | 4 | 3 | 5 | 5 | 5 |
| Base filters | {8, 16, 32, 64} | 32 | 32 | 8 | 32 | 16 | 32 | 16 | 32 | 32 | 32 | 32 |
| Spatial dropout rate | [0.0, 0.6] | 0.4541 | 0.1657 | 0.0420 | 0.7004 | 0.1740 | 0.7431 | 0.7074 | 0.6783 | 0.0696 | 0.2211 | 0.3269 |
| Optimizer | {SGD, RMS, Adam} | Adam | SGD | RMS | RMS | Adam | RMS | RMS | RMS | RMS | RMS | RMS |
| Trade-off factor, $\alpha$ | [0.0, 1.0] | 0.9934 | 0.9989 | 0.8871 | 0.8009 | 0.9843 | 0.9133 | 0.6269 | 0.9473 | 0.5264 | 0.5214 | 0.3361 |
| Mini-batch size | [1, 128] | 13 | 27 | 126 | 19 | 33 | 31 | 42 | 8 | 28 | 12 | 28 |
| Learning rate, $10^{\gamma}$ | [-6, -1] | -3.8958 | -2.2390 | -3.9932 | -5.2986 | -3.2798 | -2.7654 | -1.4476 | -2.7624 | -3.1089 | -4.387 | -5.5111 |
| Number of epochs | [10, 120] | 15 | 38 | 72 | 92 | 63 | 25 | 114 | 110 | 68 | 64 | 96 |

Since the dataset here did not have any fully-labeled data, we instead defined the PaNN final loss as

$$\mathcal{L}_{\text{prior}}(I', \hat{I}') = \alpha \mathcal{L}_{\text{P}}(I', \hat{I}') + (1 - \alpha)\mathcal{L}_{\text{C}}(I', \hat{I}'), \quad (8)$$

where $\alpha \in [0, 1]$ again is a parameter that determines the trade-off between two losses, the $\mathcal{L}_{\text{P}}$ and $\mathcal{L}_{\text{C}}$ in this case.

To address the absence of a fully-labeled subset of the data, the empirical distributions were approximated from the available, partially-labeled data.

The implementation details and training of PaNN is presented in Section 1 in the Supplementary Material.

### E. Hyper-Parameter Search

A hyper-parameter is defined as a model parameter that is not immediately updated by the optimization procedure. In CNNs, these would be for instance parameters that determine the network architecture, such as, the number of layers in the network, the number of filters in each layer, or the optimization algorithm used to train the CNNs. Current deep CNNs have a large number of hyper-parameters. There are many procedures proposed to systematically find a good set of hyper-parameters. A commonly employed approach is Bayesian optimization, where expensive to compute black-box functions can be estimated and optimized [22].

To find the hyper-parameters of the CNN models used in the present work, we employed Bayesian optimization to determine the hyper-parameters of each of the previously described models. In particular, we used the Tree-structured Parzen Estimator Approach (TPE) proposed by Bergstra *et al.* [23] through the *hyperopt* library [24] to find a good set of hyper-parameters for each network. The network architecture used in the present work was a modified U-Net model [21]. The hyper-parameter space over which we searched included:

- Network depth: The depth of the modified U-Net.
- Base filters: The number of filters used in the first layer in the modified U-Net.
- Spatial dropout rate: The fraction of the input filters to drop in each step. To reduce overfitting, we added spatial dropout [25] after each max-pooling or concatenation layer.
- Optimizer: The minimization algorithm used.
- Loss trade-off factor, $\alpha$: The parameter controlling the contribution of the DSC and CE losses in the combined loss (see (1)) or of the $\mathcal{L}_{\text{P}}$ and $\mathcal{L}_{\text{C}}$ losses in the prior-aware loss (see (8)).
- Mini-batch size: The number of slices included in each network update step.
- Learning rate: The (initial for Adam) step size used in the gradient descent-based optimization algorithms.
- Number of epochs: The number of times the entire training dataset was presented to the model.

The parameter ranges or values used in the hyper-parameter search are listed in Table I. The hyper-parameter search was performed in a fixed number of iterations, also called the optimization budget, for each model, *i.e.* a fixed number of successful trials for the hyper-parameter search were evaluated (out-of-memory issues were not counted as a successful trial). In the present work, we set the fixed number of iterations to 40.

### F. Incremental Learning

As can be seen in (4) the loss for each structure is computed individually and then summarized and normalized. Therefore, other structures can be added after some partial or full training of a model to accomplish incremental learning.

We conducted experiments on incremental learning by first letting the proposed model *learn* on $K - m$ structures, where $m \in \{1, 2, 3\}$ was the total number of incremented structures. We then extended the existing model's knowledge, *i.e.* further trained the model on the left-out structures, in other words, the incremented structures. The purpose of incremental learning is to adapt a trained model to new data without forgetting its existing knowledge, and hence does not retrain the model from its initial configuration when new data arrives.

In this work, we performed incremental learning in two modes: sequentially and concurrently incremental. In the sequential incremental learning, the existing model was trained by adding one structure after another, while in the concurrent incremental learning, it was trained by adding all left-out structures at once. Fig. 3 illustrates two settings in the incremental learning task.

Each $m \in \{2, 3\}$ (multiple classes) was experimented in both incremental learning modes. In the case $m = 1$ (single-class), both modes are the same. The set of hyper-parameters used in the incremental learning was the ones found in the hyper-parameter search for the model trained on all structures with the proposed loss function (see the "Proposed" column

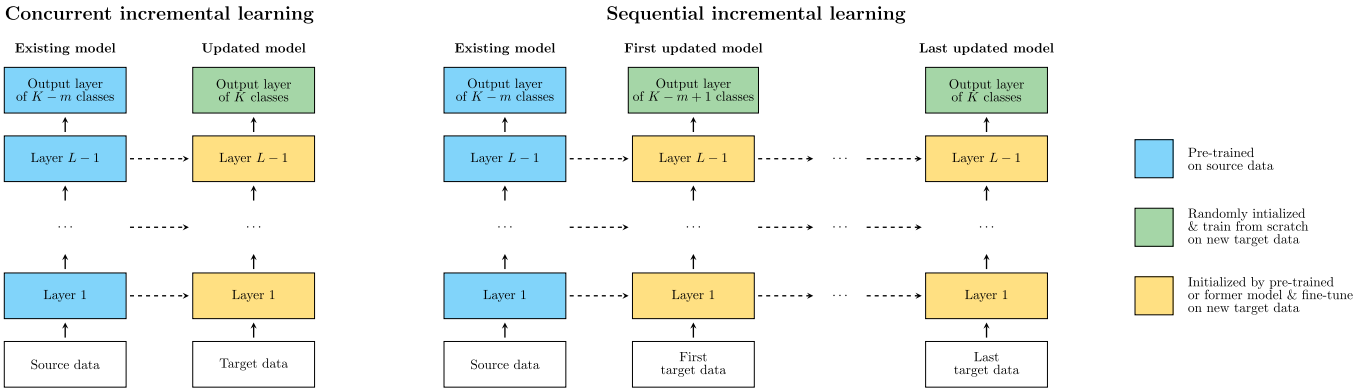


Fig. 3. Illustration of the proposed method in the incremental learning task (see Section II-F). The $K = 8$ and $m = 1, 2, 3$ denote the total number of structures and total number of incremented structures, respectively.

TABLE II

INCREMENTAL LEARNING EXPERIMENT FOR $m = 1,2,3$ LEFT-OUT STRUCTURE(S). A CHECK MARK SIGN, ✓, INDICATES WHETHER A STRUCTURE WAS INCLUDED IN THE EXPERIMENT. EXCEPT IN THE $m = 1$ CASE, THE EXPERIMENTS WERE CONDUCTED IN BOTH THE SEQUENTIAL AND THE CONCURRENT MODE. THE STRUCTURES WERE RANDOMLY SELECTED IN THE $m = 2,3$ CASES

| Structure/Experiment | Increment one structure | | | | | | | | Increment two structures | | | | | | | | Increment three structures | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
| Left breast | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Right breast | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ |
| Left breast w. ax | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ |
| Right breast w. ax | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✓ | ✗ | ✗ |
| Left lung | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ |
| Right lung | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✓ | ✗ | ✗ | ✓ | ✗ |
| Heart | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ |
| Spinal cord | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✗ | ✓ | ✓ | ✗ | ✗ | ✗ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ | ✓ | ✗ | ✓ | ✗ | ✗ | ✓ |

TABLE III

THE RESULTS OF THE NEMENYI POST-HOC TEST COMPARING ALL EVALUATED METHODS. A MINUS (−) MEANS RANKED SIGNIFICANTLY LOWER, A ZERO (0) MEANS NON-SIGNIFICANT DIFFERENCE, AND A PLUS (+) MEANS RANKED SIGNIFICANTLY HIGHER, WHEN COMPARING A METHOD IN THE ROWS TO A METHOD IN THE COLUMNS

| | Single models (II-B) | Semi-supervised [11] | PaNN [12] | Proposed method (4) | Voxel-based (5) | Slice-based (6) |
|---|---|---|---|---|---|---|
| Single models (II-B) | | + | + | 0 | 0 | 0 |
| Semi-supervised [11] | − | | − | − | − | − |
| PaNN [12] | − | + | | − | − | − |
| Proposed method (4) | 0 | + | + | | 0 | 0 |
| Voxel-based (5) | 0 | + | + | 0 | | 0 |
| Slice-based (6) | 0 | + | + | 0 | 0 | |

TABLE IV

THE ORGANS DELINEATED IN THE DATASET, AND THEIR PARTITION INTO TRAINING, VALIDATION, AND TEST DATASETS. FOR THE SPINAL CORD, THE FIRST ROW INDICATES HOW MANY SLICES WERE DELINEATED, AND THE SECOND ROW (MARKED WITH AN ASTERISK, ∗) DENOTES THE TOTAL NUMBER OF SLICES. IT MEANS THAT IF ALL SLICES HAVE DELINEATIONS, THE NUMBERS IN THESE TWO ROWS WOULD THUS BE THE SAME

| Structure | Train # Patients (# Slices) | Validation # Patients (# Slices) | Test # Patients (# Slices) |
|---|---|---|---|
| Left breast | 343 (55 151) | 149 (23 774) | 31 (4 963) |
| Right breast | 366 (57 544) | 154 (24 125) | 35 (5 529) |
| Left breast w. ax | 180 (30 460) | 75 (12 753) | 13 (2 175) |
| Right breast w. ax | 155 (26 107) | 69 (11 794) | 20 (3 468) |
| Left lung | 552 (90 705) | 240 (39 193) | 46 (7 500) |
| Right lung | 541 (86 929) | 233 (37 565) | 57 (9 396) |
| Heart | 541 (86 929) | 240 (39 188) | 45 (7 364) |
| Spinal cord | 472 (69 627) | 202 (30 656) | 46 (6 367) |
| Spinal cord (*) | 472 (81 792) | 202 (34 820) | 46 (7 667) |
| Total | 1 059 (171 463) | 455 (73 745) | 100 (16 386) |

tabulated in Table I). Due to a great number of permutations of two (56 permutations) and three (336 permutations) structures from the eight-structures set, we randomly selected eight permutations from all possible permutations. The selected ones can be seen in Table II.

### G. Statistical Tests

To formally analyze the model performances, we used the Friedman test of equivalence between all evaluated methods on the evaluated metrics using the predictions on the test set. The Friedman test, when reporting significant differences, can be followed by a Nemenyi post-hoc test of pair-wise differences [26].

## III. EXPERIMENTS

### A. Dataset

The data used in this study were collected from 1 614 breast cancer patients at the University Hospital of Umeå, Umeå, Sweden. The data were collected between 2011 and the beginning of 2020 and contained CT images and corresponding delineations of up to eight structures. The number of patient images for each target volume and structure can be found in Table IV. Table IV also contains information about

the number of patients in the training, validation, and test datasets.

During imaging, a wire was sometimes placed around the patient's breast to guide the manual segmentation. This wire was removed during pre-processing by thresholding delineations and selecting the largest structure. The pixel values in the structure were replaced by the mean of the neighborhood pixel values.

The CT slices were down-sampled from $512 \times 512$ pixels to $256 \times 256$ pixels. Pixel values below $-1\,000$ Hounsfield units were set to $-1\,000$. The images were then rescaled by adding $1\,000$, and dividing by $3\,000$; *i.e.*, the voxel values were (approximately) in the zero-one range.

### B. Implementation Details and Training

The models, the proposed loss, and the experiments overall were implemented using Keras 2.2.4[1] with TensorFlow 1.12.0[2] as the backend. The models were trained on NVIDIA Tesla K80 and GeForce RTX 2080 Ti graphics processing units (GPUs). The training time for one hyper-parameter search was between 1–3 weeks.

The U-Net architecture has been a very successful architecture for medical imaging applications, and in particular, for semantic image segmentation [3], [4], [27], and we, therefore, used a modified version of the U-Net in this project. Batch normalization was applied after every convolutional layer.

To speed up the hyper-parameter search, we introduced asynchronous updates (parallel search), allowing multiple trials to be evaluated in parallel. For a fair comparison, the same four GPUs were simultaneously employed in all hyper-parameter search experiments. The objective functions used in the hyper-parameter search were different among the models: the DSC was used for the eight baseline models, while the mean of the DSC of the eight structures was used for the rest.

In the experiment with incremental learning, we used the models trained on $K - m$ structures as starting points for both the sequential and concurrent modes. In the concurrent incremental learning, we then added $m$ output structures and corresponding sets of filters but kept the filters for the other $K - m$ structures that were already trained in the existing model. After that, we updated the existing model with a new set of output(s) by reusing the weights of all layers except for the last layer of the existing model with the purpose of continual learning. The weights for the new structures in the last layer were randomly initialized. For the sequential incremental learning, $m$ models were trained one by one; thus, the last layers of these $m$ models had $K-m+1, \ldots, K$ nodes, respectively, corresponding to $K - m + 1, \ldots, K$ structures. In the last step of both concurrent and sequential incremental learning, we trained the model, that had been updated, on data for all relevant structures.

In the segmentation task, the dimension of the final output of the baseline models was $b \times 256 \times 256 \times 1$, where $b$ is the mini-batch size; while the other methods (SSL, PaNN and proposed method) had a dimension of $b \times 256 \times 256 \times K$. In the incremental learning experiment, the dimensions of the final output of the initial models were $b \times 256 \times 256 \times (K-m)$, while the final model's output dimensions were $b \times 256 \times 256 \times K$.

[1] https://keras.io
[2] https://tensorflow.org

### C. Evaluation

This section describes the evaluation metrics that were used in this study to evaluate the segmentation performance. One of these metrics was the DSC, which is defined as

$$D(I, \hat{I}) = -\mathcal{L}_{\text{DSC}}(I, \hat{I}), \tag{9}$$

with a very small value of $\epsilon$ (*e.g.*, the machine epsilon).

The segmentation performance was also evaluated using the $95^{th}$ percentile of the Hausdorff distance (HD95), a common metric for evaluating segmentation performances. The Hausdorff distance (HD) is defined as

$$H(I, \hat{I}) = \max\left\{d(I, \hat{I}), d(\hat{I}, I)\right\}, \tag{10}$$

where

$$d(I, \hat{I}) = \max_{I_l \in I} \min_{\hat{I}_m \in \hat{I}} \|I_l - \hat{I}_m\|_2, \tag{11}$$

in which $\|I_l - \hat{I}_m\|_2$ is the spatial Euclidean distance between pixels $I_l$ and $\hat{I}_m$ on the boundaries of the delineation $I$ and mask $\hat{I}$.

We also used the relative absolute volume difference (RAVD) between the binary objects in the delineation and the mask. The RAVD is computed as the total volume difference of the delineation to the mask followed by the division by the total volume of the mask. The RAVD is defined as

$$R(I, \hat{I}) = 100 \cdot \frac{\sum_{x \in I} \delta_{x,1} - \sum_{x \in \hat{I}} \delta_{x,1}}{\sum_{x \in \hat{I}} \delta_{x,1}} \tag{12}$$

where $\delta_{x,1}$ denotes the Kronecker delta function, which takes the value 1 if $x = 1$, and 0 otherwise.

The signed RAVD numbers are reported in Table V. A negative value is interpreted as under-segmentation and a positive value as over-segmentation. To obtain a single score value, the absolute value is used. Note that a perfect value of zero can also be obtained for a non-perfect segmentation, as long as the volume of that segmentation is equal to the volume of the ground truth.

Finally, we computed the average symmetric surface distance (ASSD). This metric is closely related to the HD95, but instead of the 95th percentile, it computes the average symmetric surface distance (ASD) between the binary objects in the segmentation and the ground truth.

## IV. Results

Table I contains the eight hyper-parameters chosen for each model by the hyper-parameter search for the eight baseline models, SSL, PaNN and the model with the proposed data-adaptive loss function. We see in Table I that each model ended up having a different set of hyper-parameters. Interestingly, RMSprop tended to be the most favored optimization algorithm.

The Friedman test reported a significant difference between the methods. The results from the Nemenyi post-hoc test are

TABLE V

MEAN DSC (HIGHER IS BETTER), HD95 (LOWER IS BETTER), RAVD (CLOSE TO ZERO IS BETTER), AND ASSD (LOWER IS BETTER) AND THEIR STANDARD ERRORS (SES) (IN PARENTHESES) COMPUTED ON THE TEST SET FROM EIGHT SINGLE MODELS, SSL MODEL, PANN METHOD, AND THE FULL MODELS USING THE PROPOSED LOSS FUNCTIONS. [BLACK TEXT] HIGHLIGHTS THE TOP-PERFORMING METHODS, WHILE [GRAY TEXT] DENOTES THE UNDER-PERFORMING METHODS. NOTE THAT TOP-PERFORMING METHODS ARE ONLY HIGHLIGHTED BASED ON THEIR MEAN VALUES

| Structure | DSC | HD95 | RAVD | ASSD |
|---|---|---|---|---|
| Semi-supervised [11] | | | | |
| Left breast | 0.78 (0.005) | 5.07 (0.203) | 2.70 (0.535) | 1.93 (0.121) |
| Right breast | 0.82 (0.004) | 3.76 (0.101) | 1.27 (0.260) | 1.12 (0.043) |
| Left breast with ax | 0.64 (0.009) | 7.90 (0.285) | 7.71 (1.943) | 3.09 (0.139) |
| Right breast with ax | 0.60 (0.007) | 7.76 (0.247) | 6.73 (1.192) | 3.17 (0.120) |
| Left lung | 0.88 (0.003) | 3.77 (0.135) | 5.98 (0.898) | 1.15 (0.061) |
| Right lung | 0.90 (0.003) | 2.73 (0.089) | 3.02 (0.733) | 0.60 (0.032) |
| Heart | 0.89 (0.003) | 1.80 (0.060) | 0.55 (0.284) | 0.60 (0.023) |
| Spinal cord | 0.70 (0.004) | 1.81 (0.039) | -0.18 (0.012) | 0.59 (0.007) |
| Prior-aware Neural Network [12] | | | | |
| Left breast | 0.85 (0.004) | 3.08 (0.101) | 0.83 (0.264) | 1.10 (0.041) |
| Right breast | 0.83 (0.004) | 4.21 (0.133) | 1.96 (0.445) | 1.70 (0.064) |
| Left breast with ax | 0.79 (0.007) | 4.57 (0.121) | 0.11 (0.101) | 1.41 (0.044) |
| Right breast with ax | 0.82 (0.005) | 4.30 (0.091) | 0.24 (0.124) | 1.41 (0.036) |
| Left lung | 0.86 (0.004) | 3.16 (0.122) | 2.87 (0.462) | 1.11 (0.067) |
| Right lung | 0.94 (0.002) | 2.09 (0.073) | 0.78 (0.307) | 0.43 (0.025) |
| Heart | 0.91 (0.003) | 1.53 (0.055) | 0.43 (0.136) | 0.61 (0.025) |
| Spinal cord | 0.68 (0.004) | 1.18 (0.053) | -0.13 (0.011) | 0.43 (0.025) |
| Single models (II-B) | | | | |
| Left breast | 0.88 (0.004) | 3.66 (0.191) | 0.09 (0.068) | 1.63 (0.126) |
| Right breast | 0.92 (0.003) | 2.09 (0.044) | 0.15 (0.091) | 0.64 (0.014) |
| Left breast w. ax | 0.83 (0.006) | 4.62 (0.164) | 0.17 (0.125) | 1.45 (0.090) |
| Right breast w. ax | 0.84 (0.004) | 4.13 (0.080) | 0.03 (0.032) | 1.30 (0.030) |
| Left lung | 0.97 (0.001) | 1.43 (0.061) | 0.01 (0.006) | 0.29 (0.022) |
| Right lung | 0.97 (0.001) | 1.51 (0.055) | 0.00 (0.006) | 0.23 (0.010) |
| Heart | 0.94 (0.002) | 1.28 (0.046) | 0.19 (0.089) | 0.44 (0.017) |
| Spinal cord | 0.76 (0.004) | 1.05 (0.019) | -0.15 (0.012) | 0.36 (0.007) |
| Proposed method (see (4)) | | | | |
| Left breast | 0.88 (0.004) | 3.65 (0.187) | -0.01 (0.015) | 1.69 (0.130) |
| Right breast | 0.92 (0.003) | 1.98 (0.043) | -0.00 (0.005) | 0.65 (0.015) |
| Left breast w. ax | 0.85 (0.005) | 4.12 (0.110) | 0.14 (0.098) | 1.32 (0.037) |
| Right breast w. ax | 0.86 (0.004) | 4.38 (0.128) | 0.03 (0.009) | 1.21 (0.028) |
| Left lung | 0.97 (0.001) | 1.60 (0.066) | 0.02 (0.016) | 0.28 (0.021) |
| Right lung | 0.97 (0.001) | 1.86 (0.069) | -0.00 (0.005) | 0.21 (0.005) |
| Heart | 0.92 (0.003) | 1.38 (0.046) | -0.04 (0.006) | 0.47 (0.015) |
| Spinal cord | 0.75 (0.004) | 1.06 (0.010) | -0.14 (0.018) | 0.38 (0.004) |
| Proposed voxel-based method (see (5)) | | | | |
| Left breast | 0.87 (0.004) | 3.82 (0.191) | 0.10 (0.148) | 1.71 (0.131) |
| Right breast | 0.91 (0.003) | 2.16 (0.049) | -0.00 (0.010) | 0.65 (0.016) |
| Left breast with ax | 0.86 (0.005) | 4.09 (0.103) | 0.01 (0.036) | 1.25 (0.033) |
| Right breast with ax | 0.86 (0.004) | 3.82 (0.079) | 0.00 (0.045) | 1.14 (0.026) |
| Left lung | 0.97 (0.001) | 2.00 (0.071) | 0.00 (0.006) | 0.29 (0.019) |
| Right lung | 0.97 (0.001) | 1.92 (0.059) | -0.00 (0.004) | 0.24 (0.005) |
| Heart | 0.93 (0.002) | 1.40 (0.046) | -0.00 (0.014) | 0.48 (0.015) |
| Spinal cord | 0.75 (0.004) | 1.06 (0.007) | -0.10 (0.007) | 0.41 (0.004) |
| Proposed slice-based method (see (6)) | | | | |
| Left breast | 0.88 (0.004) | 3.72 (0.189) | 0.09 (0.049) | 1.77 (0.133) |
| Right breast | 0.92 (0.003) | 2.17 (0.051) | 0.10 (0.016) | 0.76 (0.020) |
| Left breast with ax | 0.86 (0.005) | 3.96 (0.104) | 0.14 (0.068) | 1.25 (0.036) |
| Right breast with ax | 0.87 (0.004) | 3.56 (0.073) | 0.07 (0.039) | 1.16 (0.025) |
| Left lung | 0.93 (0.003) | 1.79 (0.070) | -0.02 (0.019) | 0.28 (0.019) |
| Right lung | 0.97 (0.001) | 1.62 (0.053) | -0.01 (0.004) | 0.25 (0.007) |
| Heart | 0.93 (0.002) | 1.33 (0.045) | 0.04 (0.020) | 0.47 (0.017) |
| Spinal cord | 0.75 (0.004) | 1.06 (0.010) | -0.17 (0.005) | 0.39 (0.003) |

tabulated in Table III. The test gives no significant differences between any methods, except for the SSL and PaNN which performed significantly worse than the other methods. Among these worst methods, the PaNN performed better than the SSL.

Except for in Table V, we only report the results on the proposed model without weights (*i.e.*, using (4)), since the differences were non-significant, per the Nemenyi test, between the method without weights and the voxel-based and slice-based methods (*i.e.*, (5) and (6), respectively).

Table V provides the mean DSC, HD95, RAVD, and ASSD and their SEs (in parentheses) computed on the test set. The metrics were computed on (i) the eight single models, that were trained on the eight different structures independently, (ii) the SSL, and (iii) the model using the proposed loss function.

Tabel I in the Supplementary Material shows the mean DSC, HD95, RAVD and ASSD and their SEs (in parentheses) of $K-1$ structures (before) and $K$ structures (after) using incremental learning computed on the test set of eight evaluated structures.

Fig. 4 and Fig. 1 in the Supplementary Material present the learning curves, showing how the DSC changes as a function of the epoch number on the validation set during training. Illustrated in Fig. 4 are the learning curves in the first four, while Fig. 1 in the Supplementary Material are these in the last four sequential and concurrent incremental learning experiments (see Table II). The sequential incremental learning experiments show the DSC *vs.* epoch for $K-m$ structures which were trained from the beginning. Then $m$ additional structures were sequentially added after every 50 epochs starting from epoch 96. The concurrent incremental learning experiments, otherwise, present the learning curves for $K-m$ structures from the beginning of training, and the $m$ structures added at epoch 96 and their development until the end of the training. In both figures, the initial structures are illustrated in black color, while the added structures are illustrated with thicker red/purple/blue lines.

Fig. 5 illustrates the qualitative results of the baseline model, the SSL model, PaNN, and the proposed model on eight structures. Note that the image samples in Fig. 5 were randomly selected, and are mainly for illustrative purposes. However, there are still a few observations that can be made, that are related to the quantitative results in Table V, and are further discussed in Section V.

## V. DISCUSSION

In this section, we compare the proposed model to other recent approaches using all metrics introduced in Section III-C. We then discuss the performance of the task of incremental learning when employing the proposed data-adaptive loss function. Finally, we discuss the qualitative results of the baseline models, the SSL model, and the proposed approach on the segmentation task.

### A. Quantitative Analysis

The Nemenyi post-hoc test (Table III) reveals that the proposed methods performed on par with the baseline models and that the SSL and PaNN methods performed significantly worse than the other methods. Among the worst-performing methods, the PaNN was better than the SSL. From Table V we see that: (i) the performance of the full model with the data-adaptive loss function is comparable to the eight single models in all evaluated metrics while reducing the training time by a factor

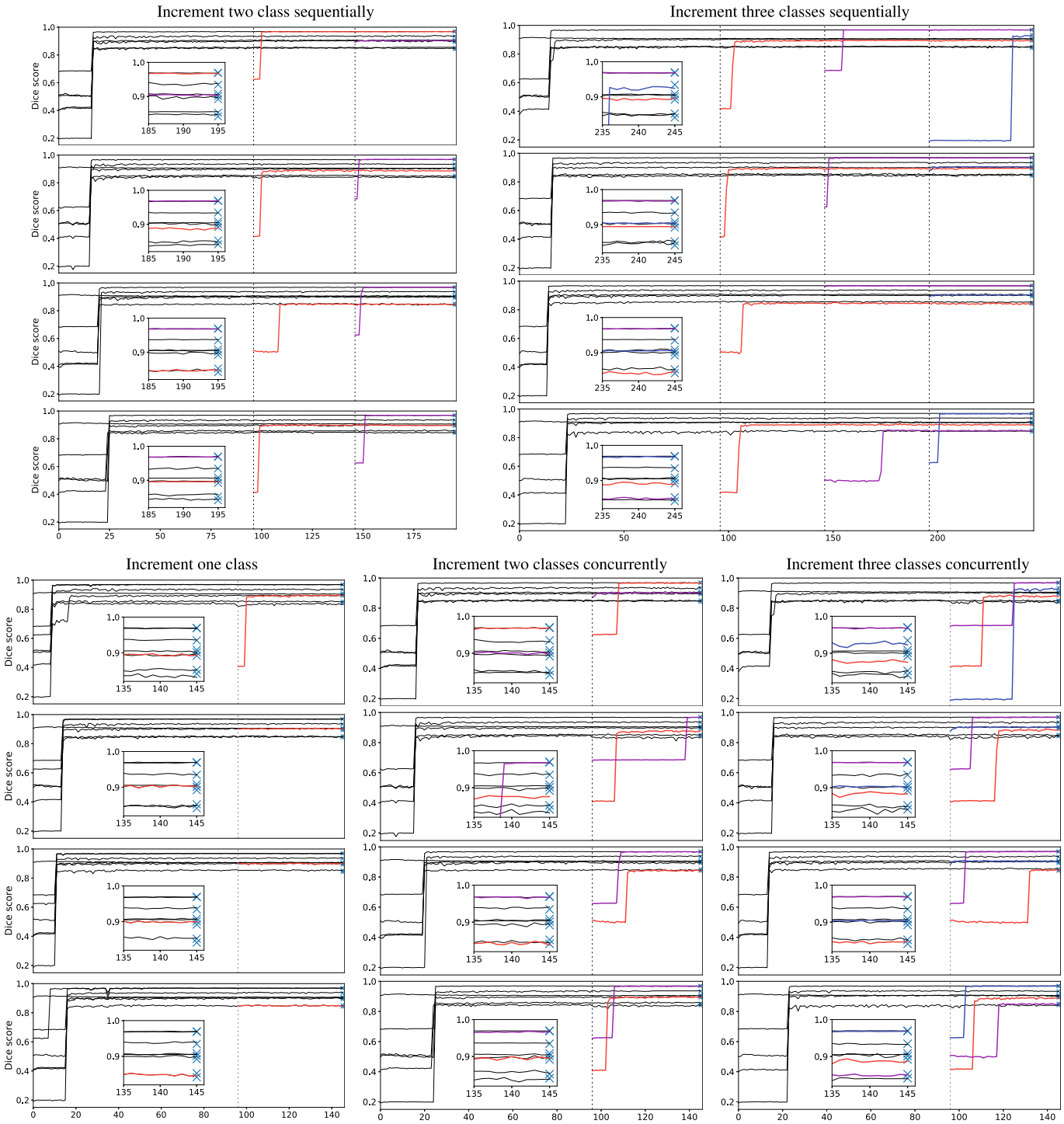


Fig. 4. Illustration of the learning curves showing DSC *vs.* epoch in the first four sequential and concurrent incremental learning (see Table II in the main document). The embedded subplots show a zoomed-in version of that plot of the last ten epochs. In this experiment, $K - m$ structures were trained from the beginning. The $m$ additional structure was added starting at epoch 96 (dashed vertical line) in the concurrent learning experiment, while they were sequentially added starting at epoch 96/146/196 in the sequential learning experiment. The initial structures are shown in black color, while the added structures are displayed with thick red/purple/blue line(s). The light-blue X marks show the DSC of the eight single baselines on the validation set.

of four, (ii) the models with the two proposed voxel-based and slice-based data-adaptive loss functions did not outperform the baseline models, nor the model with the data-adaptive loss function without weights, (iii) PaNN performs better than the top-performing models only in 2/32 categories (see the bold text), and (iv) the proposed method outperformed the SSL and PaNN models by large margins in all the evaluated metrics, with a much shorter training time.

A possible explanation as to why the SSL model underperforms compared to the other methods is that we used the

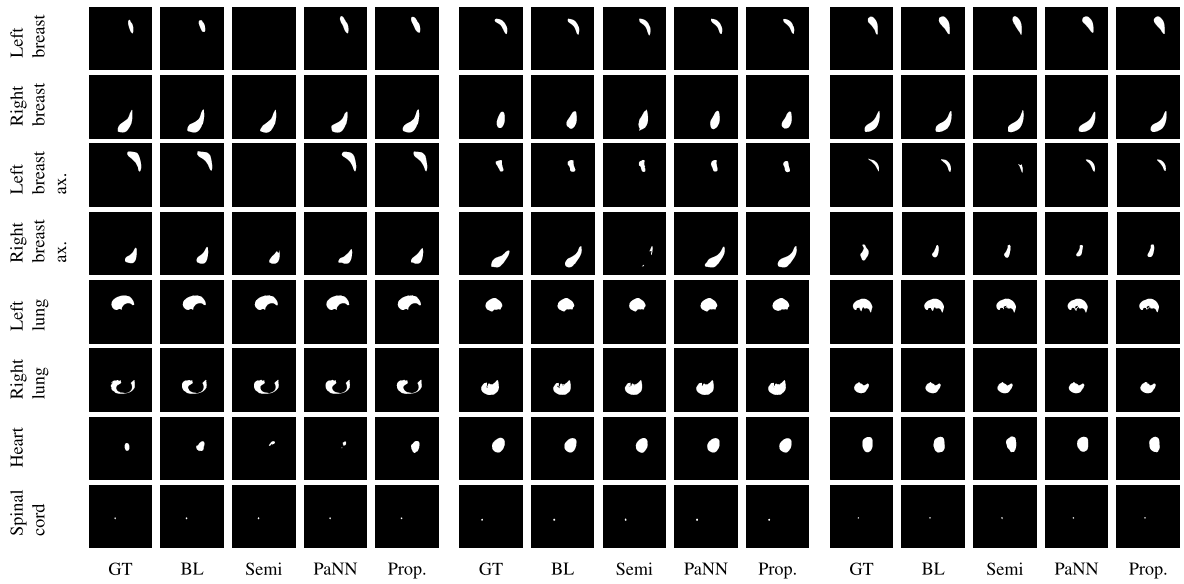


Fig. 5. Qualitative results of the proposed method on eight structures: left breast, right breast, left breast with lymph nodes, right breast with lymph nodes, left lung, right lung, heart, and spinal cord. From left to right: delineation (GT), baseline model (BL), SSL model (Semi), PaNN method (PaNN), and the proposed method (Prop.).

full set of labels containing both the available delineations and the predicted pseudo-masks in the SSL. It thus raised a problem that the pseudo-masks, that were generated by the pseudo-labelling models, might be wrong, making the semi-supervised model solve the wrong problem.

There are several possible explanations for that PaNN performed worse than the baseline models and the proposed methods. First, there is no fully-labeled data in the evaluated dataset; hence, the segmentation model might be improperly initialized, thus making the training procedure unstable, as mentioned in [12]. Second, the evaluated dataset is remarkably imbalanced, which might affect the performance of PaNN. Last, similar to the possible reason for SSL to under-perform, the quality of predicted pseudo-masks might not be good enough, making the segmentation model suffer.

As can be seen in Table V, the DSC and HD95 scores of the spinal cord on all top-performing models tend to be the worst (0.75–0.76) and the best (1.05–1.06), respectively, compared to the other organs. These findings could be explained by the fact that compared to the other structures, the spinal cord occupies a much smaller area. This explanation is supported when comparing the last row of Fig. 5 (small regions) to other rows (large regions). The proposed weighted models were intended to resolve this problem, but it turned out that the performance was unchanged whether or not we used either weighting scheme. This implies that the model with the proposed data-adaptive loss function is not particularly sensitive to the amount of data available for each structure, nor to the size of the structures.

From Table V, we also see that when comparing the performance of the baseline models and the proposed model on the RAVD numbers, the baseline models seem to make over-segmentation on all structures (seven are positive and one is negative), while the proposed method appears to be more balanced (five are negative and three are positive). Another three advantages of the proposed method over the baseline models are that: (i) the proposed method works with incomplete data (ii) in practice, it is much easier to maintain a single model instead of maintaining multiple models, and (iii) the training time for the optimal proposed method was about 3 days, while it took four times longer or about twelve days to train all the baseline models (see Table I).

It can be seen from Tabel I in the Supplementary Material that the evaluated metrics are similar for the $K-1$ initial structures before and after performing incremental learning using the proposed method with the data-adaptive loss function. This means that the knowledge on existing models was retained, and further transferred to the new structure when new training data became available. In addition to that, comparing Tabel I in the Supplementary Material and Table V, we see that the updated models trained on $K$ structures facilitating incremental learning perform on par with the baseline models for the added structures, implying that the models with the data-adaptive loss functions work well in the incremental learning setting.

By looking at Fig. 4 and Fig. 1 in the Supplementary Material, it is interesting to note that in all single-class incremental learning experiments the DSC of the additional structures converged very quickly (after being added in epoch 96), while it took longer when two or three structures were incremented at once. There are two possible explanations for that behavior. First, the existing/initial models of single-class incremental learning experiments had more knowledge (trained on more structures at the initial stage) than these of multi-classes incremental learning. Second, training on a single structure converged faster than multi-classes on the same dataset.

For the sequential incremental learning on multi-classes in Fig. 4 and Fig. 1 in the Supplementary Material, we can see

that the later structures seemed to converge more quickly than the former ones. It is desirable as the later models are "more knowledgeable" than the former ones; thus, it took less time to extend knowledge on new structures. It is important to emphasize that the DSC scores of the existing organs remained unchanged after the additional structures were added in both settings of the incremental learning experiment.

### B. Qualitative Analysis

Predictions in Fig. 5 might not be an accurate representation of the performance of the models since the slices are randomly selected. The baseline models and the proposed method are in many of the examples more similar to the delineations compared to the PaNN model, especially SSL model. One example can be found in the first row where the predictions of the SSL model are empty for both the left breast and the left breast with lymph nodes. Other examples are predictions of the right breast with lymph nodes in the second row as well as in the left breast with lymph nodes in the third row. In the randomly selected single slices in Fig. 5, the SSL model under-predicted more often than the baseline models and the proposed method.

These observations align with the quantitative analysis and the results in Tabel I in the Supplementary Material, where the SSL and PaNN models under-performs compared to the baseline models and the proposed method.

## VI. Conclusion

We have presented a novel data-adaptive loss function for semantic image segmentation. The proposed method has not only been shown to work well when training on incomplete data but also when compared to state-of-the-art SSL and PaNN methods. Furthermore, the proposed method works well in the incremental learning setting, where it is able to learn new structures without forgetting the ones that were already learned. Interesting venues for future work might include, for instance, to determine how rapidly a new structure is learned, or how much data is required to learn a new structure.

## Acknowledgment

The computations were enabled by resources provided by the Swedish National Infrastructure for Computing (SNIC) at the High Performance Computer Center North (HPC2N) in Umeå, Sweden.

## References

[1] H. Sung *et al.*, "Global cancer statistics 2020: GLOBOCAN estimates of incidence and mortality worldwide for 36 cancers in 185 countries," *CA, Cancer J. Clinicians*, vol. 71, no. 3, pp. 209–249, Feb. 2021.

[2] B. Jönsson and N. Wilking, "New cancer drugs in Sweden: Assessment, implementation and access," *J. Cancer Policy*, vol. 2, no. 2, pp. 45–62, Jun. 2014.

[3] Z. Liu *et al.*, "Segmentation of organs-at-risk in cervical cancer CT images with a convolutional neural network," *Phys. Medica*, vol. 69, pp. 184–191, Jan. 2020.

[4] S. Vesal, N. Ravikumar, and A. K. Maier, "A 2D dilated residual U-Net for multi-organ segmentation in thoracic CT," in *Proc. CEUR Workshop*, vol. 2349, 2019, pp. 1–4.

[5] D. Shen, G. Wu, and H. Suk, "Deep learning in medical image analysis," *Annu. Rev. Biomed. Eng.*, vol. 19, pp. 221–248, Jun. 2017.

[6] W. Bai *et al.*, "Semi-supervised learning for network-based cardiac MR image segmentation," in *Medical Image Computing and Computer Assisted Intervention*. Cham, Switzerland: Springer, 2017, pp. 253–260.

[7] J. Lee *et al.*, "Development of delineation for the left anterior descending coronary artery region in left breast cancer radiotherapy: An optimized organ at risk," *Radiotherapy Oncol.*, vol. 122, no. 3, pp. 423–430, Mar. 2017.

[8] M. Roach, J. Nam, G. Gagliardi, I. E. Naqa, J. O. Deasy, and L. B. Marks, "Radiation dose-volume effects and the penile bulb," *Int. J. Radiat. Oncol., Biol., Phys.*, vol. 76, no. 3, pp. 103–134, 2010.

[9] P. Joshi and P. Kulkarni, "Incremental learning: Areas and methods—A survey," *Int. J. Data Mining Knowl. Manage. Process*, vol. 2, no. 5, p. 43, 2012.

[10] M. Liu, J. Zhang, C. Lian, and D. Shen, "Weakly supervised deep learning for brain disease prognosis using MRI and incomplete clinical scores," *IEEE Trans. Cybern.*, vol. 50, no. 7, pp. 3381–3392, Jul. 2020.

[11] Y. Zhou *et al.*, "Semi-supervised 3D abdominal multi-organ segmentation via deep multi-planar co-training," in *Proc. IEEE Winter Conf. Appl. Comput. Vis. (WACV)*, Jan. 2019, pp. 121–140.

[12] Y. Zhou *et al.*, "Prior-aware neural network for partially-supervised multi-organ segmentation," in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 10672–10681.

[13] S. A. Taghanaki *et al.*, "Combo loss: Handling input and output imbalance in multi-organ segmentation," *Comput. Med. Imag. Graph.*, vol. 75, pp. 24–33, Oct. 2019.

[14] K. C. L. Wong, M. Moradi, H. Tang, and T. Syeda-Mahmood, "3D segmentation with exponential logarithmic loss for highly unbalanced object sizes," in *Medical Image Computing and Computer Assisted Intervention*. Cham, Switzerland: Springer, 2018, pp. 612–619.

[15] F. Milletari, N. Navab, and S.-A. Ahmadi, "V-Net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proc. 4th Int. Conf. 3D Vis. (3DV)*, Oct. 2016, pp. 565–571.

[16] R. Y. Rubinstein and D. P. Kroese, *The Cross Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation* (Information Science and Statistics). Berlin, Germany: Springer-Verlag, 2004.

[17] M. H. Vu, G. Grimbergen, T. Nyholm, and T. Löfstedt, "Evaluation of multi-slice inputs to convolutional neural networks for medical image segmentation," *Med. Phys.*, vol. 47, no. 12, pp. 6216–6231, 2020.

[18] M. H. Vu, T. Nyholm, and T. Löfstedt, "TuNet: End-to-end hierarchical brain tumor segmentation using cascaded networks," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke Traumatic Brain Injuries*, vol. 11992. Cham, Switzerland: Springer, 2020, pp. 174–186.

[19] F. Isensee, P. Kickingereder, W. Wick, M. Bendszus, and K. H. Maier-Hein, "No new-net," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke Traumatic Brain Injuries*. Cham, Switzerland: Springer, 2019, pp. 234–244.

[20] M. H. Vu, T. Nyholm, and T. Löfstedt, "Multi-decoder networks with multi-denoising inputs for tumor segmentation," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke Traumatic Brain Injuries*, vol. 12658. Cham, Switzerland: Springer, 2021, pp. 412–423.

[21] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer Assisted Intervention*. Cham, Switzerland: Springer, 2015, pp. 234–241.

[22] J. Snoek, H. Larochelle, and R. P. Adams, "Practical Bayesian optimization of machine learning algorithms," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 2951–2959.

[23] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, 2011, pp. 2546–2554.

[24] J. Bergstra, D. Yamins, and D. D. Cox, "Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures," in *Proc. Int. Conf. Mach. Learn.*, vol. 28, 2013, pp. 174–186.

[25] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proc. Comput. Vis. Pattern Recognit. (CVPR)*, 2015, pp. 648–656.

[26] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, pp. 1–30, Jan. 2006.

[27] O. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning dense volumetric segmentation from sparse annotation," in *Medical Image Computing and Computer Assisted Intervention*. Cham, Switzerland: Springer, 2016, pp. 424–432.