Decoupled Semantic Prototypes enable learning from diverse annotation types for semi-weakly segmentation in expert-driven domains

Simon Reiß 1 Constantin Seibold 1 Alexander Freytag 2 Erik Rodner 3 Rainer Stiefelhagen 1

1 Karlsruhe Institute of Technology 2 Carl Zeiss AG 3 University of Applied Sciences Berlin

{simon.reiss, constantin.seibold, rainer.stiefelhagen}@kit.edu, alexander.freytag@zeiss.com, erik.rodner@htw-berlin.de

Abstract

In this supplementary material, we provide additional details and results to complement and strengthen the findings of the main paper and to make reproducibility as easy as possible. A detailed derivation of how to efficiently implement DSP’s loss term can be found in Section 1. Section 2 gives an explanation for the choice of the segmentation architecture as used for the experiments in the main paper and contains details about the configurations. In Section 3, we provide additional information on dataset sizes and construction of weak labels. Section 4 contains additional details with respect to baseline methods and their ablations. Quantitative results as numerical values are shown in Section 5, thereby complementing the graphical result representations given in the main paper. In addition, Section 6 contains additional qualitative results, with a specific focus on failure cases and a visualization of DSP’s embedding space during training. Finally, we give in Section 7 answers to several questions about DSP.

1. Details on efficiently implementing DSP’s loss term

As shown in the main paper, DSP’s loss term $L_{DSP}$ is built on the idea of decoupled contrastive learning by Yeh et al. [22]. This is especially attractive for segmentation tasks, i.e., with large amounts of pixels, as it removes the necessity to compute different negatives for each positive case. In the following, we will give a more detailed description of how the final loss term as shown in Equation (9) of the main paper can be implemented efficiently (as hinted on in the last part of section 3.5 of the main paper).

Let us start with the loss term $L_{DSP}$ as in Equation (9) of the main paper:

$$L_{DSP} = \sum_{l \in \{m,b,p,im\}} \lambda_l \sum_{c=1}^{C} \sum_{f_i \in \Omega_{lc}} L(f_i, c) \quad . \quad (S.1)$$

Let us look at the last sum in Equation (S.1). For a specific class $c$ and an annotation type $l$, the last sum can be expressed as follows given the definition of $L(f_i, c)$ from Equation (6) of the main paper:

$$\sum_{f_i \in \Omega_{lc}} L(f_i, c) = \sum_{f_i \in \Omega_{lc}} -\log \frac{\exp(s_c(f_i, P_c)/\tau)}{Z_{i,c}} \quad . \quad (S.2)$$

Recall that $Z_{i,c}$, as defined in Equation (8) of the main paper is the denominator of our decoupled contrastive term:

$$Z_{i,c} = \sum_{j=1}^{B \cdot H \cdot W} \sum_{k=1, c \notin \mathcal{A}_j \land k \neq c}^{C} \exp(s_k(f_j, P_k)/\tau) \quad . \quad (S.3)$$

As a reminder from section 3.4: the requirement $c \notin \mathcal{A}_j$ OR $k \neq c$ as given under the second sum is motivated by the design that all associations between pixel-embeddings and prototypes shall be counted as negatives either when i) $k \neq c$ (i.e. a negative association can safely be assumed) or ii) for the case of $k = c$ but then only if $c \notin \mathcal{A}_j$ (i.e. the class $c$ is not possible under the available annotation for pixel $j$).

Due to the these two requirements, $Z_{i,c}$ and hence the denominator in Eq. (S.2) becomes independent of $f_i$. Thus, we write $Z_c$ for short. Simultaneously, we apply the logarithmic division law and simplify the right part of Eq. (S.2):
to obtain:

\[
\sum_{f_i \in \Omega_c^l} L(f_i, c) = \sum_{f_i \in \Omega_c^l} - (\log \exp (s_c(f_i, P_c)/\tau) - \log Z_c) \\
= \sum_{f_i \in \Omega_c^l} - (s_c(f_i, P_c)/\tau - \log Z_c) \\
\]  

(S.4)

At this point, with the independence of \(Z_c\) from \(f_i\), we could simply scale the negatives \(Z_c\) with the number of positives in the set \(\Omega_c^l\) to bring it in front of the summation:

\[
\sum_{f_i \in \Omega_c^l} L(f_i, c) = |\Omega_c^l| \cdot \log Z_c - \sum_{f_i \in \Omega_c^l} s_c(f_i, P_c)/\tau \\
\]  

(S.5)

However, we found that scaling the negatives \(Z_c\) by a value \(>1\) creates large loss magnitudes which were experimentally hard to weight. Hence, we instead re-scale the loss as given in Eq. (S.6) by dividing by \(|\Omega_c^l|\):

\[
L(c) = \log Z_c - \frac{1}{|\Omega_c^l|} \sum_{f_i \in \Omega_c^l} s_c(f_i, P_c)/\tau. \\
\]  

(S.7)

Putting everything together, we obtain the final implementation \(L_{DSP}^*\) for \(L_{DSP}\) as used in our code as follows:

\[
L_{DSP}^* = \sum_{l \in \{m, b, p, im\}} \lambda_l \sum_{c=1}^C L(c) \cdot \delta(|\Omega_c^l| \neq 0) \\
\]  

(S.8)

where the delta function \(\delta(\cdot)\) prevents from division by zero in the case of \(|\Omega_c^l| = 0\) (i.e. when class \(c\) does not occur in the batch). This loss is used for weakly- and strongly augmented inputs.

2. A note on architecture choices

In order to obtain comparable results, all methods in our evaluation share the same segmentation architecture. The results as presented in the main paper have all been obtained with a Unet [16]. To be more precise, we leverage a Unet with successive feature-map channel sizes of \{64, 128, 256, 512, 1024\} in the encoder and the corresponding reverse channel sizes in the decoder. This gives a versatile and yet efficient network with \(\sim 22\) million trainable parameters.

So – why Unets and no recent architectures that give state-of-the-art results on domains like Cityscapes [4]? In preliminary experiments, we indeed explored recent transformer architectures. Specifically, we analyzed some baseline methods on Segformers by Xie et al. [21] and on Swin-Unets by Cao et al. [2]. However, we observed unstable trainings and extremely poor segmentation results, both in cases when trained from scratch as well as when initialized with ADE-20K [24] pre-trained weights. Qualitative results of failed segmentations with these architectures are shown in Figure 1. As can be seen, the segmentation results which we were able to obtain by training these architectures are clearly not acceptable. We assume that one main reason for insufficient applicability of Segformers and Swin-Unets in the task at hand is the lack of pre-trained weights for the domain of electron microscopy imagery. At the same time, we expect that weights obtained from the natural domain do not allow these architectures to transfer sufficiently well to application domains which are starkly different from natural images. Furthermore, we observed training instability of these architectures with respect to hyperparameters and schedulers of optimization parameters, e.g., learning rate warm starts and learning rate decay, which Unets are known to be robust against. Finally, especially Segformers might be negatively impacted from the padding of images for same-sized images in the batch.

Given the previous considerations, and given the requirements for a statistically sound evaluation which requires to robustly train \(>1500\) models for our experiments, we decided for Unets. This choice is further supported in many non object-centric, expert-driven domains, especially in the medical area [7, 9, 10]. At the same time, we expect that improvements like pre-training on in-domain data [13] will enable segmentation transformers also for this domain. For the future, we are especially interested to see how transformers perform when being adapted to and trained with Decoupled Semantic Prototype.

3. Additional information about the datasets

As described in the main paper, we conducted experiments on a total of four cell organelle datasets from

<table>
<thead>
<tr>
<th>Input</th>
<th>Segmentation</th>
<th>Ground-truth</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Image](321x634 to 394x707)</td>
<td>![Image](471x555 to 544x628)</td>
<td>![Image](471x555 to 544x628)</td>
</tr>
<tr>
<td><img src="508x320" alt="Image" /></td>
<td><img src="508x320" alt="Image" /></td>
<td><img src="508x320" alt="Image" /></td>
</tr>
</tbody>
</table>

Figure 1. Results for segmentation transformers trained from scratch with full supervision on HEla-2.
<table>
<thead>
<tr>
<th>Class name</th>
<th>HEKA-2</th>
<th>HEKA-3</th>
<th>MACROPHAGE-2</th>
<th>JURKAT-1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Extracellular Space</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Plasma Membrane</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Mitochondria</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Vesicle</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>MVB</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Lysosome</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Endoplasmic Reticulum</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>Nucleus</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
<tr>
<td>Nucleolar Envelope</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Microtubule</td>
<td>✓</td>
<td>✓</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Cytosol</td>
<td>✓</td>
<td>✓</td>
<td>✓</td>
<td></td>
</tr>
</tbody>
</table>

Figure 2. Top: Organelle classes which are present in at least three sub-volumes in the different datasets of Heinrich et al. [7] for train/validation/test cross-validation are indicated with checkmarks. Bottom: Example images from the different cell organelle electron microscopy datasets used in the main paper.

the OPENORGANELLE data as outlined in Heinrich et al. [7]. In low annotation scenarios, rigorous cross-validation is necessary to get robust insight into method performances. Therefore, we performed cross-sub-volume cross-validation, i.e. we split the set of annotated sub-volumes of the datasets into train/validation/test sets multiple times. As mentioned in the main paper, this was done 10 times for HEKA-2 and 5 times for the remaining datasets.

For creating these splits, we obviously can only include classes that occur in at least three annotated sub-volumes to distribute them among train, validation, and test. Directly applying this constraint to the four cell organelle datasets would leave us with little annotated data. Therefore, as mentioned in the main paper, we merged the original 37 classes into 17 merged classes. For merging classes, we followed the the organelle class hierarchy provided in the code of the original publication\(^1\). In the top part of Fig. 2, we list the subset of classes from these 17 merged classes that satisfy the occurrence requirement in three sub-volumes on the different datasets.

To get an intuition about the difficulty, properties, and differences of these cell organelle datasets, we show in the lower part of Fig. 2 example images from all FIB-SEM datasets used in the experiments. As can clearly be seen, the different datasets show a large variance in appearance as well as aspect ratios (different aspect ratios of labeled regions also occur within datasets).

### 3.1. Dataset sizes

As we outline in the main paper, we performed cross-sub-volume cross-validation either 10 or 5 times. Here, we report the average number of images in the training-, validation-, and testing sets for the four datasets:

- **HELA-2**: 2321 training images, 924 validation images, and 930 testing images
- **HELA-3**: 1634 training images, 731 validation images, and 791 testing images
- **MACROPHAGE-2**: 1482 training images, 685 validation images, and 740 testing images
- **JURKAT-1**: 1525 training images, 745 validation images, and 742 testing images

---

\(^1\) [https://github.com/maalkeila/CNNectome/blob/7c5250edf2ba8ce43127c457b755ea30721f638f/CNNectome/utils/hierarchy.py](https://github.com/maalkeila/CNNectome/blob/7c5250edf2ba8ce43127c457b755ea30721f638f/CNNectome/utils/hierarchy.py)
3.2. How to obtain weak annotations from pixel-wise masks

The OPENORGANELLE dataset by Heinrich et al. [7] comes with pixel-wise annotations. From these, we also derived weak annotations to conduct semi-weakly supervised experiments with diverse annotation types. For creating image-level labels, we simply extracted the unique classes that occur in each pixel-wise mask. To extract bounding boxes, we computed the connected components of each mask, extracted bounding boxes, and assigned the class from the corresponding mask. For creating point annotations from masks, we finally would have several choices. We draw inspiration from the way humans generally point at objects, i.e. clicking on the medial axis [6] or in the center of regions [3]. Hence, we computed the medoids of the connected components of the mask and obtained one point annotation from each medoid location, associated with the class label of the corresponding mask.

4. Additional details for baseline methods and their ablation

Batch construction During training, we ensured for the creation of mini-batches that on average all annotation types used in a training scenario are equally present. This “stratified sampling” was applied for all methods. Thereby, losses based on specific annotation types are computed on a roughly regular schedule rather than after an irregular number of iterations. This is especially useful when the training scenario has severely unbalanced portions of pixel-wise and weak annotations (e.g. at ACR = 64). To obtain this, we oversampled images from less frequent annotation types, similar to the commonly done oversampling of the set of labeled images for semi-supervised segmentation [15, 17].

Pseudo-label [12] We implemented an online version of pseudo-labeling, i.e. computing the labels for un- or weakly labeled images on the fly while training the segmentation network. Although this is in contrast to pre-computing pseudo-labels and then continuing training with fixed pseudo-labels, we decide for the online version as all other compared methods also work in an online setting.

FixMatch [17] The original FixMatch approach was designed for image classification and computes pseudo-labels with a given threshold on the predictions. Thereby, it disregards pseudo-labeled images that have a lower prediction score than this threshold. We investigated the effect of such a threshold when being applied in the segmentation scenario and report results in the left part of Table 1. Surprisingly simply not applying a threshold lead to the best results. Furthermore, FixMatch leverages strong augmentations, including CutOut [5] which randomly sets image regions of size $32 \times 32$ in the input image to black. We analyzed the effect of how often to apply CutOut (i.e. for how often to successively apply CutOut with a probability of 0.5). Results of this ablation study are given in Table 1. We see that the best result is obtained with applying Cutout up to nine times. We also use these strong augmentations for DSP and Con2R [15].

<table>
<thead>
<tr>
<th>threshold</th>
<th>DICE</th>
<th># CutOut</th>
<th>DICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0</td>
<td>51.7 ± 3.6</td>
<td>0</td>
<td>48.5 ± 3.7</td>
</tr>
<tr>
<td>0.1</td>
<td>51.3 ± 4.2</td>
<td>1</td>
<td>49.0 ± 4.1</td>
</tr>
<tr>
<td>0.2</td>
<td>51.2 ± 3.9</td>
<td>2</td>
<td>50.0 ± 3.1</td>
</tr>
<tr>
<td>0.3</td>
<td>51.4 ± 3.5</td>
<td>3</td>
<td>50.3 ± 3.6</td>
</tr>
<tr>
<td>0.4</td>
<td>51.4 ± 4.2</td>
<td>4</td>
<td>50.4 ± 4.0</td>
</tr>
<tr>
<td>0.5</td>
<td>51.0 ± 4.0</td>
<td>5</td>
<td>50.4 ± 4.0</td>
</tr>
<tr>
<td>0.6</td>
<td>51.6 ± 3.9</td>
<td>6</td>
<td>50.4 ± 4.0</td>
</tr>
<tr>
<td>0.7</td>
<td>51.0 ± 3.7</td>
<td>7</td>
<td>50.8 ± 3.3</td>
</tr>
<tr>
<td>0.8</td>
<td>51.0 ± 3.6</td>
<td>8</td>
<td>50.8 ± 4.2</td>
</tr>
<tr>
<td>0.9</td>
<td>50.8 ± 3.6</td>
<td>9</td>
<td>51.7 ± 3.6</td>
</tr>
<tr>
<td>0.95</td>
<td>50.5 ± 3.4</td>
<td>10</td>
<td>51.1 ± 3.4</td>
</tr>
</tbody>
</table>

Table 1. Ablation study for the baseline method FixMatch by Sohn et al. [17] for a standard semi-supervised learning scenario. Validation accuracy measured as DICE score on HELA-2, ACR = 8. Left: Varying the pseudo-label threshold in FixMatch. Right: Varying the maximum number of applying CutOut as strong augmentation.

Classification branch For the classification branch architecture, which is trained on top of the Unet architecture [16], we followed the original publication from Mlynarski et al. [14] as closely as possible. To gauge with the different image size used in the original implementation, we merely needed to add four more convolutions with ReLU activation after the mean pooling operation and single convolution to end up at the desired size of $11 \times 11$. From there, we apply the exact same classification branch architecture consisting of linear layers, ReLU activations, and residual connection.

We also investigated whether the method of Bae et al. [1]
Table 2. Segmentation accuracy of different methods at increasing pixel-wise annotation compression rates, measured as mean and standard deviation in DICE. Results are obtained from 10 splits. Random baseline: 5.1 ± 0.3 DICE. Class-prior baseline: 6.1 ± 0.6 DICE.

<table>
<thead>
<tr>
<th>Method</th>
<th>I</th>
<th>B</th>
<th>P</th>
<th>ACR = 1</th>
<th>ACR = 2</th>
<th>ACR = 4</th>
<th>ACR = 8</th>
<th>ACR = 16</th>
<th>ACR = 32</th>
<th>ACR = 64</th>
</tr>
</thead>
<tbody>
<tr>
<td>UNet</td>
<td></td>
<td></td>
<td></td>
<td>50.1 ± 4.6</td>
<td>50.3 ± 5.6</td>
<td>48.2 ± 4.9</td>
<td>43.6 ± 7.0</td>
<td>34.2 ± 6.6</td>
<td>24.6 ± 3.7</td>
<td>20.2 ± 6.5</td>
</tr>
<tr>
<td>CLS Branch [14]</td>
<td>✓</td>
<td></td>
<td></td>
<td>50.4 ± 5.5</td>
<td>50.3 ± 5.1</td>
<td>50.4 ± 4.6</td>
<td>47.3 ± 5.8</td>
<td>43.8 ± 6.4</td>
<td>34.6 ± 7.3</td>
<td>26.1 ± 8.2</td>
</tr>
<tr>
<td>Box Proj. [18]</td>
<td></td>
<td>✓</td>
<td></td>
<td>48.4 ± 4.9</td>
<td>49.7 ± 5.2</td>
<td>49.9 ± 4.6</td>
<td>47.8 ± 5.9</td>
<td>43.0 ± 5.7</td>
<td>33.5 ± 6.5</td>
<td>26.7 ± 8.5</td>
</tr>
<tr>
<td>Euclidean branch</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>51.3 ± 4.6</td>
<td>50.9 ± 5.2</td>
<td>50.4 ± 5.3</td>
<td>48.7 ± 6.1</td>
<td>41.4 ± 6.9</td>
<td>31.0 ± 5.9</td>
<td>19.9 ± 7.5</td>
</tr>
<tr>
<td>Geodesic branch</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>50.4 ± 4.2</td>
<td>49.6 ± 5.4</td>
<td>50.4 ± 4.0</td>
<td>48.6 ± 6.0</td>
<td>42.2 ± 6.0</td>
<td>33.5 ± 5.8</td>
<td>23.6 ± 6.1</td>
</tr>
<tr>
<td>Pseudo-label [12]</td>
<td></td>
<td>✓</td>
<td>✓</td>
<td>50.1 ± 4.6</td>
<td>50.5 ± 5.5</td>
<td>50.2 ± 4.9</td>
<td>49.3 ± 5.4</td>
<td>46.1 ± 6.2</td>
<td>35.8 ± 6.9</td>
<td>27.9 ± 6.8</td>
</tr>
<tr>
<td>Con2R [15]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>51.6 ± 4.3</td>
<td>52.4 ± 4.9</td>
<td>51.4 ± 5.2</td>
<td>48.9 ± 7.1</td>
<td>43.4 ± 5.7</td>
<td>32.6 ± 5.3</td>
<td>22.1 ± 8.8</td>
</tr>
<tr>
<td>FixMatch [17]</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>52.9 ± 3.9</td>
<td>53.5 ± 4.5</td>
<td>53.6 ± 4.1</td>
<td>53.0 ± 5.1</td>
<td>48.0 ± 6.7</td>
<td>37.8 ± 7.9</td>
<td>22.4 ± 11.7</td>
</tr>
<tr>
<td>DSP (Ours)</td>
<td>✓</td>
<td></td>
<td>✓</td>
<td>53.4 ± 3.6</td>
<td>53.8 ± 5.0</td>
<td>53.5 ± 4.5</td>
<td>52.0 ± 5.6</td>
<td>49.8 ± 6.3</td>
<td>42.4 ± 8.2</td>
<td>32.5 ± 10.4</td>
</tr>
</tbody>
</table>

can give additional benefits, i.e. if restricting the segmentation prediction by the classifier’s output during inference can improve the result. However, we did not find better results in ablation studies. We assume that the initial observation by Bae et al. specifically holds for exceptionally good classifiers as available in object-centric image domains, but does not necessarily generalize to cell organelle images.

**Euclidean/Geodesic point branch** This baseline leverages an auxiliary output-head which regresses distance maps that are generated via point labels. For each class, a distance map is computed based on point annotations, i.e. the smallest distance from every pixel to all point annotations of this class is computed. We implement these distance maps for the Euclidean distance and the Geodesic distance. The choice of Geodesic distance is inspired by point- or click annotations. These are frequently used for medical interactive segmentation [20] and weakly supervised medical segmentation [23], where the Geodesic distance can be exploited. Regressing distance maps is also commonly explored in medical and cell datasets [8, 11]. We visualize an example of such a point-based geodesic map in Figure 3. During training, such Geodesic maps serve as targets to efficiently exploit point annotations and to supply the model with more structural information than singular points alone could offer.

**Box loss** We integrated the bounding box-based loss of Tian et al. [18] for the scenario with pixel-wise annotations and boxes and directly applied it on top of the segmentation output-head for the boxes that are supplied in this scenario.

5. Quantitative results in numerical form

In the main paper, we provided all results as graphical representations by plotting segmentation accuracy against increasing ACRs. The graphical representation aimed at making the progression of accuracy easier to view and better to interpret. For completeness, we provide here underlying numerical results (rounded to one decimal) for all presented experiments.

5.1. Numerical results for all methods

The HELA-2 results for annotation type pairs are displayed in Tab. 2 with checkmarks indicating the weak annotation type used alongside masks. For models trained with all supervision types mixed, results are shown in Tab. 3 for HELA-2, HELA-3, MACROPHAGE-2, and JURKAT-1.

5.2. The benefit of pseudo-label filtering

We further investigated the effect of pseudo-label filtering in the scenario of semi-weakly supervised segmentation. I.e., does the filtering of pseudo-labels based on available weak annotation give benefits for training? For this,
we trained the best semi-supervised learning baseline FixMatch [17] with and without pseudo-label filtering. Results can be found in Tab. 2, first row in FixMatch results for without pseudo-label filtering, and remaining three rows in FixMatch results with pseudo-label filtering. We observe that in the low ACR scenarios, pseudo-label filtering does not give additional benefits as only few weakly labeled examples are integrated. However, especially when training with few pixel-wise masks, i.e. ACRs ∈ {16, 32, 64} we see a tremendous positive effect. Hence, we conclude that FixMatch coupled with pseudo-label filtering is a simple and strong semi-weakly supervised baseline.

6. Supplementary qualitative results

6.1. Segmentation results

In the top three rows of Fig. 4, we provide additional qualitative results from DSP on an image from the HEALA-2 dataset. As can be seen, DSP is able to uncover small organelle structures and start capturing their semantics even with very few pixel-wise annotated masks in training and a mix of annotation types.

In the middle four rows of Figure 4, we show the qualitative segmentation results of DSP when being trained with different combinations of annotation types (all scenarios from the main paper). We find two observations worth to note. First, we see point annotations allow to quickly learn to find central regions which lie on the organelles. Second, we also see that training with bounding boxes leads to models which slightly oversegment the organelles. These observations combined can serve as explanation why the combination of these annotation types (in the mixed scenario) works better than each type individually.

Finally, we provide a failure case where DSP leads to an insufficient segmentation in the last row of Fig. 4. As can be seen, DSP is not able to clearly delineate the nuclear envelope (dark blue) and fails in correctly assigning the class microtubule (purple). We expect these errors to originate from a too strong learned shape bias for the microtubules.

6.2. Evolution of Decoupled Semantic Prototypes

Finally, we were interested in a deeper understanding about the semantic prototype vectors of DSP. To this end, we plotted t-SNE projections [19] of class-wise prototypes \( P_i \) alongside several few pixel-embeddings \( f_i \) after the 10th, 50th and 100th epoch. These are shown in Fig. 5. It can be seen that over the time of training, clusters are formed and the prototypes delimit which semantics occupy what regions of the embedding space.

7. FAQ – Good questions and honest answers

During the reviewing process and in internal discussions, several valid and insightful questions have been raised. We list some of them together with honest answers here, as we
<table>
<thead>
<tr>
<th>Input</th>
<th>Ground-truth</th>
<th>$ACR = 2$</th>
<th>4</th>
<th>8</th>
<th>16</th>
<th>32</th>
<th>64</th>
</tr>
</thead>
<tbody>
<tr>
<td>![Unet][16]</td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
<td><img src="Image" alt="Unet" /></td>
</tr>
<tr>
<td>![FixMatch][17]</td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
<td><img src="Image" alt="FixMatch" /></td>
</tr>
<tr>
<td>DSP (Ours)</td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
</tr>
<tr>
<td>DSP (Z)</td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
</tr>
<tr>
<td>DSP (P)</td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
</tr>
<tr>
<td>DSP (B)</td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
</tr>
<tr>
<td>DSP (Mixed)</td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
<td><img src="Image" alt="DSP" /></td>
</tr>
</tbody>
</table>

Figure 4. From left to right: test image, ground-truth segmentation of the organelle structures, and predicted segmentations from models trained with diverse annotation types and decreasing number of annotated pixel-wise masks on *HELA-2*. Top three rows: Models trained in the mixed annotation type scenario as outlined in the main paper. Middle four rows: Our method *Decoupled Semantic Prototypes* trained with different combinations of annotation types to show their effects. Bottom row: Failure case of DSP in the mixed scenario.
believe that they are relevant to a broader audience.

**Q: Does $L_{DSP}$ use unlabeled images?** $L_{DSP}$ is mainly based on annotations, specifically it unifies arbitrary annotation-types. Still, from unlabeled images, prototype associations that do not belong to the positive class can be derived. You can see this by closely observing the final loss $L_{DSP}$ in Eq. (9): although the sum goes only over annotation types $l \in \{m, b, p, im\}$, also unlabeled images are contributing via negative associations in $L(f_i, c)$ specifically, are included in computing $Z_{i,c}$ (Eq. (8): criteria $k \neq c$ is fulfilled). Furthermore, unlabeled images are also used in the filtered pseudo-labels via $L_{PLF}$.

**Q: What is the difference of base model with a linear projection + cross-entropy compared to your DSP architecture?** Our proposed prototypes are a matrix of shape $D \times C \times |PC|$. From that point of view, a plain projection head of size $D \times C \times |PC|$ can indeed be used to produce $1 \times C|PC|$ scores. However, while standard projection heads rely on dot product, we exploit the cosine distance in Eq. (1) which uses normalized vectors to bound the similarity scores (as typically done in contrastive learning [22]). Furthermore, a common projection head design would directly predict $C$ class scores, rather than predicting $C|PC|$ scores followed by averaging the $|PC|$ values per class as done in DSP.

With respect to results: on HELA-2, all splits, ACR=8, a plain Unet achieves $43.6 \pm 7.0\%$ (this is the special case of projection head and $|PC| = 1$). The implementation of a projection head with $|PC|$ scores per class, class-wise averaging, and trained with $L_{CE}$ results in $45.1 \pm 6.6\%$. The same projection head with cosine similarity instead of dot product but still only trained with $L_{CE}$ leads to $48.2 \pm 5.7\%$. Finally, DSP is able to integrate all annotation-types via $L_{DSP}$ and reaches $54.2 \pm 4.6\%$.

**Q: Isn’t the computation for training quite expensive?** In fact, all approaches in our experiments had access to the same resources available to us and are therefore comparable. Furthermore, since hardware is scalable via a higher budget, we investigated the limited resource ‘expert-availability’ which can’t be scaled easily.

**Q: Would post processing give better performance?** Absolutely, yes. We clearly acknowledge that all approaches would benefit from smart post-processing rules. Here, we focused on a plain comparison of underlying training algorithms.

**Q: Since ACR does not reflect varying efforts for weak annotations, is ACR a good measure for expert-centric scenarios?** We think it is! The idea of expert-centrism is that a segmentation training is not restricted to masks but can handle any preference of annotation combinations (as DSP does).

**Q: Why don’t you additionally compare with purely weakly supervised methods?** A comparison to weakly supervised methods would give nice lower bounds but would not be a fair comparison as they don’t consider mask annotations.

**Q: Why don’t you additionally compare on standard computer vision datasets?** We agree: more datasets are always better (and we’re curious too!). Yet, training the over 1600 networks on the four datasets was already a considerable effort given our available compute hardware.
References


