A Hyperparameters for self-training performance at fixed n_l

This section provides a detailed list of the chosen hyperparameters foreach dataset. Self-training hyperparameters can be found in Table 1. The other hyperparameters are listed in Table 2.

Table 1: Training hyperparameters used for the experiments of Section 6.1.

Dataset	n_l	iter/epoch	tile size	W	E
MoNuSeg	2	100	512	10	50
SegPC-2021	30	300	512	10	50
GlaS	8	225	384	10	50

Table 2: Self-training hyperparameters used for the experiments of Section 6.1 for MoNuSeg and SegPC-2021. The same hyperparameters have been used for GlaS except for the combination "*constant*" and $C = 0.2$.

Weight	C	w_{min}	η	Datasets M	S	G
constant	0.01			✓	✓	
constant	0.5			✓	✓	✓
constant	1.0			✓	✓	✓
constant	2.0				✓	
entropy		0.1		✓	✓	✓
consistency			2	✓	✓	✓
merged		0.1	2	✓	✓	✓

B Weighting strategies

This section provides more details about our weighting strategies.

Constant. This strategy consists in setting $w_{ij}^{\text{cst}} = C$ where $C \in \mathbb{R}^+$ is an hyperparameter. Because $w_{ij} = 1$ for ground truth pixels, this allows to manually balance the relative contributions of ground truth and pseudo-labeled pixels. The special case $C = 1$ assigns the same weight to ground truth and pseudo-labels and therefore corresponds to removing $w_{ij,b}$ from Equation 6 in the main article.

Entropy. Unlike the previous, this strategy is not concerned with balancing the contributions but instead penalizes pseudo-labels for which the model was uncertain. It considers the prediction $\hat{y}_{ij}$ as a probability and tune the contribution

down using the Shannon entropy. The use of entropy is motivated by its use in several self-training methods [1,2]. First, an intermediate weight ω_{ij} is computed as:

$$\omega_{ij} = 1 + \hat{y}_{ij} \log_2(\hat{y}_{ij}) + (1 - \hat{y}_{ij}) \log_2(1 - \hat{y}_{ij}). \tag{1}$$

Early experiments have shown that directly using ω_{ij} as a weight resulted in unstable training. Indeed, during early self-training rounds, the model typically produces $\hat{y}_{ij} \sim 0.5$ which results in $\omega_{ij} \sim 0$ for most pixels in a patch, leaving only foreground ground truth pixels to be evaluated in the loss. In order to avoid this behavior, we introduce a new hyperparameter $w_{min} \in]0, 1]$ which allows rescaling linearly the weights ω_{ij} to $w_{ij}^{ent} \in [w_{min}, 1]$ as defined in:

$$w_{ij}^{ent} = (1 - w_{\min})\, \omega_{ij} + w_{\min}. \tag{2}$$

Consistency. Self-training algorithms often enforce consistency between the teacher and student models predictions. Inspired from this, we exploit another form of consistency for this strategy. In structured output tasks like segmentation, there is a correlation between predictions that are spatially close, as, for most pixels, it is unlikely that the true label should differ between a pixel and its neighbours. Therefore, we use the pseudo-label consistency between a pixel and its neighbours as a proxy to evaluate reliability of this pseudo-label:

$$w_{ij}^{\text{cty}} = 1 - \frac{\sum_{k=-\eta}^{\eta} \sum_{l=-\eta}^{\eta} (\hat{y}_{ij} - \hat{y}_{(i+k)(j+l)})^2}{\eta^2 - 1} \tag{3}$$

where η is the size of the neighbourhood and an hyperparameter of the method. We consider a square neighbourhood around the central pixel and ignore pixels outside of the image at the image borders. We have arbitrarly chosen $\eta = 2$ as default value for all our experiments involving the "*consistency*" weighting strategy.

Merged. This strategy assigns a high weight to pixels for which the model is both certain and consistent (spatially). It achieves this by multiplying together the consistency weight w_{ij}^{cty} and the entropy raw weight ω_{ij}. Because ω_{ij} suffers from the issue described earlier, we apply the same re-scaling operation after multiplication:

$$w_{ij}^{\text{mgd}} = (1 - w_{min}) \left(w_{ij}^{\text{cty}} \times \omega_{ij} \right) + w_{min}. \tag{4}$$

C Additional weighting strategies evaluated for the fixed n_l experiment

This section reports performance for all the weighting schemes actually evaluated (see Figures 1, 2 and 3) for the fixed n_l experiment.

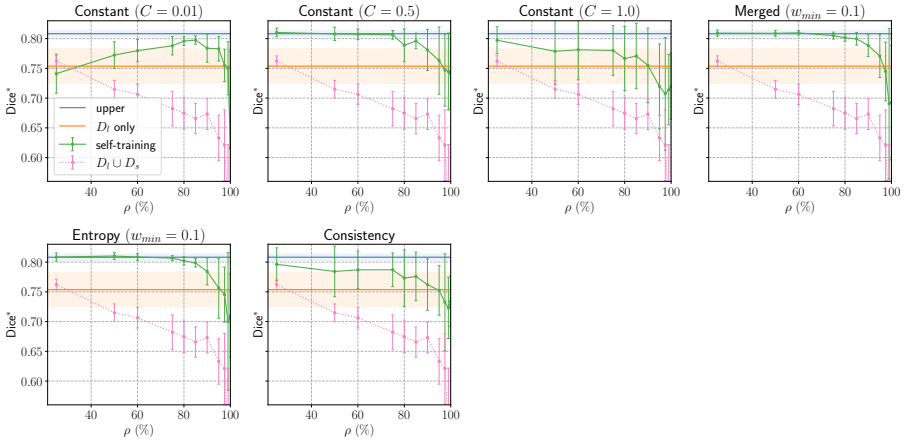


Fig. 1: MoNuSeg , see Figure 2 (main article) for explanation.

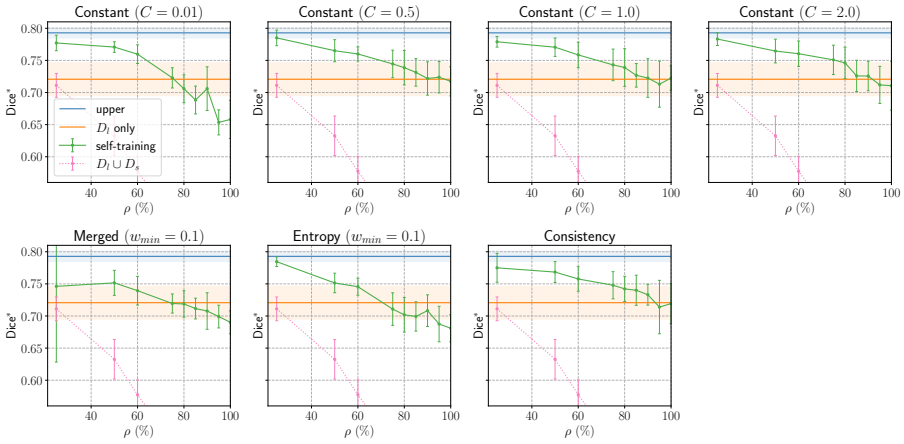


Fig. 2: SegPC-2021 , see Figure 2 (main article) for explanation.

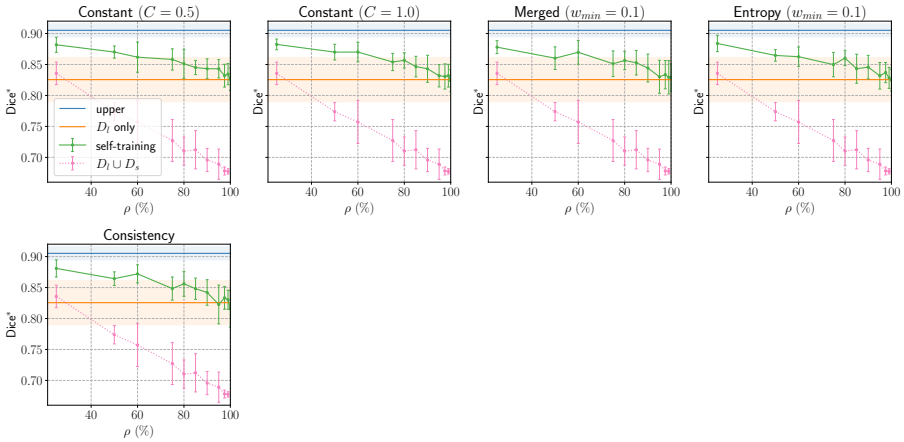


Fig. 3: GlaS , see Figure 2 (main article) for explanation.

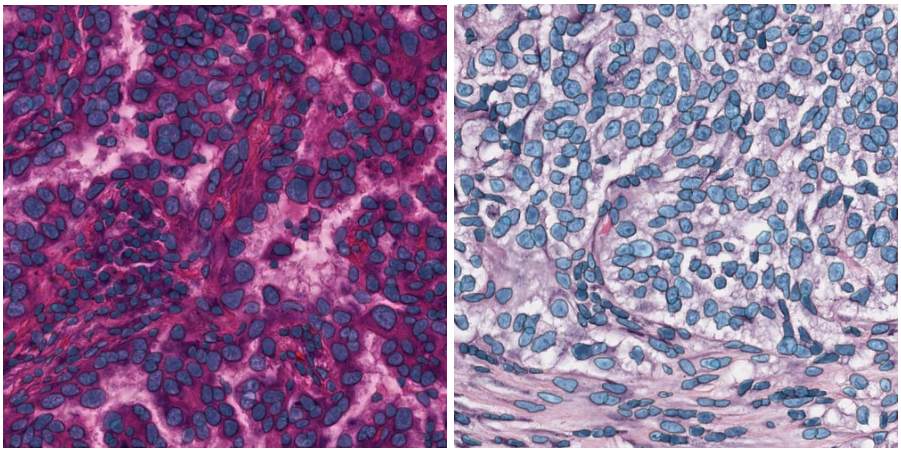
D Samples from the public datasets

In this section, we illustrate the public datasets we have used to support our experiments: MoNuSeg (see Figures 4a and 4b), SegPC-2021 (see Figures 4c and 4d) and GlaS (see Figures 4e and 4f).

E About splitting Thyroid FNAB training set into $\mathcal{D}_l$ and $\mathcal{D}_s$

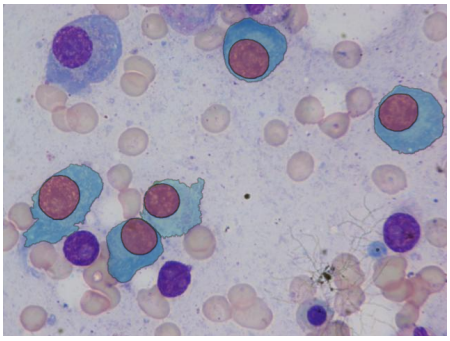
The Thyroid FNAB dataset was labeled by experienced pathologists from Isabelle Salmon's team from Erasme Hospital (Brussels, Belgium) who followed a detailed ontology to categorize their annotations:

1. Architectural patterns (see examples in Figure 5):
   - Normal follicular architectural pattern
   - Proliferative follicular architectural pattern
   - Proliferative follicular architectural pattern (minor sign)
2. Nuclear features (see examples in Figure 6):
   - Papillary cell NOS
   - Normal follicular cells
   - Normal follicular cell with pseudo-inclusion (artefact)
   - Papillary cell with ground glass nuclei
   - Papillary cell with nuclear grooves
   - Papillary cell with inclusion
3. Others:
   - Macrophages
   - Red blood cells

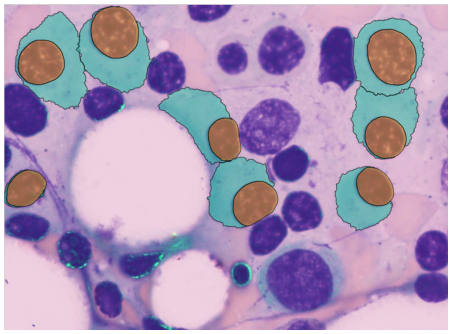


(a) MoNuSeg

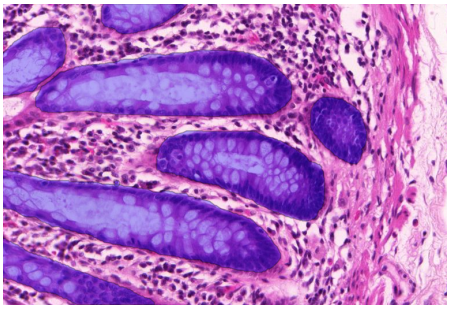
(b) MoNuSeg



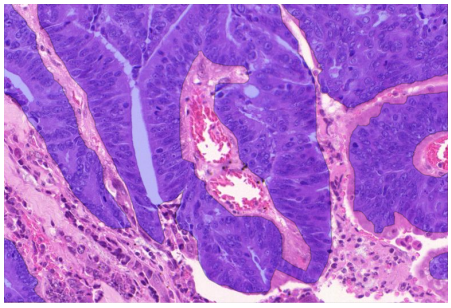
(c) SegPC-2021



(d) SegPC-2021



(e) GlaS

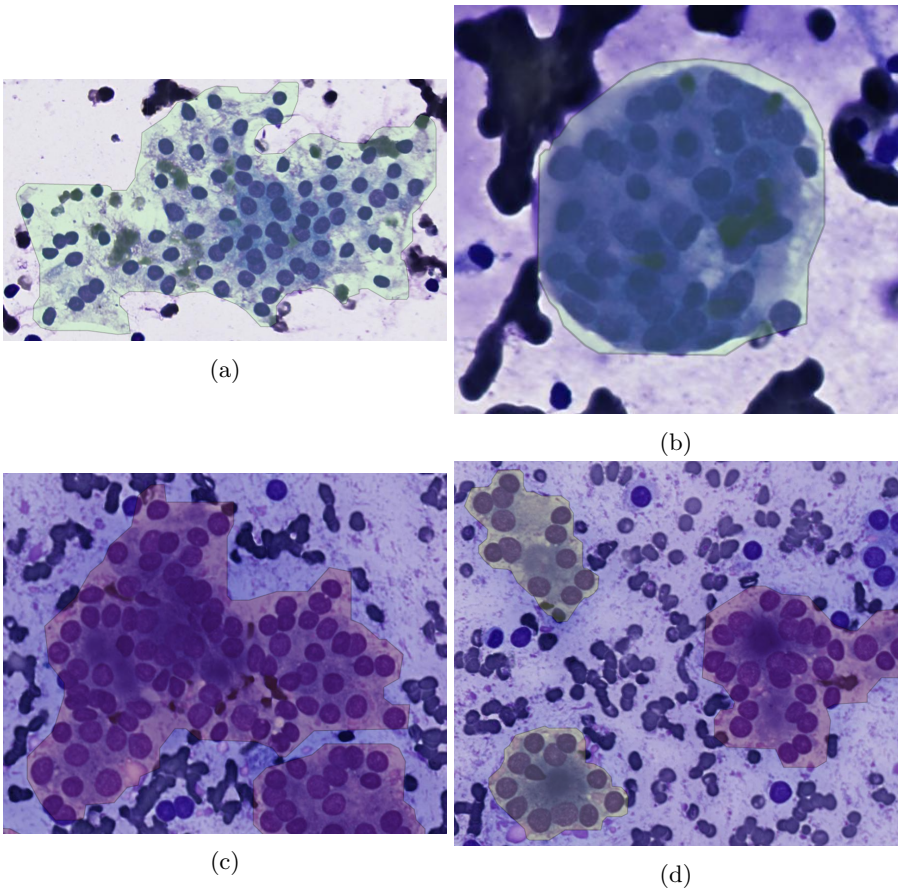


(f) GlaS

Fig. 4: Samples from MoNuSeg, GlaS and SegPC-2021 used in this work.

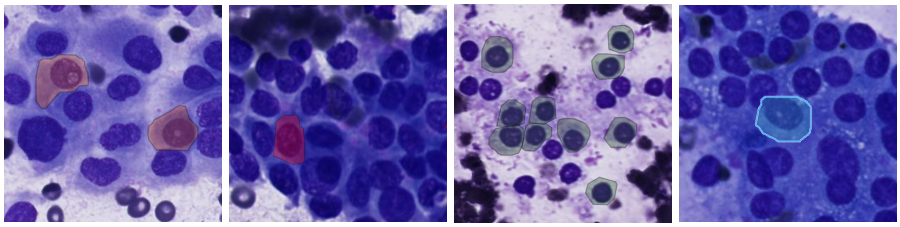
- PN (polynuclear)
- Colloid
- Artefacts
- Background

Given how the labeling process was carried out, we hypothesize that crops of architectural patterns are less likely to contain unlabeled cells than crops of nuclear features. Indeed, the former usually consist in large polygons delineating areas containing cell aggregates. Nuclear features, unlike architectural patterns, were usually labeled more sparsely and it is frequent to find annotations of a single cell within an unlabeled cell aggregate. This can be seen in Figures 5 and 6. These observations and hypothesis motivate the assignment of architectural patterns to $\mathcal{D}_l$ and nuclear features to $\mathcal{D}_s$.



(a) (b) (c) (d)

Fig. 5: Examples of architectural patterns annotations made by pathologists for Thyroid FNAB.



(a) (b) (c) (d)

Fig. 6: Examples of nuclear features annotations made by pathologists for Thyroid FNAB.

References

1. Grandvalet, Y., Bengio, Y.: Semi-supervised learning by entropy minimization. Advances in neural information processing systems **17** (2004)
2. Lee, D.H., et al.: Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In: Workshop on challenges in representation learning, ICML. vol. 3, p. 896 (2013)