# Weakly Supervised Few-Shot Segmentation Via Meta-Learning

Pedro H. T. Gama, Hugo Oliveira, José Marcato Junior, Jefersson A. dos Santos

*Abstract*—Semantic segmentation is a classic computer vision task with multiple applications, which includes medical and remote sensing image analysis. Despite recent advances with deep-based approaches, labeling samples (pixels) for training models is laborious and, in some cases, unfeasible. In this paper, we present two novel meta learning methods, named WeaSeL and ProtoSeg, for the few-shot semantic segmentation task with sparse annotations. We conducted extensive evaluation of the proposed methods in different applications (12 datasets) in medical imaging and agricultural remote sensing, which are very distinct fields of knowledge and usually subject to data scarcity. The results demonstrated the potential of our method, achieving suitable results for segmenting both coffee/orange crops and anatomical parts of the human body in comparison with full dense annotation.

*Index Terms*—Semantic Segmentation; Few-Shot; Meta Learning; Weakly Supervised; Agriculture; Remote Sensing; Medical Imaging Analysis.

## I. INTRODUCTION

Image segmentation is a classical computer vision problem where, given an image, a model is required to assign a class to every pixel, defining fine boundaries to the objects of interest that compose the image. It has applications in many scenarios, including medical image analysis [25, 35], remote sensing [15, 20], and others. State-of-the-art approaches to segmentation mostly use Deep Neural Networks (DNNs) methods, especially variations of Convolutional Neural Networks (CNNs). These approaches became popular after the work [17] and the advances of Graphical Processing Units (GPUs) that allowed the training of large complex models. The main limitation of current state-of-the-art deep models is the reliance on a large annotated training set, hampering the use of such models in more specific real-world scenarios out of the mainstream visual learning tasks. It is common for DNNs to present *underfitting* or *overfitting* [11] problems when trained with a limited amount of data samples.

Common semantic segmentation methods rely on labels for all pixels in an image. From now on, this annotation strategy will be referred to as *full/dense annotation*, being characterized by the highly laborious process required for producing such ground truths. The expensive process for producing *dense annotations* is further aggravated in certain scenarios such

Pedro Gama and Jefersson A. dos Santos are with the Department of Computer Science, Universidade Federal de Minas Gerais, Belo Horizonte, Brazil; phtg@dcc.ufmg.br

Hugo Oliveira is with the Institute of Mathematics and Statistics (IME), at University of São Paulo (USP), São Paulo, Brazil.

José Marcato Junior is with the Faculty of Engineering, Architecture and Urbanism and Geography, Federal University of Mato Grosso do Sul, Campo Grande, Brazil.

as medical imaging or remote sensing, where usually only specialists are able to produce labels correctly. Thus *sparse annotations* becomes an interesting solution, as they consist in only presenting a label for a small set of pixels of the image. This type of annotation reduces the time required to produce the labels for an image but, it can be challenging to train a model with such limitations on the amount of information available. Multiple methods [18, 33, 35] have successfully used sparse labels for image segmentation.

Another strategy to reduce the cost of labeling datasets is to reduce the total number of images in it, and consequently the number of labeled images. Such scenarios with small dataset sizes are commonly known as few-shot and have recently gained the interest of the computer vision community. The few-shot learning literature contains a vast amount of works focused on image classification with notable examples [9, 23, 29, 34], although some methods for semantic segmentation have been proposed in recent years [7, 13, 24, 36, 38].

One methodology that has been successfully applied to few-shot problems in recent years is the meta-learning framework [9, 29, 34]. Normally understood as *learning to learn*, meta-learning is an umbrella term for a collection of methods that improve the generalization of a learning algorithm through multiple multi-task learning episodes. A recent survey [12] formalizes the meta-learning framework and proposes different forms to categorize methods that use this approach. One can summarize meta-learning methods as an algorithm that learns a set of parameters $\omega$ called *meta-knowledge*, trained using a distribution of tasks, such that $\omega$ generalizes well for the tasks in said distribution. The way a model achieves the training of the *meta-knowledge* is used to group meta-learning methods into categories.

In this work, we extensively evaluated our previously proposed method WeaSeL [10] in a vast array of scenarios. Additionally, we introduced a fully novel semantic segmentation method (ProtoSeg), to problems with few-shot sparse annotated images. These two approaches are based on the meta-learning algorithms of Model-Agnostic Meta-Learning (MAML) [9], and Prototypical Networks (ProtoNets) [29], respectively.

The main contributions of this work are: (1) A novel meta-learning method for the problem of semantic segmentation with few-shot sparse annotated images; (2) An extensive evaluation of our previous and new proposals in a large collection of tasks from Medical and Remote Sensing scenarios; (3) A comparative analysis of five styles of sparse annotations named: Points, Grid, Contours, Skeletons, and Regions; and (4) Two novel publicly available crop segmentation datasets

with semantic labels for coffee and orange orchard crop regions. The coffee crop dataset has been previously published in previous works [8, 22], but only for the task of patch classification. This work will be the first time this dataset is made fully publicly available with its semantic segmentation labels.

## II. RELATED WORK

### A. Weakly Supervised/Sparse Label Semantic Segmentation

Approaches to the problem of semantic segmentation with sparse labels can be mostly divided into two main groups: 1) methods that use the sparse labels without any kind of augmentation [4, 6, 28, 40]; and 2) strategies that try to reconstruct dense annotations from the sparse labels [3, 5, 18, 39].

In the first group, Çiçek et al. [6] and Bokhorst et al. [4] use a weighted loss, Silvestri and Antiga [28] imply the use of padding in the sparse labels, and Zhu et al. [40] use a quality model to ensure a good segmentation based of the sparse annotation.

Lin et al. [18] are one of the first to use sparse labels for semantic segmentation. They use a label propagation scheme in conjunction with a FCN for segmentation. This propagation uses the scribble annotation provided and the prediction of the FCN network. They train their model by alternating which part is trained at each iteration.

Tajbakhsh et al. [31] present a thorough review of deep learning solutions to medical image segmentation problems. They include a section for segmentation with noisy/sparse labels, in which the methods belong to one of the two groups described previously. All the methods reviewed by Tajbakhsh et al. [31] use a selective loss. That is, a type of loss function that has different weights to unlabeled pixels/voxels and thus can ignore such pixels when the total cost is computed.

### B. Few-Shot Semantic Segmentation

As in the few-shot classification problem, information from the support set has an important role in the semantic segmentation case. Multiple works try to insert this information directly into the model's processing flow.

Many works use a two-branch structure, where one branch is responsible for extracting information from the support samples, which is fused into the other branch that processes the query images. Dong and Xing [7] use a two-branch model, where one network produces prototypes of each class, similarly to ProtoNets. The first branch uses the support set and query image to produce prototypes, which are used for image classification in this branch. The encoded query image in the second branch is then fused with the prototypes to produce the probabilities maps. Hu et al. [13] introduce a highly interconnected two-branch attention-based model. The attention modules receive query and support feature maps, and are present in multiple layers of the model. Zhang et al. [38] present another two-branch model. One branch, called *guidance*, is used to extract feature vectors from both query and support images. They compute class prototypes using masked average pooling in the features from support images. A similarity map between the query features and prototypes

Table I: Summary of related work.

| Work | Semantic Segmentation | Few-Shot | Sparse Annotations |
|------|:---:|:---:|:---:|
| Lin et al. [18] | ✓ | | ✓ |
| Vernaza and Chandraker [33] | ✓ | | ✓ |
| Wang et al. [35] | ✓ | | ✓ |
| Zhang et al. [39] | ✓ | | ✓ |
| Bai et al. [3] | ✓ | | ✓ |
| Cai et al. [5] | ✓ | | ✓ |
| Çiçek et al. [6] | ✓ | | ✓ |
| Bokhorst et al. [4] | ✓ | | ✓ |
| Silvestri and Antiga [28] | ✓ | | ✓ |
| Zhu et al. [40] | ✓ | | ✓ |
| Snell et al. [29] | | ✓ | |
| Finn et al. [9] | | ✓ | |
| Dong and Xing [7] | ✓ | ✓ | |
| Hu et al. [13] | ✓ | ✓ | |
| Zhang et al. [38] | ✓ | ✓ | |
| Wang et al. [36] | ✓ | ✓ | |
| Rakelly et al. [24] | ✓ | ✓ | ✓ |
| WeaSeL (Ours) | ✓ | ✓ | ✓ |
| ProtoSeg (Ours) | ✓ | ✓ | ✓ |

is calculated and fused with the query features to compute the final prediction.

Other approaches use a single network to face the few-shot semantic segmentation problem. Wang et al. [36] propose a direct adaptation of the Prototypical Networks. They use a CNN to produce feature vectors of the images in the support set and compute the prototypes for each class using masked average pooling, as [38]. During the training they include an *alignment* loss, where the prototypes are computed from the query image, and the support set is the segmentation target. Rakelly et al. [24] proposed the Guided Networks (Guided Nets), the first algorithm for few-shot sparse segmentation. Although it fuses information from the support set in query features, this model uses a single feature extraction network. This network is a pre-trained CNN backbone that extracts features of the support set and the query image. The support features are averaged through a masked pooling using the sparse annotations provided for the images and further globally averaged across all the support images available. This single averaged support feature multiplies the query features, reweighting them. Then, these features are further processed by a small convolutional segmentation head that gives the final predictions.

In Table I, we present a summary of the related works and how our proposed methods fit in the literature.

## III. METHODOLOGY

### A. Problem Definition

For our problem setup, we employ most of the definitions from Gama et al. [10].

A dataset $\mathcal{D}$ is a set of pairs $(\mathbf{x}, \mathbf{y})$, where $\mathbf{x} \in \mathbb{R}^{H \times W \times B}$ is an image with dimensions $H \times W$ and $B$ bands/channels, and $\mathbf{y} \in \mathbb{R}^{H \times W}$ is the semantic label of the pixels in the image. This dataset is partitioned in two sets: $\mathcal{D}^{sup}$ (support set) and $\mathcal{D}^{qry}$ (query set), such that $\mathcal{D}^{sup} \cap \mathcal{D}^{qry} = \emptyset$. Given a dataset $\mathcal{D}$ and target class $\mathcal{T}$, we define a segmentation task $\mathcal{S}$ as a tuple $\mathcal{S} = \{\mathcal{D}^{sup}, \mathcal{D}^{qry}, T\}$ (or, $\mathcal{S} = \{\mathcal{D}, \mathcal{T}\}$, for simplicity).

A few-shot semantic segmentation task $\mathcal{F}$ is a specific type of segmentation task. It is also a tuple $\mathcal{F} = \{\mathcal{D}^{sup}, \mathcal{D}^{qry}, \mathcal{T}\}$, but the samples of $\mathcal{D}^{sup}$ have their labels sparsely annotated, and the labels in $\mathcal{D}^{qry}$ are absent or unknown. Moreover, the

number of samples $k = |\mathcal{D}^{sup}|$ is a small number (e.g., 20 or less); thus, we also call a few-shot task a $k$-shot task.

Finally, the problem of few-shot semantic segmentation with sparse labels is defined as follows. Given a few-shot task $\mathcal{F}$ and available segmentation tasks $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$, we want to segment the images from the $\mathcal{D}_{\mathcal{F}}^{qry}$ using information from tasks $\mathcal{S}_i$, and information from the $\mathcal{D}_{\mathcal{F}}^{sup}$. Also, there is no information of the tasks $\mathcal{F}$ target objects, other than the sparse annotations of $\mathcal{D}_{\mathcal{F}}^{sup}$ samples. That is, no pair of image/semantic label of $\mathcal{F}$ is present in any task $\mathcal{S}_i$ in either $sup$ or $qry$ partition.

### B. Gradient-based Sparse Segmentation with WeaSeL

We reintroduce our previously proposed method in Gama et al. [10]. The **Wea**kly-supervised **Se**gmentation **L**earning (WeaSeL) is an adaptation of the supervised MAML algorithm [9], as depicted in Figure 1.

Our meta-tasks $\mathcal{T}_i \sim p(\mathcal{T})$ are *segmentation tasks* (i.e the set $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$), as defined in Section III-A. We employ the Cross-Entropy loss ($\mathcal{L}_{CE}$) commonly used in segmentation tasks, defined for a single pixel as:

$$\mathcal{L}_{CE} = -\sum_{i}^{C} y_i \log f_\theta(x)_i, \qquad (1)$$

where C is the number of classes, $y_i$ is the true probability of a class $i$ for the pixel, and $f_\theta(x)_i$ is the predicted probability by the model $f_\theta$ for the class $i$ to the specific pixel. The loss in equation 1 is averaged over all pixels to produce the final loss for an image.

The meta-tasks have dense annotated samples. To train the model in a scenario similar to the target few-shot task, we simulate sparse annotations for the samples in the meta-tasks support set. That is, for all $\mathcal{T}_i \sim p(\mathcal{T})$, the labels of samples in $\mathcal{D}^{sup}$ are randomly converted to a sparse version of themselves (this operation will be further discussed in Section IV-B). With this, we expect that the model learns to predict dense labels from sparse annotations and more easily adapts to few-shot tasks.
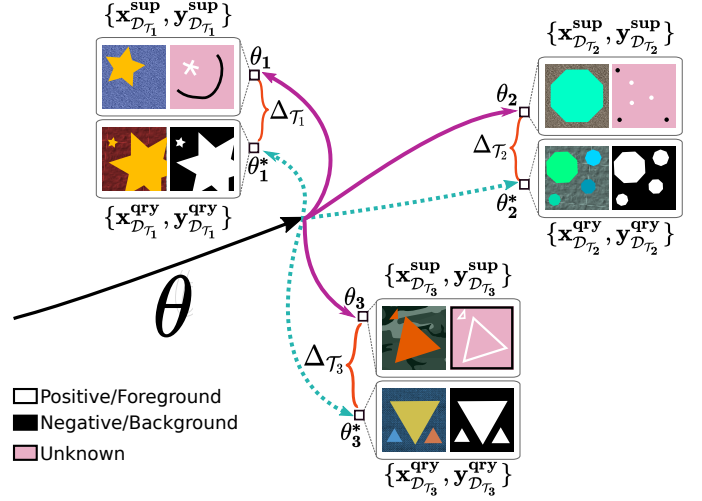
Given that the labels are sparse in the inner loop of meta-training, and during tuning in the few-shot task, we modify the classical Cross-Entropy to a Selective Cross-Entropy (SCE) loss as follows:

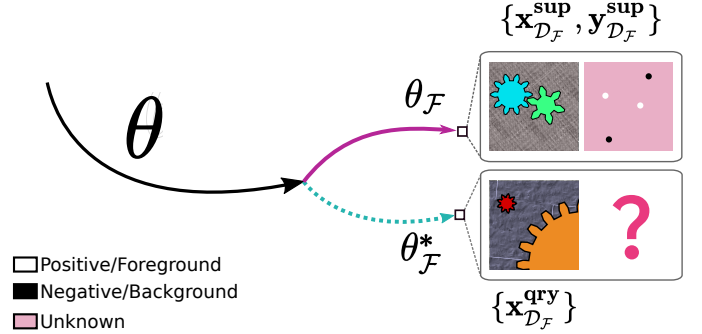$$\mathcal{L}_{SCE} = -\frac{1}{N} \sum_{j} \sum_{i}^{C} w_i y_i^j \log f_\theta(x)_i^j, \qquad (2)$$

where $j$ is a pixel, $N$ is the total number of labeled pixels, and $w_i$ is a indicator, where $w_i = 0$, if $i$ is an unknown label and $w_i = 1$, otherwise. That is, $\mathcal{L}_{SCE}$ ignores pixels with unknown labels via the binary weight parameter, averaging the loss for all pixels with annotations.

Algorithm 1 summarizes the meta-training procedure using the segmentation meta-task distribution $p(\mathcal{T})$. In the inner loop, the loss is computed using the simulated sparse annotations of the support set of a task, and the outer loss using the dense labels of the query set of a task $\mathcal{T}_i$.

After the meta-training phase, we adapt the model to the few-shot task, performing a simple fine-tuning with samples



(a) Visualization of the meta-training process. The global parameter $\theta$ is optimized for different tasks obtaining the parameters $\theta_i$ through optimization using the sparse labels from the tasks support sets. The $\theta_i^*$ is an optimal parameter that could be obtained if the model were trained with the query set samples and dense annotations, which are only used to compute the task outer loss. This hypothetical difference $\Delta$ between parameters is expected to be minimized during the meta-training, leading to a fast/better learner to the few-shot task.



(b) Illustration of the fine-tuning step. The meta-optimized $\theta$ is supervised trained with the sparse annotated samples of the few-shot support set. The labels of the query are unknown, i.e., not seen by the model.

Figure 1: Illustration of the WeaSeL method with toy examples in the meta-training/meta-test phase (a), and in the few-shot tuning phase (b).

from the support set of the few-shot task $\mathcal{F}$. That is, we use pairs $(\mathbf{x}, \mathbf{y}) \in \mathcal{D}_{\mathcal{F}}^{sup}$ to train the model in a supervised manner using a Selective Cross-Entropy loss.

### C. Prototypical Seeds for Sparse Segmentation Via ProtoSeg

The proposed method for semantic segmentation based on the Prototypical Networks [29] is a straightforward adaptation of the original method. It uses the same premise of constructing a prototype vector to each class, with the distinction that prototypes are computed using the labeled pixels instead of whole image instances.

Given a support set $S = \{(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)\}$, where $x_i \in \mathbb{R}^{H \times W \times B}$ is an image with height $H$, width $W$ and $B$ channels, and $y_i \in \mathbb{R}^{H \times W}$ is a label image

**Algorithm 1** Training algorithm for WeaSeL .

---

**Require:** $p(\mathcal{T})$: distributions over tasks
**Require:** $\alpha, \beta$: step size hyperparameters
  Randomly initialize $\theta$
  **while** not done **do**
    Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
    **for all** $\mathcal{T}_i$ **do**
      Sample batch of datapoints $S_i = \{(\mathbf{x}, \mathbf{y})\}$ from $\mathcal{D}_{\mathcal{T}_i}^{sup}$
      Compute $\nabla_\theta \mathcal{L}_{CE}(f_\theta)$ using $S_i$ and $\mathcal{L}_{SCE}$
      Update parameters: $\theta_i = \theta - \alpha \nabla_\theta \mathcal{L}_{SCE}(f_\theta)$
      Sample batch of datapoints $Q_i = \{(\mathbf{x}, \mathbf{y})\}$ from $\mathcal{D}_{\mathcal{T}_i}^{qry}$
    **end for**
    Update $\theta \leftarrow \theta - \beta \nabla_\theta \sum_{\mathcal{T}_i} \mathcal{L}_{CE}(f_{\theta_i})$ using $Q_i$ and $\mathcal{L}_{CE}$
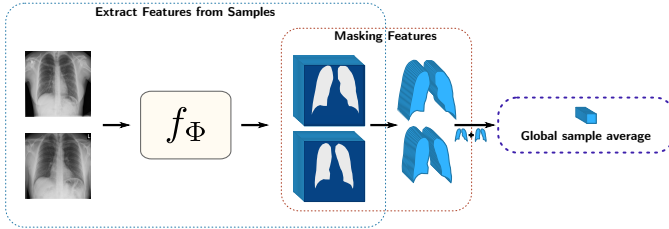  **end while**

---



Figure 2: Illustration of our Average Pooling. After masking the features our process create the *global sample average* by considering all pixels in the set.

with the semantic class of each pixel in $x_i$. Since $y_i$ can be sparse, the possible values of an pixel $j$ in $y_i$ are in the set $\{0, 1, 2, \ldots, K\}$, where $K$ is the total of classes and 0 represents the unknown class.

In our adaptation of ProtoNets, we define the $n$-dimensional prototype vector $\mathbf{c}_k$, of a class $k$ as:

$$\mathbf{c}_k = \frac{1}{N_k} \sum_{(x_i, y_i) \in S} \sum_j [f_\Phi(x_i) \odot \mathbb{1}_k(y_i)]^j, \quad (3)$$

where $f_\Phi : \mathbb{R}^{H \times W \times B} \to \mathbb{R}^{H \times W \times n}$ is our embedding function parametrized with $\Phi$ (a CNN), $\odot$ is point-wise multiplication, and $\mathbb{1}_k(y_i) \in \{0, 1\}^{H \times W}$ is a mask matrix where each value is defined as

$$\mathbb{1}_k^j(y_i) = \begin{cases} 1, & \text{if } y_i^j = k \\ 0, & \text{otherwise} \end{cases}$$

And $N_k = \sum_{y_i} \sum_j \mathbb{1}_k^j(y_i)$ is the total number of pixels of the class $k$, across all the support set $S$. This means that our prototype vector $\mathbf{c}_k$ is the mean vector of all pixels of a class existent in the support set. This is similar to a masked average pooling, but considering all pixels globally, opposed to averaging for each sample and then averaging these pooled vectors. (See Figure 2).

The inference is the same of the original Prototypical Networks, but applied to a pixel in the image. Formally, the probability of a pixel $j$ of a query image $\mathbf{q} \in \mathbb{R}^{H \times W \times B}$ belonging to a class $k$ is computed as follows:

$$p_\Phi(y^j = k | \mathbf{q}) = \frac{\exp(-d(f_\Phi(\mathbf{q})^j, \mathbf{c}_k))}{\sum_i \exp(-d(f_\Phi(\mathbf{q})^j, \mathbf{c}_i))}, \quad (4)$$

where $d$ is the squared euclidean distance: $d(\mathbf{u}, \mathbf{v}) = ||\mathbf{u} - \mathbf{v}||^2$.

Similar to the case of the WeaSeL method, given the presence of unknown labeled pixels in training, we modify our loss function to ignore such pixels. We define our new loss function $J(\Phi)$ as follows:

$$J(\Phi) = -\frac{1}{|Q|} \sum_{(\mathbf{x}, \mathbf{y}) \in Q} \sum_{j \in \mathbf{x}} \sum_{k=1}^{K} \log p_\Phi(\mathbf{y}^j = k | \mathbf{x}^j), \quad (5)$$

where $Q$ is the set of images used to compute the loss, $j$ is a pixel coordinate and $k$ represents a class. Note that $k$ starts from 1, thus not considering the unknown class $k = 0$. We use $p_\Phi$ as defined in equation 4.

Given equations 3 and 4, the model $f_\Phi$ is trained using a episodic training strategy. This strategy resembles the training algorithm of WeaSeL and is presented in algorithm 2. It uses the same distribution over tasks $p(\mathcal{T})$ as the first method, with the automatically generated sparse annotations of the meta-tasks in training. At each iteration, a batch of tasks is sampled, and for each task $\mathcal{T}_i$, a support set $S_i$ is constructed and used for training.

---

**Algorithm 2** Training algorithm for ProtoSeg .

---

**Require:** $p(\mathcal{T})$: distributions over tasks
  randomly initialize $\Phi$
  **while** not done **do**
    Sample batch of tasks $\mathcal{T}_i \sim p(\mathcal{T})$
    **for all** $\mathcal{T}_i$ **do**
      Sample a support set $S_i = \{(x_1, y_1), \ldots, (x_N, y_N)\}$ from $\mathcal{D}_{\mathcal{T}_i}^{sup}$
      Compute $\mathbf{c}_k$ using $S_i$, for all $k$ using equation 3
      Sample query batch $Q_i = \{(\mathbf{x}_1, \mathbf{y}_1), \ldots, (\mathbf{x}_b, \mathbf{y}_b)\}$ from $\mathcal{D}_{\mathcal{T}_i}^{qry}$
      Compute the loss $J(\Phi)$ as defined in equation 5, using $Q_i$.
      Update $\Phi$ using gradient descent and $\nabla J$
    **end for**
  **end while**
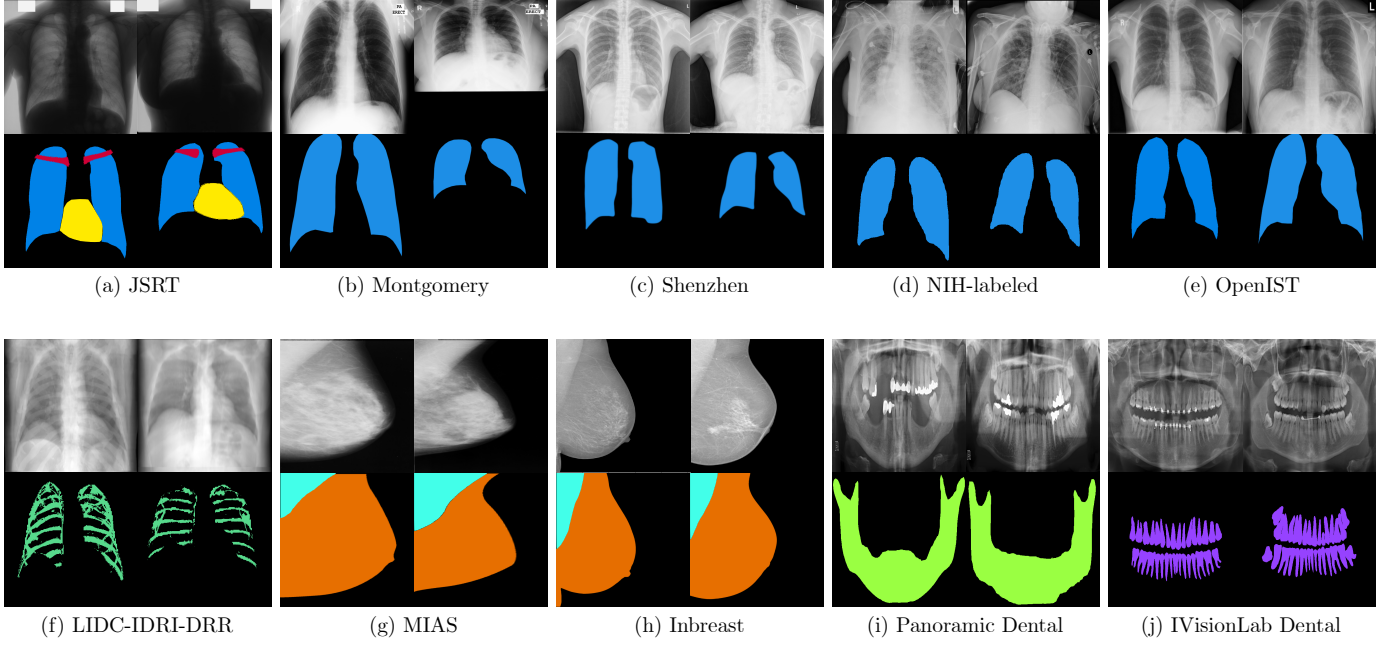
---

## IV. EXPERIMENTAL SETUP

In this section, we present the configurations used for our experiments. In Section IV-A, we briefly present the datasets used. The evaluated sparse labels annotation styles are listed in Section IV-B. Next, in Section IV-C, we introduce the FCN architecture used, and in Section IV-D the baselines, protocol, and metrics are presented.

All the code for the experiments were written in the Python3 language. For the models, we use the Pytorch[1] framework and the Torchmeta[2] module. In relation to the machine, all the experiments were performed on Ubuntu SO, 64-bit Intel i9 7920X machine with 64GB of RAM memory, and a GeForce RTX 2080 TI/Titan XP GPU (only one GPU was used during the experiments).

---

[1] https://pytorch.org
[2] https://github.com/tristandeleu/pytorch-meta

Figure 3: Examples of biomedical imaging datasets included in our medical meta-dataset.

## A. Datasets

We design experiments to evaluate the proposed methods in medical and remote sensing applications for semantic segmentation. These areas have some similarities that contrast them from others. Their images are rather distinct from common RGB images taken with surveillance or cellphone cameras, for instance. This hinders the knowledge transfer from other generic domains or the use of pre-trained models on large datasets such as ImageNet. Another common aspect of these two areas is the limitation of availability of images due to multiple factors. Medical image datasets have to face privacy and ethical concerns, also requiring a highly specialized radiologist to provide precise annotations. In remote sensing the annotation is extremely laborious and sometimes unfeasible since it typically requires that a specialist collect information from large geographical areas, maybe even requiring visits to the site for producing ground truths for these data.

*1) Medical Imaging Datasets:* We use a total of ten medical datasets in our experiments (Figure 3). Of these datasets, six are Chest X-Ray datasets (CRX): JSRT [26] with labels for lungs, heart and clavicles; the Montgomery/Shenzhen sets [14], an annotated subset of Chest X-Ray 8 [37] by Tang et al. [32] referred to as NIH-labeled, OpenIST[3] with labels for lung segmentation, and the LIDC-IDRI-DRR dataset [21], with generated ribs annotations. We include two Mammographic X-Ray (MRX) image sets, namely INbreast [19] and MIAS [30], with labels for breast region and pectoral muscle segmentation. Also, two Dental X-Ray (DRX) datasets are included: Panoramic X-Ray [1] with labels for the inferior mandible and IVisionLab [27] annotated for teeth segmentation.

*2) Remote Sensing Datasets:* The Remote Sensing meta-dataset is composed of rural scenes for crop segmentation (Figure 4). More specifically, we use the Brazilian Coffee dataset, composed of images of 4 municipalities – namely, Arceburgo, Guaranésia, Guaxupé and Montesanto – with pixel-level annotations for coffee crops regions, as well as the Orange Orchards (Ubirajara county, Brazil) dataset, with annotations for orange crop regions. Both datasets will be made public available upon the acceptance of this work.

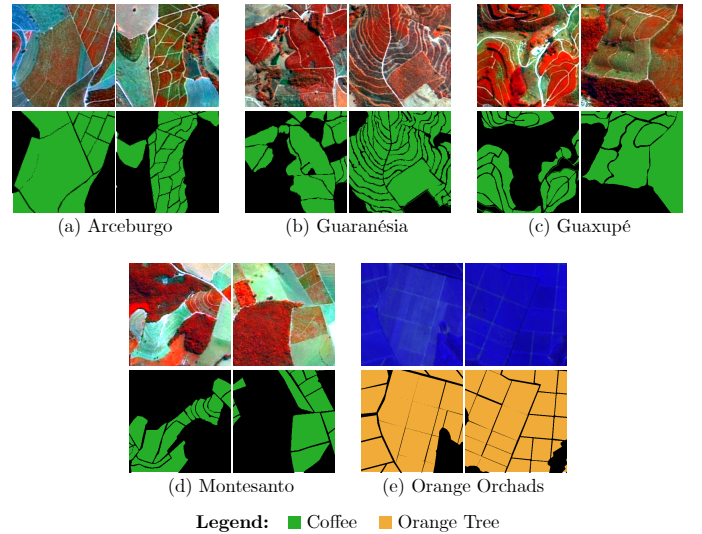Further description of all datasets are presented in the supplementary material.



Figure 4: Examples from the Brazilian Coffee and Orange Orchards Datasets.

[3]https://github.com/pi-null-mezon/OpenIST

### B. Types of sparse annotation

In the experiments we evaluate five types of sparse annotation, namely: *points*, *grid*, *contours*, *skeletons*, and *regions*. As mentioned, we simulate these annotations from the original dense labels of an image. Visual examples of these annotations are show in Figure 5. We can describe each type of annotation and explain how the sparse annotations are generated as follow:

I) **Points:** It simulates an annotator alternately picking pixels from the foreground and background classes. We use a parameter $n$ and randomly choose $n$ pixels from the foreground and $n$ from the background. The remainder pixels are set as unknown.

II) **Grid:** The annotator receives a pre-selected collection of pixels of the image, which are initially assumed to be from the background class. The pixels considered foreground should be annotated. These pre-selected pixels are disposed of in a grid pattern that was generated by using a parameter $s$. First, a random pixel $p_0$ is selected within the following rectangular region: {upper left corner: $(0,0)$ and bottom right corner: $(s,s)$ }. Afterward, a grid is created from this $p_0$ position with $s$ spacing horizontally and vertically. Pixels outside the grid are set as unknown.

III) **Contours:** The annotator denotes the inner and outer boundaries of foreground objects. This style is useful for cases where a single connected foreground object is present. We simulate these annotations by using morphological operations on the original binary dense labels. We used an erosion operation followed by a marching squares algorithm[4] to find the inner contours. To the outer contours, we use a dilation operation on the original label mask and the same marching squares algorithm. Additionally, we use a parameter $d$ that determines the density of the sparse annotation.

IV) **Skeleton-Based Scribble:** It resembles an annotator drawing a scribble roughly at the center of the foreground objects that more or less approximate the object form. The same process is applied to the background. These annotations are generated using the *skeletonize algorithm*[5] in the binary dense label masks, which returns the skeletons of the foreground objects. The same process is applied to the negative dense label masks to obtain the skeleton of the background class. Dilation is applied to add thickness to the skeletons. We use a parameter $d$ to control the density of the annotation. We generate random binary blobs (using this[6] function) that occupy $d$ percentage of the image space and use them to mask the computed skeletons.

V) **Regions:** This type of annotation represents the process of an annotator appointing classes to *pure superpixels*. We define a *pure superpixel* as a usually small connected set of pixels with the same class. The annotator is provided with the superpixels of the image and then appoints the class of a subset of pure foreground and background superpixels. To generate these annotations, first, we compute the superpixels of the images using the SLIC algorithm [2] with empirically chosen parameters for each dataset. Once superpixels were computed, we randomly selected a $d$ percentage of the superpixels for the foreground and a $d$ percentage of superpixels for the background.

### C. miniUNet architecture

The network model used in all experiments – Baselines, WeaSeL, and ProtoSeg– is a simplified version of the UNet architecture [25]. We will call it miniUNet since it is a smaller version of the original network. More information about the miniUNet architecture can be seen in the supplementary material of this manuscript.

In ProtoSeg, since we want to generate $n$-dimensional feature vectors, the last layer of the network is ignored and the output is gathered from the last decoder block. That is, the embedding function $f_\Phi$ is the miniUNet model excluding the last $1 \times 1$ convolutional layer, with this, the prototypes are 32-dimensional.

### D. Evaluation Protocol

*1) Baselines:* We use two baselines for comparison with our approaches: 1) *From Scratch* and 2) *Fine-Tuning*. Given our few-shot semantic segmentation problem parameters in the form of the set of segmentation task $\{\mathcal{S}_1, \mathcal{S}_2, \ldots, \mathcal{S}_n\}$, and a few-shot task $\mathcal{F}$, we define our baselines as follow:

**From Scratch**: Given our miniUNet network, we perform a simple supervised train with the Few-shot task support set ($\mathcal{D}_\mathcal{F}^{sup}$). We use the same Cross-Entropy loss ignoring unlabeled pixels as our cost function (Equation 2). The Adam optimizer [16] was used, with the same parameter used in the training of our methods.

**Fine-tuning**: We use the miniUNet architecture. We choose one task $\mathcal{S}_i$ from our tasks set, and perform a supervised train with the $\mathcal{D}_{\mathcal{S}_i}^{sup}$. Once finished the training on $\mathcal{S}_i$, we perform the fine-tune (a supervised training) using the $\mathcal{D}_\mathcal{F}^{sup}$ set. Again, the same Cross-Entropy loss function (Equation 2) is used with the same parametrized Adam optimizer.
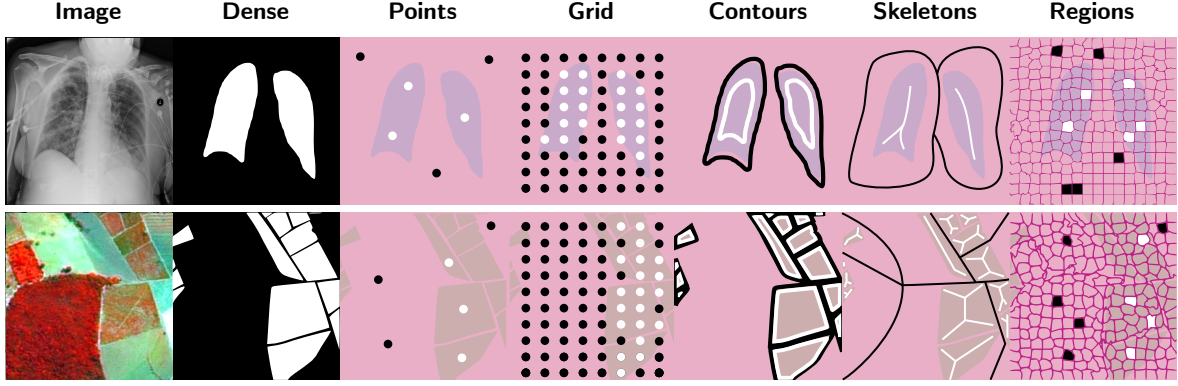
We choose to not present the Guided Nets [24] as a baseline in this work. The use of pre-trained CNN as features extractors seen to be essential to the efficiency of the model, and did not translate well to our evaluate scenarios (medical and remote sensing). To the best of our efforts the model was not able to converge to a usable model with our Meta-Datasets. Thus, it did not seem fair to compare the Guided Nets to our approach.

*2) Protocol and Metrics:* In order to assess the performance of our methods in a certain setting, we employ a Leave-One-Task-Out methodology. That is, all but the pair $(dataset, class)$ chosen as the Few-shot task ($\mathcal{F}$) are used in the Meta-Dataset, reserving $\mathcal{F}$ for the tuning/testing phase. This strategy serves to simultaneously hide the target task from the meta-training while also allowing the experiments

---

[4]https://scikit-image.org/docs/0.8.0/api/skimage.measure.find_contours.html#find-contours

[5]https://scikit-image.org/docs/0.8.0/api/skimage.morphology.html?highlight=skeletonize#skeletonize

[6]https://scikit-image.org/docs/dev/api/skimage.data.html#skimage.data.binary_blobs

| Image | Dense | Points | Grid | Contours | Skeletons | Regions |

**Legend:** ☐ Foreground/Positive Labels ■ Background/Negative Labels ☐ Unknown Labels
■ Superpixel Boundaries

Figure 5: Illustration of the types of sparse annotations used. Annotations are illustrative and upscaled to better visualization.

to evaluate the proposed algorithm and baselines in a myriad of scenarios. Moreover, we divide our tasks into two groups to perform the experiments: (I) **Medical Tasks**: all the medical datasets and their classes are used for these tasks, totaling 13 tasks; (II) **Remote Sensing Tasks**: we used rural datasets for these tasks. There are 5 tasks in total (4 from the Brazilian Coffee and 1 from the Orange Orchards dataset).

For each method, we used a different number of epochs in each of their training phases. In table II, we show these numbers that differ mostly due to training time. The Remote Sensing datasets are, in general, larger than the Medical datasets, and this made the training process (that includes validation) more time-consuming. We use the Adam optimizer [16] with learning rate 0.001, weight decay 0.0005, and momentum 0.9. Our batch size was set to 5. The number of tasks sampled for the inner loop of WeaSeL and ProtoSeg was set to 6 in Medical experiments and 4 in Remote Sensing experiments due to memory constraints, in general, and the total number of tasks in Remote Sensing experiments.

Table II: Number of epochs for training the methods in different experiments.

| Method | Medical Experiments | | Remote Sensing Experiments | |
|---|---|---|---|---|
| *Total Epochs* | **Pre/Meta-Training** | **Tuning** | **Pre/Meta-Training** | **Tuning** |
| **WeaSeL** | 2000 | 80 | 200 | 40 |
| **ProtoSeg** | 2000 | - | 200 | - |
| **Fine-Tuning** | 200 | 80 | 100 | 80 |
| **From Scratch** | - | 80 | - | 100 |

We use a 5-fold cross-validation protocol in the experiments. Each dataset had a training and validation partition for each fold. Once fix the experiment fold, the support sets for the tasks are obtained from the training partition of the dataset, while the query sets are the entire validation partition.

All images and labels are resized to $256 \times 256$ for remote sensing images and $128 \times 128$ for medical images prior to being fed to the models. This was due to our infrastructure limitations and done to standardize the input size and minimize the computational cost of the methods, especially the memory

footprint of WeaSeL method, due to the computation of second derivatives, on high-dimensional outputs.

The metric within a fold is computed for all images in the query set according to the dense labels, and is averaged in relation to the images in that fold. The metric used is the Jaccard score (or Intersection over Union – IoU) of the validation images, a common metric for semantic segmentation.

## V. RESULTS AND DISCUSSION

In this section, we present and discuss the results of our experiments. Section V-A shows a comparison of the results of the proposed methods and baselines using multiple sparse annotations and their densely annotated counterparts. Section V-A1 focuses on the medical imaging datasets, while Section V-A2 describes the results obtained from remote sensing data. At last, in Section V-B we evaluate different sparse annotation styles regarding the number of user inputs and segmentation performance.

### A. Few-shot Semantic Segmentation: Sparse vs Dense labels

In this section, we present the results of our methods in multiple few-shot tasks in the Medical and Remote Sensing scenarios. We evaluated different number of shots and parameters for each type of sparse annotation. In Sections V-A1 and V-A2, we present the results grouped in plots organized by sparse annotation type and number of shots. Dashed lines in the graphs represent the scores of the methods trained with dense annotations.

*1) Medical Tasks:* Analyzing the results in the CRX tasks, two trends can be easily seen. First, an obvious insight that holds for most methods and scenarios is that better scores are obtained with more data. Larger support sets (higher $k$-shots) and more sparsely annotated pixels result in better performance for the algorithms. A second observed result is that WeaSeL surpassed the performances of ProtoSeg and baselines in tasks with a larger domain shift to other tasks in the meta-dataset. This was observed mainly in the *JSRT Lungs* (Figure 6) and the *JSRT Heart* (Figure 7) experiments, as the

JSRT dataset is visually the most distinct of the CRX datasets. Additionally, the *Heart* class is annotated only in this dataset, resulting in a large domain shift in the semantic space for this task in comparison to the other tasks used in the meta-training.

For the remaining tasks, we observe that either the ProtoSeg method or some fine-tuning baseline is the best performer. Since some datasets are visually similar, fine-tuning for a task from a model trained in a similar dataset is a known viable solution that works well in these cases. Fine-tuning from similar tasks (e.g., OpenIST, Montgomery, or Shenzhen for lung segmentation) yields the best Jaccard scores in most cases, as exemplified in Figure 8 for the OpenIST dataset. Also, we observe that the ProtoSeg is consistently comparable to these fine-tuned baselines. The same plots (as in Figure 8) for Montgomery and Shenzhen tasks, omitted in this text due to size constraints, can be found in the supplementary material.

MRX and DXR tasks present similar tendencies as the ones observed in the CRX datasets. Again, fine-tuning from similar tasks appears as a solid solution, with WeaSeL obtaining comparable results to the baselines in most cases. In the *MIAS Breast* task (Figure 9), fine-tuning from the *INbreast Breast* proved to be the best method, mainly due to having the same semantic space on the source and target domains, closely followed by WeaSeL in most scenarios.

Two DXR datasets are included in the meta-dataset in our experiments, assuring that in experiments with one DXR dataset as target, the other one is always used for the pretraining. However, the *Panoramic* dataset is labeled for mandibles while *IVisionLab* data are labeled for teeth, hence never sharing the same label space. In this scenario, without a task with similar semantic space to fine-tune from, the WeaSeL yields the best performance in the segmentation tasks, achieving the highest scores in the majority of experiments with both dense and sparse annotations. This can be observed in Figure 10 for the *Panoramic Mandible* task. Being the most distinct tasks, even the from scratch baseline yields comparable results to the more complex alternatives, in some cases even achieving the best results. ProtoSeg underperforms by a large margin in comparison to the other methods in *Panoramic Mandible*, which can be explained by the low prevalence of DXR data in the meta-dataset used for meta-training and by the large semantic space domain shift even among DXR datasets.

*2) Remote Sensing Tasks:* In general, all remote sensing tasks proved to be considerably harder than the medical ones. Overall, no method achieved a Jaccard score above $0.8$ in any of the evaluated tasks, not even when using dense labels. Figures 11 and 12 depict the results for the *Montesanto Coffee* and *Arceburgo Coffee* tasks, from the Brazilian Coffee dataset, while Figure 13 shows results for the *Orange Orchard* task. One can easily observe that the WeaSeL method consistently outperforms fine-tuning and from scratch baselines, especially in configurations with few data (1-shot tasks). Although having the same label space, the coffee segmentation presents an intrinsic domain shift across the 4 different counties in the dataset. This is due to distinct geographical features, coffee crop cycles and/or plantation methods, explaining why simple fine-tuning is not always the best solution for coffee crop segmentation.

ProtoSeg had consistent results in most agricultural tasks. For the *Coffee* tasks, it generally obtained Jaccard scores around $0.6$, while the performance in the *Orange* task revolved around $0.4$. In a tendency similar to the Medical experiments, ProtoSeg seems to benefit from very related/similar tasks, particularly regarding the semantic space. When choosing *Orange Orchard* as the target task, only the *Coffee* tasks were available to be used in training, explaining its lower performance in comparison to the *Coffee* datasets.
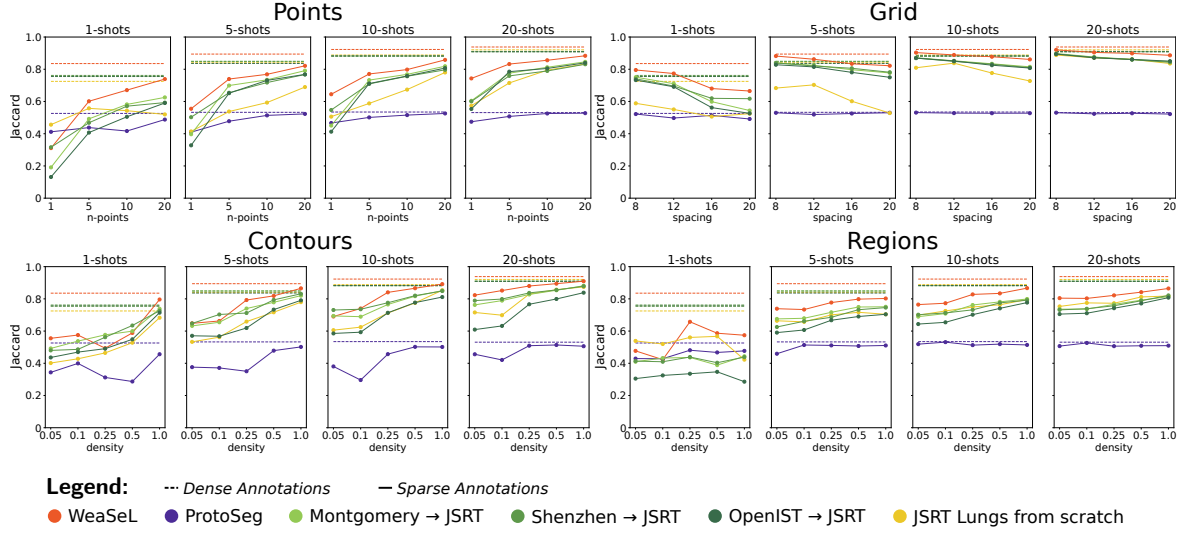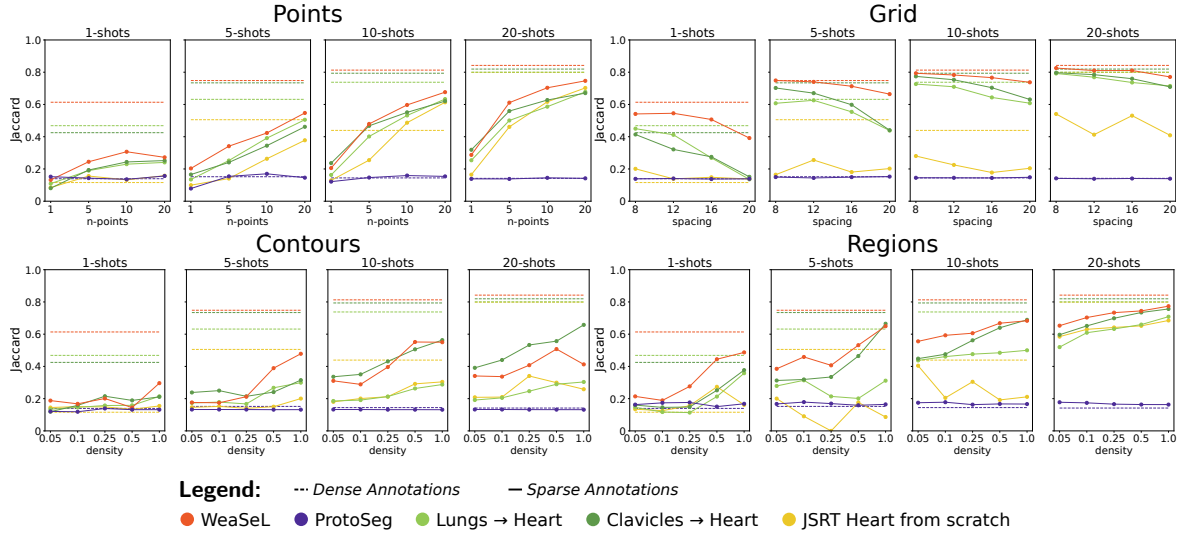
*B. Sparse Label Efficiency Comparison*

In this section, we present results for three types of sparse annotations: *Points*, *Grid*, and *Regions*. *Contour* and *Skeleton* annotations are not evaluated due to our methods to generate them. We define the number of user inputs for a type of annotation as the number of interactions an annotator would have to perform to annotate the image sparsely using said type.

For a single image, the number of inputs for a $n$-point *Points* annotation is $2n$: the $n$ positive and $n$ negative pixels selected. For the *Grid* annotation, the number of inputs is the total positive labeled pixels in the grid since they are initially assumed to be negative, and the user picks the positive ones. As for the *Regions* annotation, the number of inputs is defined as the total regions selected, independent of being positive or negative. After the number of inputs for a single image is computed, the values are summed for all images in the support set of the $k$-shot task. Then, the number of inputs of the $k$-shot are averaged across the five folds.

Figures 14 and 15 present results label efficiency plots for *JSRT Lungs* and *Montesanto Coffee*, respectively. We observe that, as seen in Section V-A, the WeaSeL method overall performs better with more data, that increases with the number of user inputs. We also clearly see how the ProtoSeg method is almost indifferent to the sparsity and quantity of annotations by having a low deviation score in the presented tasks. For the *JSRT Lungs* task (Figure 14), and Medical tasks, in general, we see that the *Grid* annotation usually achieve a higher score for the same number of user inputs as the other types of annotation. On the other hand, the *Region* annotation is commonly the best annotation type for the Remote Sensing tasks, having higher scores with the same number of inputs. The *Points* annotation is, at most times, the worse performer. This was expected since with the same number of inputs, this type of annotation will have fewer labeled pixels in total than the other types.

By comparing these results and the ones of the previous section (Section V-A) we can draw some discussions. The *Grid* annotation is a solid annotation type that can lead to good results and usually is one of the best types for Medical cases. However, this is the most user-consuming type, requiring a larger number of user inputs. The *Regions* annotation is also a solid option for annotations. When the superpixel segmentation produces clean regions easier to be labeled, this type can make annotation simpler and quicker and produce precise models, especially for Remote Sensing tasks. The *Points* annotation requires less from the user. It does not produce the most optimal models but can lead to comparable results

Figure 6: Jaccard score of experiments with *JSRT Lungs* task.



Figure 7: Jaccard score of experiments with *JSRT Heart* task.

requiring far fewer inputs. Also, this annotation guarantees a balanced number of pixels samples for each class in training, making optimization of the models easier. The other two types of annotations, *Contours* and *Skeletons*, appear as valid options as well. The way we designed the process of generating these annotations made it difficult to translate to a countable user input, which is why these types are not compared in this section. However, the results presented in Section V-A show that *Contours* and *Skeletons* annotations are suitable styles, specially *Contours* in the Medical tasks and *Skeletons* in Remote Sensing tasks.

## VI. CONCLUSION AND FUTURE WORKS

In this work, we proposed one method (ProtoSeg) to the problem of weakly supervised few-shot semantic segmentation, also conducting extensive experiments on similar previously proposed method (WeaSeL [10]). Despite being common in shallow interactive segmentation methods, few-shot segmentation from sparse labels is still not fully integrated with

the advances in computer vision brought by Deep Learning. We evaluated our two meta-learning methods in a large number of experiments to verify their generalization capabilities in multiple image modalities, number of shots, annotation types and label densities. We chose to focus the experiments in two areas that can benefit from few-shot sparse labeled semantic segmentation: medical imaging and remote sensing.

WeaSeL [10], obtained promising results, mainly in scenarios with a large domain shift between the target and source tasks. The proposed ProtoSeg method yielded reliable segmentation predictions in the cases wherein there are multiple closely related source datasets, as good results from ProtoSeg appear to be correlated to the availability of similar tasks during training. The five annotation types evaluated in our experiments — *Points*, *Grid*, *Contours*, *Skeletons*, and *Regions* — have their own pros and cons. The *Grid* annotation proved to be highly reliable and produce some of the best results, even though it often requires more user intervention. *Region* annotations can be a more efficient option, but its usefulness is
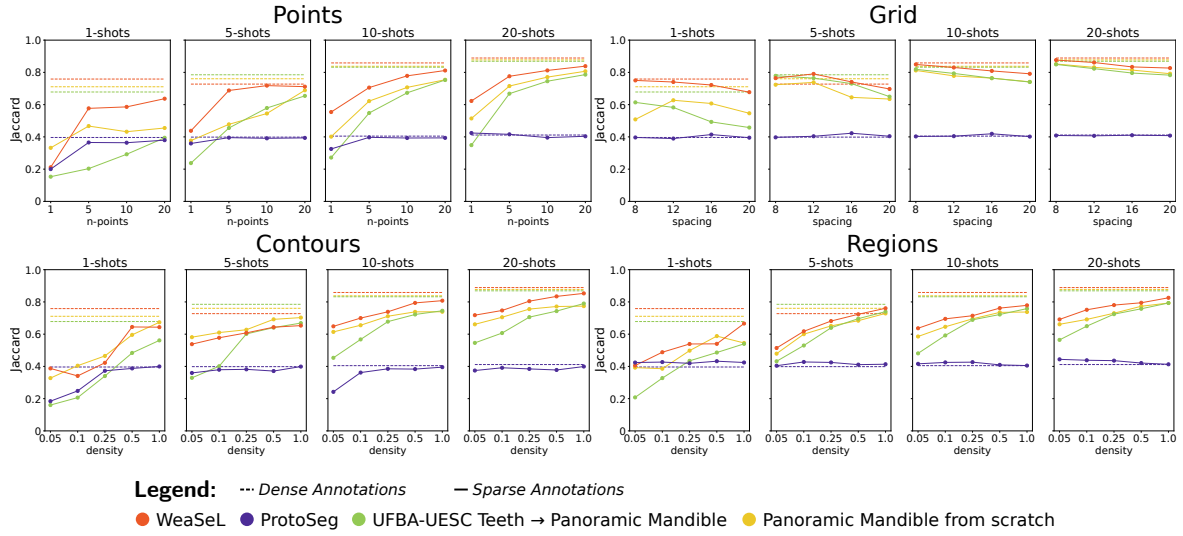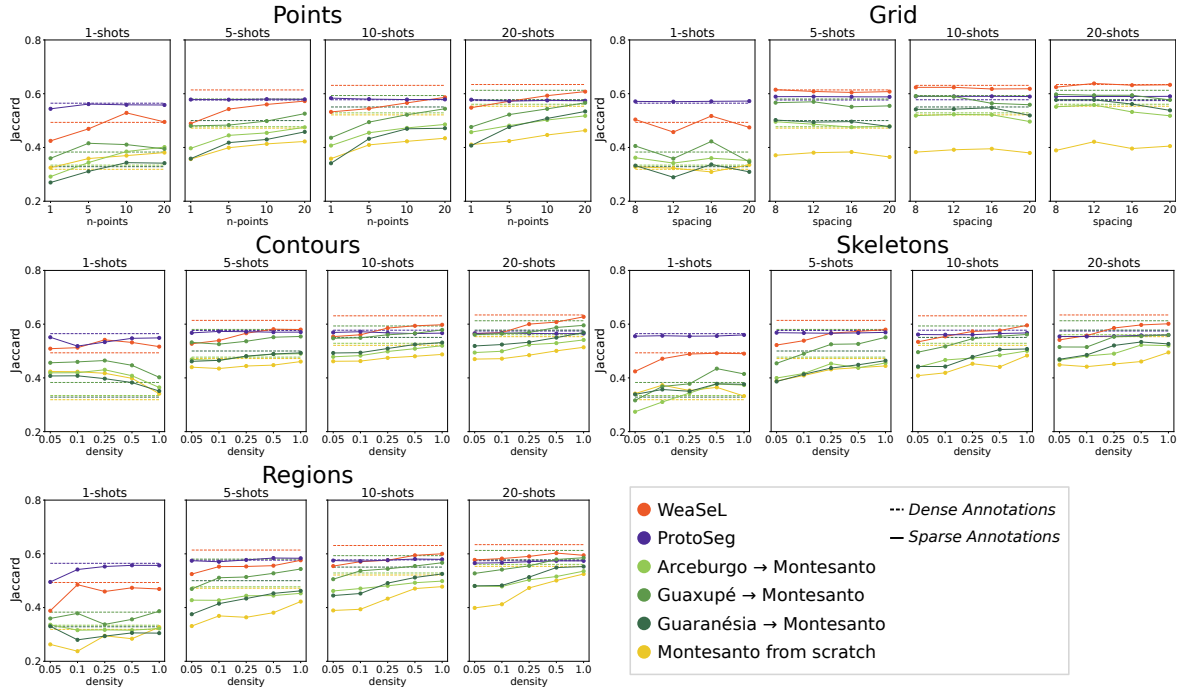
Figure 8: Jaccard score of experiments with *OpenIST Lungs* task.



Figure 9: Jaccard score of experiments with *MIAS Breast* task.

highly affected by the performance of the superpixel segmentation algorithm. *Points* annotations are the less user demanding, but it also yields the greater gaps for dense annotation scores. *Contours* and *Skeletons* appear as valid options for medical imaging and remote sensing tasks, respectively. However more experiments must be conducted to confirm their efficiency in comparison to the other label modalities.

For future works, we intend to investigate adding spacial reasoning into the segmentation predictions in order to account for the location and feature representations of a given pixel in comparison to annotated pixels. Additionally, further experiments in other medical imaging (e.g. other 2D x-ray exams, volumetric images, etc) and remote sensing (e.g. urban segmentation tasks) will be conducted using both ProtoSeg and WeaSeL. At last, our team shall investigate the need for real annotations at all during the meta-training phase. Instead, we plan to replace the sparse masks for organs and crops by automatically generated weakly supervised masks of regions obtained by shallow unsupervised segmentation algorithms.

## REFERENCES

[1] A. H. Abdi, S. Kasaei, and M. Mehdizadeh, "Automatic Segmentation of Mandible in Panoramic X-Ray," *Journal of Medical Imaging*, vol. 2, no. 4, p. 044003, 2015.

[2] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Süsstrunk, "SLIC Superpixels Compared to State-of-the-Art Superpixel Methods," *IEEE TPAMI*, vol. 34, no. 11, pp. 2274–2282, 2012.

[3] W. Bai, H. Suzuki, C. Qin, G. Tarroni, O. Oktay, P. M. Matthews, and D. Rueckert, "Recurrent Neural Networks for Aortic Image Sequence Segmentation with Sparse Annotations," in *MICCAI*. Springer, 2018, pp. 586–594.

[4] J.-M. Bokhorst, H. Pinckaers, P. van Zwam, I. Nagtegaal, J. van der Laak, and F. Ciompi, "Learning from Sparsely Annotated Data for Semantic Segmentation in Histopathology Images," in *International Conference on Medical Imaging with Deep Learning*, 2018.

[5] J. Cai, Y. Tang, L. Lu, A. P. Harrison, K. Yan,

Figure 10: Jaccard score of experiments with *Panoramic Mandible* task.



Figure 11: Jaccard score of experiments with *Montesanto Coffee* task.

J. Xiao, L. Yang, and R. M. Summers, "Accurate Weakly Supervised Deep Lesion Segmentation on CT Scans: Self-Paced 3D Mask Generation from RECIST," *arXiv preprint arXiv:1801.08614*, 2018.

[6] Ö. Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation," in *MICCAI*. Springer, 2016, pp. 424–432.

[7] N. Dong and E. Xing, "Few-Shot Semantic Segmentation with Prototype Learning," in *BMVC*, vol. 3, no. 4, 2018.

[8] E. Ferreira, H. Oliveira, M. S. Alvim, and J. A. dos Santos, "A Comparative Study on Unsupervised Domain Adaptation for Coffee Crop Mapping," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2018, pp.

72–80.

[9] C. Finn, P. Abbeel, and S. Levine, "Model-Agnostic Meta-Learning for Fast Adaptation of Deep Networks," in *ICML*. PMLR, 2017, pp. 1126–1135.

[10] P. H. T. Gama, H. Oliveira, and J. A. dos Santos, "Weakly supervised medical image segmentation," *arXiv preprint arXiv:2108.05476*, 2021.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[12] T. Hospedales, A. Antoniou, P. Micaelli, and A. Storkey, "Meta-Learning in Neural Networks: A Survey," *arXiv preprint arXiv:2004.05439*, 2020.

[13] T. Hu, P. Yang, C. Zhang, G. Yu, Y. Mu, and C. G. Snoek, "Attention-based Multi-Context Guiding for Few-Shot
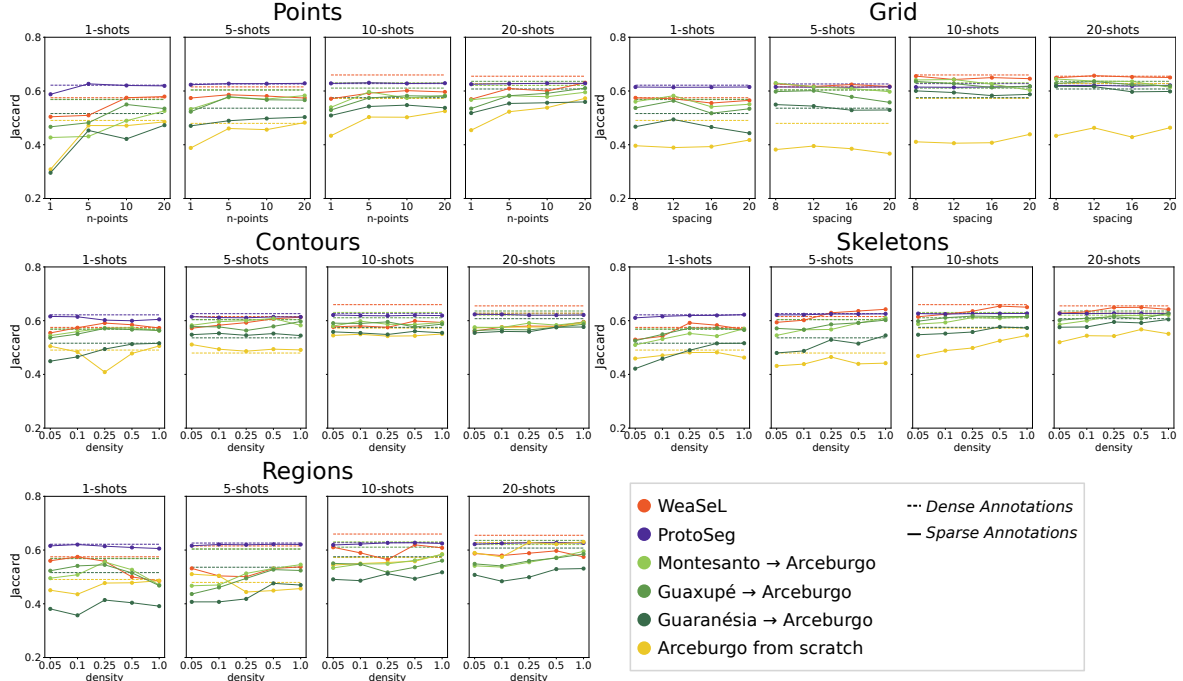
Figure 12: Jaccard score of experiments with *Arceburgo Coffee* task.
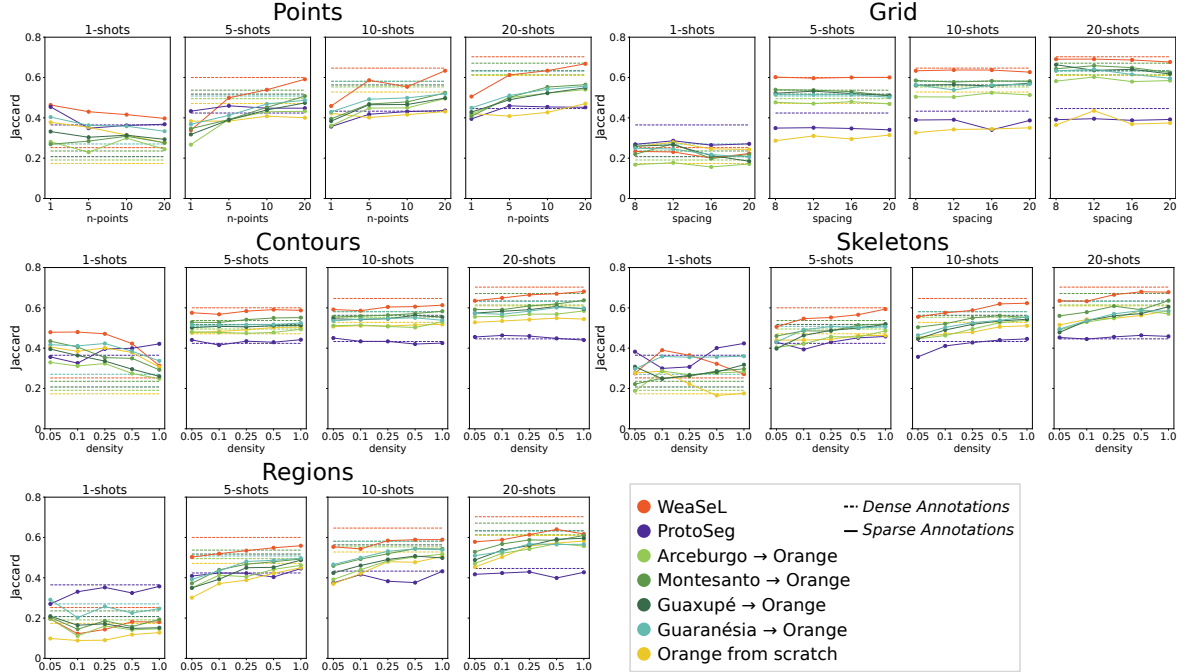


Figure 13: Jaccard score of experiments with *Orange Orchard* task.

Semantic Segmentation," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 8441–8448.

[14] S. Jaeger, S. Candemir, S. Antani, Y.-X. J. Wáng, P.-X. Lu, and G. Thoma, "Two Public Chest X-Ray Datasets for Computer-Aided Screening of Pulmonary Diseases," *Quantitative Imaging in Medicine and Surgery*, vol. 4, no. 6, p. 475, 2014.

[15] H. Kataoka, S. Shirakabe, Y. Miyashita, A. Nakamura, K. Iwata, and Y. Satoh, "Semantic Change Detection with Hypermaps," *arXiv preprint arXiv:1604.07513*, vol. 2, no. 4, 2016.

[16] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," *arXiv preprint arXiv:1412.6980*, 2017.

[17] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet Classification with Deep Convolutional Neural Networks," in *NIPS*, 2012, pp. 1097–1105.

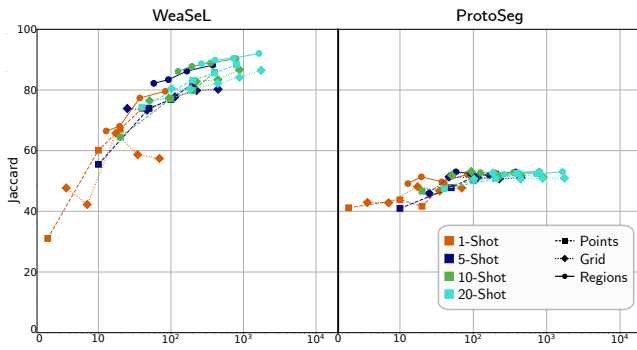[18] D. Lin, J. Dai, J. Jia, K. He, and J. Sun, "ScribbleSup:

Figure 14: Number of user inputs versus Jaccard score in the *JSRT Lungs* task.
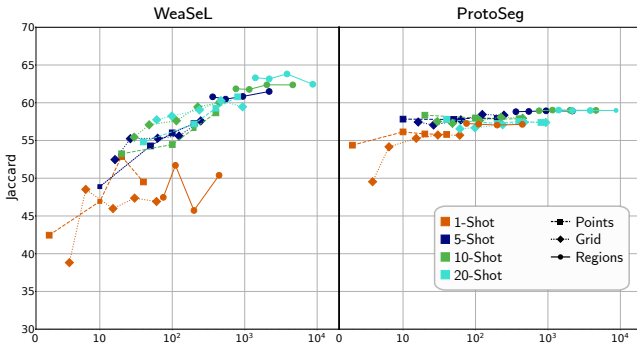


Figure 15: Number of user inputs versus Jaccard score in the *Montesanto Coffee* task.

Scribble-Supervised Convolutional Networks for Semantic Segmentation," in *CVPR*, 2016, pp. 3159–3167.

[19] I. C. Moreira, I. Amaral, I. Domingues, A. Cardoso, M. J. Cardoso, and J. S. Cardoso, "INbreast: Toward a Full-Field Digital Mammographic Database," *Academic Radiology*, vol. 19, no. 2, pp. 236–248, 2012.

[20] K. Nogueira, W. R. Schwartz, and J. A. dos Santos, "Coffee Crop Recognition Using Multi-Scale Convolutional Neural Networks," in *Iberoamerican Congress on Pattern Recognition*. Springer, 2015, pp. 67–74.

[21] H. Oliveira, V. Mota, A. M. Machado, and J. A. dos Santos, "From 3D to 2D: Transferring Knowledge for Rib Segmentation in Chest X-Rays," *Pattern Recognition Letters*, vol. 140, pp. 10–17, 2020.

[22] O. A. Penatti, K. Nogueira, and J. A. Dos Santos, "Do Deep Features Generalize from Everyday Objects to Remote Sensing and Aerial Scenes Domains?" in *CVPR Workshop*, 2015, pp. 44–51.

[23] A. Raghu, M. Raghu, S. Bengio, and O. Vinyals, "Rapid Learning or Feature Reuse? Towards Understanding the Effectiveness of MAML," *arXiv preprint arXiv:1909.09157*, 2019.

[24] K. Rakelly, E. Shelhamer, T. Darrell, A. A. Efros, and S. Levine, "Few-Shot Segmentation Propagation with Guided Networks," *CoRR*, vol. abs/1806.07373, 2018.

[25] O. Ronneberger, P. Fischer, and T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation," in *MICCAI*. Springer, 2015, pp. 234–241.

[26] J. Shiraishi, S. Katsuragawa, J. Ikezoe, T. Matsumoto,

T. Kobayashi, K.-i. Komatsu, M. Matsui, H. Fujita, Y. Kodera, and K. Doi, "Development of a Digital Image Database for Chest Radiographs with and without a Lung Nodule: Receiver Operating Characteristic Analysis of Radiologists' Detection of Pulmonary Nodules," *American Journal of Roentgenology*, vol. 174, no. 1, pp. 71–74, 2000.

[27] G. Silva, L. Oliveira, and M. Pithon, "Automatic Segmenting Teeth in X-Ray Images: Trends, a Novel Data Set, Benchmarking and Future Perspectives," *Expert Systems with Applications*, vol. 107, pp. 15–31, 2018.

[28] G. Silvestri and L. Antiga, "Stereology as Weak Supervision for Medical Image Segmentation," 2018.

[29] J. Snell, K. Swersky, and R. Zemel, "Prototypical Networks for Few-Shot Learning," vol. 30, 2017, pp. 4077–4087.

[30] J. Suckling *et al.*, "The Mammographic Image Analysis Society Digital Mammogram Database," 1994.

[31] N. Tajbakhsh, L. Jeyaseelan, Q. Li, J. N. Chiang, Z. Wu, and X. Ding, "Embracing Imperfect Datasets: A Review of Deep Learning Solutions for Medical Image Segmentation," *Medical Image Analysis*, p. 101693, 2020.

[32] Y. Tang, Y. Tang, J. Xiao, and R. M. Summers, "XLSor: A Robust and Accurate Lung Segmentor on Chest X-Rays Using Criss-Cross Attention and Customized Radiorealistic Abnormalities Generation," 2019.

[33] P. Vernaza and M. Chandraker, "Learning Random-Walk Label Propagation for Weakly-Supervised Semantic Segmentation," in *CVPR*, 2017, pp. 7158–7166.

[34] O. Vinyals, C. Blundell, T. Lillicrap, D. Wierstra *et al.*, "Matching Networks for One Shot Learning," in *NIPS*, 2016, pp. 3630–3638.

[35] G. Wang, W. Li, M. A. Zuluaga, R. Pratt, P. A. Patel, M. Aertsen, T. Doel, A. L. David, J. Deprest, S. Ourselin *et al.*, "Interactive Medical Image Segmentation Using Deep Learning with Image-Specific Fine Tuning," *TMI*, vol. 37, no. 7, pp. 1562–1573, 2018.

[36] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng, "PANet: Few-Shot Image Semantic Segmentation with Prototype Alignment," in *ICCV*, 2019, pp. 9197–9206.

[37] X. Wang, Y. Peng, L. Lu, Z. Lu, M. Bagheri, and R. M. Summers, "ChestX-Ray8: Hospital-Scale Chest X-Ray Database and Benchmarks on Weakly-Supervised Classification and Localization of Common Thorax Diseases," in *CVPR*, 2017, pp. 2097–2106.

[38] X. Zhang, Y. Wei, Y. Yang, and T. S. Huang, "SG-One: Similarity Guidance Network for One-Shot Semantic Segmentation," *IEEE Transactions on Cybernetics*, vol. 50, no. 9, pp. 3855–3865, 2020.

[39] Z. Zhang, J. Li, Z. Zhong, Z. Jiao, and X. Gao, "A Sparse Annotation Strategy Based on Attention-Guided Active Learning for 3D Medical Image Segmentation," *arXiv preprint arXiv:1906.07367*, 2019.

[40] H. Zhu, J. Shi, and J. Wu, "Pick-and-Learn: Automatic Quality Evaluation for Noisy-Labeled Image Segmentation," in *MICCAI*. Springer, 2019, pp. 576–584.

# Weakly Supervised Few-Shot Segmentation Via Meta-Learning: Supplementary Material

Pedro H. T. Gama        Hugo Oliveira
José Marcato Junior        Jefersson A. dos Santos

2021

## 1   miniUnet Architecture

The network is comprised of three encoder blocks, a center block, three decoder blocks and a $1 \times 1$ convolution layer that works as a pixel-classification layer. The network's blocks configuration can be seen in Table 4, where $C$ is the number of input channels obtained from the image domain (e.g. $C = 3$ for RGB images or $C = 1$ for radiology images) and the input/output features represent the number of feature dimensions that the input/output volume has (e.g. *Encoder Block 1* receives an image of size $h \times w \times C$ and outputs a volume of size $\frac{h}{2} \times \frac{w}{2} \times 32$). Similar to the classical UNet architecture [**?** ], skip connections are present in this model. This means that each decoder block receives as input the concatenation of the last block output and the corresponding encoder output. For instance, *Decoder Block 1* receives as input the concatenation of the output volume of *Decoder Block 2* and the output volume of *Encoder Block 1*.

Unlike the original architecture, we pad the images with zeros prior to the convolutions in order to preserve the spacial dimensions if the input. Hence, in the miniUnet architecture only pooling and transposed convolution operations affect the spatial dimensions of the volume during a forward pass in the network. A visualization of the miniUnet architecture can be seen in Figure 17.

Table 4: Descriptions of the miniUNet blocks.

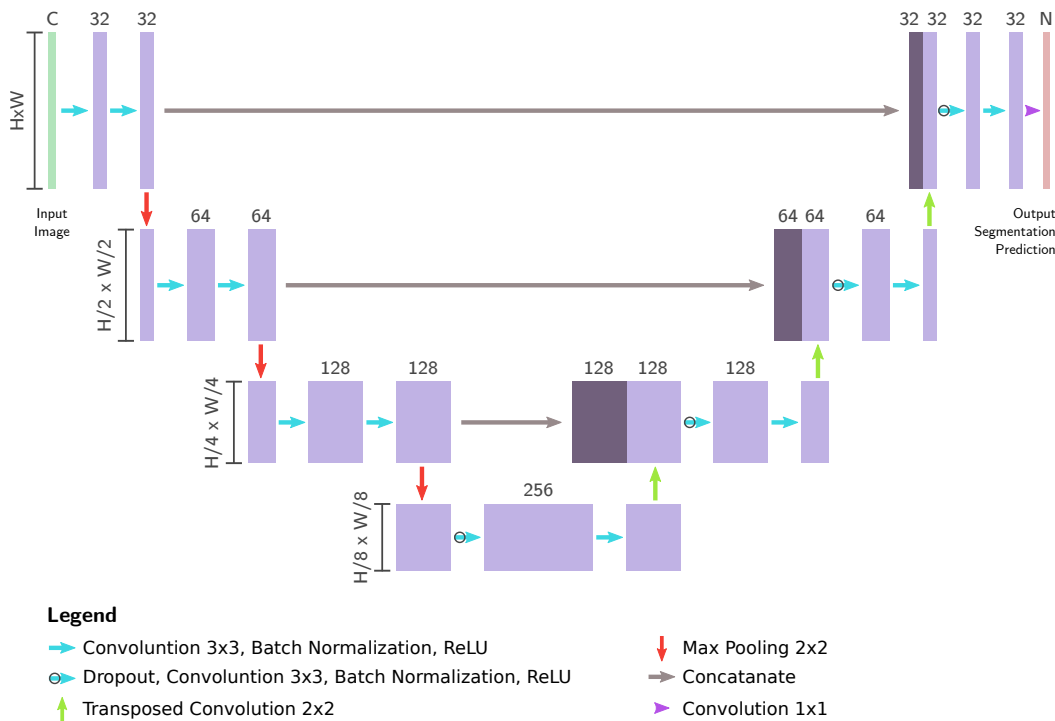| Block Name | Encoder Blocks (1, 2, 3) | Center Block |
|---|---|---|
| Layers | Conv $3 \times 3$<br>Bath Norm.<br>ReLU<br>Conv $3 \times 3$<br>Bath Norm.<br>ReLU<br>MaxPool $2 \times 2$ | Dropout<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU<br>Transposed Conv $2 \times 2$ |
| Input Feat./ Output Feat. | $C$/32, 32/64, 64/128 | 128/128 |
| **Block Name** | *Decoder Blocks (3, 2)* | *Decoder Block (1)* |
| Layers | Dropout<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU<br>Transposed Conv $2 \times 2$ | Dropout<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU<br>Conv $3 \times 3$<br>Bath Normalization<br>ReLU |
| Input Feat./ Output Feat. | 256/64, 128/32 | 32/32 |

Figure 17: Illustration of the miniUNet architecture. The upper numbers represent the feature dimension of the volumes, while on the side is the spatial dimensions.

## 2  Dataset Details

### 2.1  Medical Datasets

**1.A) JSRT Database** [? ] is a collection 247 of chest radiographs initially proposed for lung nodules identification. Masks for lungs, clavicles and hearts structures were obtained from [? ]. From each anatomical structure, a task is derived. All the images have a resolution of $2048 \times 2048$ and 12 bit pixel resolution, and are gray scale. Examples are show in Figure 3(a).

    **1.B) Montgomery Dataset** [? ] is a set of chests X-rays collected from patients in Montgomery County, Maryland, USA. There are 138 frontal X-rays, from which 58 are from cases of Tuberculosis - the initial use case of the dataset. Alongside with information of the patient, for each X-ray, there are binary masks for segmentation of each lung. All the X-rays are 12 bit gray scale images, and either have size $4020 \times 4892$ or $4892 \times 4020$. Examples are shown in Figure 3(b).

    **1.C) Shenzhen Dataset** [? ] were publicized along with the Montogomery dataset. This set is comprised of chests X-rays collected from patients in Shenzhen, China. There are a total of 662 frontal X-rays, from which 336 are from cases of Tuberculosis. There is also binary masks for segmentation of each lung. The size of the X-rays vary, but average a $3000 \times 3000$ resolution, and are gray scale. Examples are shown in Figure 3(c).

    **1.D) NIH-labeled dataset** [? ] is a subset of the original NIH-labeled dataset [? ]. The

original dataset is comprised of 108, 948 frontalview X-ray images of 32, 717 unique patients, and labeled with NLP for 14 different diseases. The subset used in this experiments will be named simply NIH-labeled or XLSor-NIH. This dataset proposed in [**?** ] is comprised of 100 chest X-rays from the original NIH with manually annotated lung masks for these X-rays. All images have a spatial resolution of $512 \times 512$ and are gray scale. Examples can be seen in Figure 3(d).

**1.E) OpenIST Chest X-Rays dataset**[1] is a set of X-rays collected from the following original domain: `http://www.chestx-ray.com/index.php/education/normal-cxr-module-train-your-eye#` `!1`. These images were used to train medical students in recognizing a normal X-ray. There are in total 225 chest X-ray images, with binary masks for the lungs. The images are gray scale with 8-bit resolution and their sizes are not fixed. Examples in Figure 3(e).

**1.F) LIDC-IDRI-DRR dataset** [**?** ] is a dataset derived from LIDC [**?** ]. This dataset is composed of flattened 2D Digitaly Reconstructed Radiographs (DRR) computed from chest CT-scans, as well as generated labels for the ribs. All the 835 images in the dataset are gray scale, and have size $512 \times 512$. Examples of the scans and labels are show in Figure 3(f).

**1.G) MIAS database** [**?** ] is a collection of data from the Mammographic Image Analysis Society (MIAS), a research group in the UK with interest in mammograms. This dataset is composed of 322 digitized mammograms, grayscaled and with a resolution of $1024 \times 1024$. The original dataset only provides labels of location and size of nodules in the images, but we had access to label masks that segment the pectoral muscles and the breast in each image. Example of samples are shown in Figure 3(g).

**1.H) INbreast database** [**?** ] is a collection of 410 images collected from womans in the Breast Center located in the Centro Hospitalar de S. Joao [CHSJ], Porto, Portugal. From these 410 images, only 200 are from Mediolateral Oblique (MLO) view (a side view of the breasts), and have labeled pectoral muscles for them. We also obtained labels for the breasts. All images are gray level with 14 bit resolution and their sizes are either $3328 \times 4084$ or $2560 \times 3328$, depending on the patient. Examples can be seen in Figure 3(h).

**1.I) Panoramic Dental X-rays** [**?** ] dataset is a set of panoramic dental X-rays of 116 patients, taken at Noor Medical Imaging Center, Qom, Iran. The images were mannually segment by three specialists from which label masks for the mandibles were generated. All the images are gray scale and have a size of approximately $2900 \times 1250$ pixels. Examples of sample images and labels are presented in Figure 3(i).

**1.J) IVisionLab Dental Images Dataset** [**?** ] (or, simply, IVisionLab Dataset) is composed of a series of panoramic X-ray dental images. There is a total of 1500 images with annotated teeth labels, with a variety of cases of dental problems and/or formations defects. All the images are gray scale with dimensions of $2440 \times 1292$ pixels. Examples are shown in Figure 3(j).

## 2.2 Remote Sensing Datasets

**2.A) Brazilian Coffee** [**?** **?** ] is a dataset comprised of 4 large satellite images from 4 municipalities of the state of Minas Gerais, Brazil - one satellite image for each county. These Counties being: Arceburgo, Guaranésia, Guaxupé and Montesanto. In the satellite images only three bands were considered, namely the Red, Green, and Near Infrared bands. Along these images, a binary ground truth label image is provided, with positive value representing a coffee crop and negative value representing the background. For evaluation purpose, each of the images where cropped in non-overlapping patches of size $256 \times 256$, and only crops with a percentage of 25% or more of pixels of coffee where maintained for training and validation of the models. Given that each county has distinct geographical features, which lead to coffee plantation distinctions,

---

[1]`https://github.com/pi-null-mezon/OpenIST`

we consider each municipality a different task. Examples of patches of this dataset and their respective ground truths can be seen in Figure 4(a)-(d).

**2.B) Orange Orchards** dataset is comprised of satellite images from an Orange Orchard located at the municipality of Ubirajara, São Paulo, Brazil. In the satellite images four bands are presented, Red, Green, Blue, and Near InfraRed, but only three of the bands were used: all but the Blue band. In addition to the images, annotation masks of the orange plantations were provided, making two classes, namely, Oranges and Background. Each of the satellite images where cropped in non-overlapping patches of size $384 \times 384$, and only crops with a percentage of 10% or more of pixels of plantation where maintained for training and validation of the models. Since this dataset is focused in a single region and have only one interest class, we use it as a single task in our experiments. Examples of patches of this dataset and their respective ground truths can be seen in Figure 4(e).

# 3 Additional Results

In this section, we present extra results for the Section V-A, that were omitted for brevity.

## 3.1 Extra Medical Tasks

This section include the results of four omitted tasks of the Medical Experiments. These tasks are: *JSRT Clavicles* (Figure 18), *Montgomery Lungs* (Figure 19),*Shenzhen Lungs* (Figure 20), *NIH-labeled Lungs* (Figure 21)*MIAS Pectoral Muscle* (Figure 23), *INbreast Breast* (Figure 22), and *IVisionLab Teeth* (Figure 24).
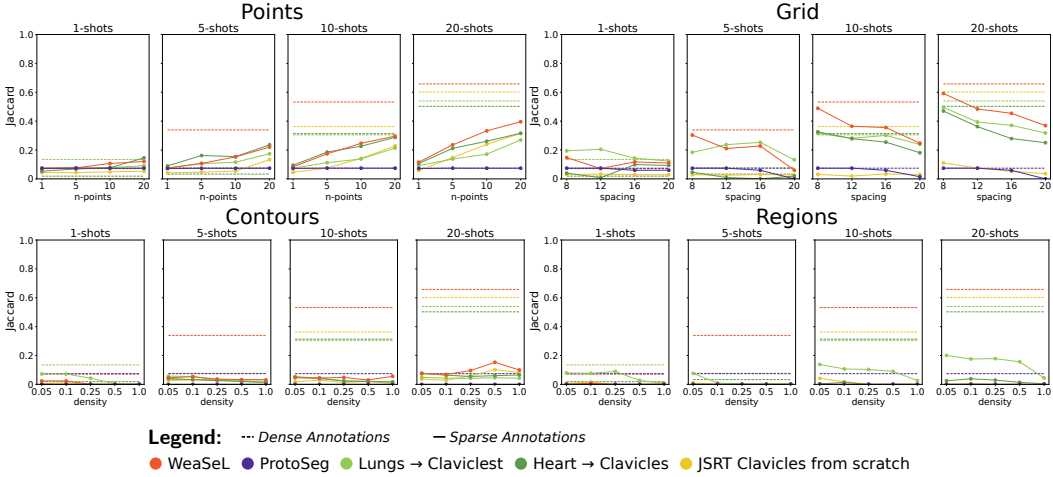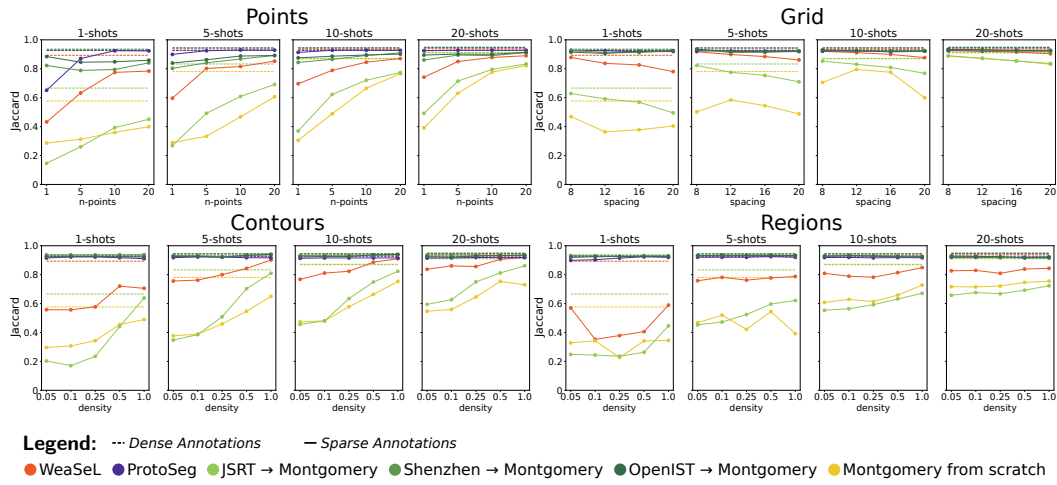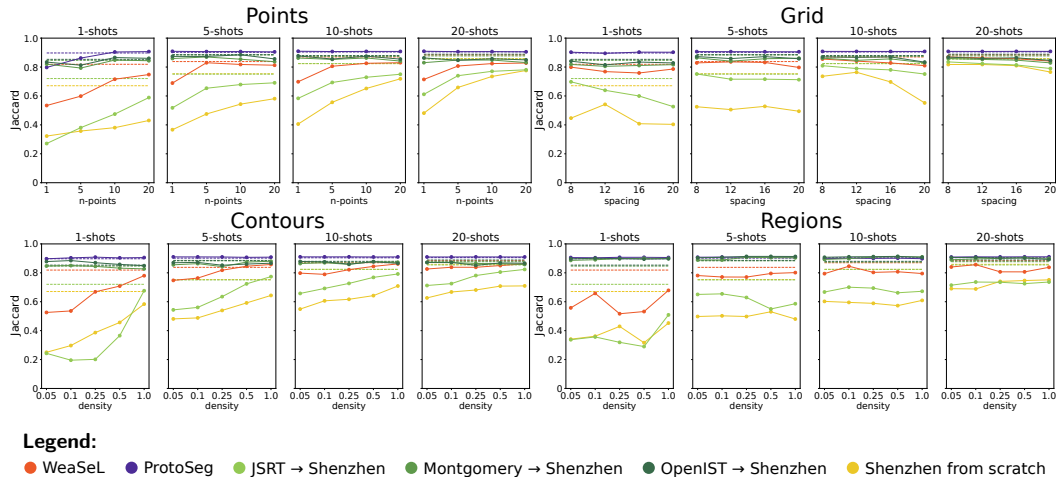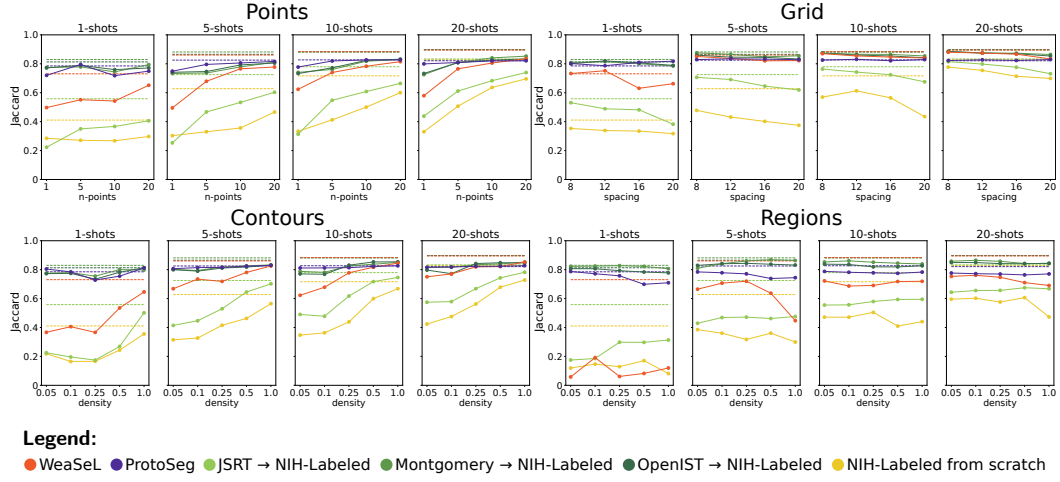


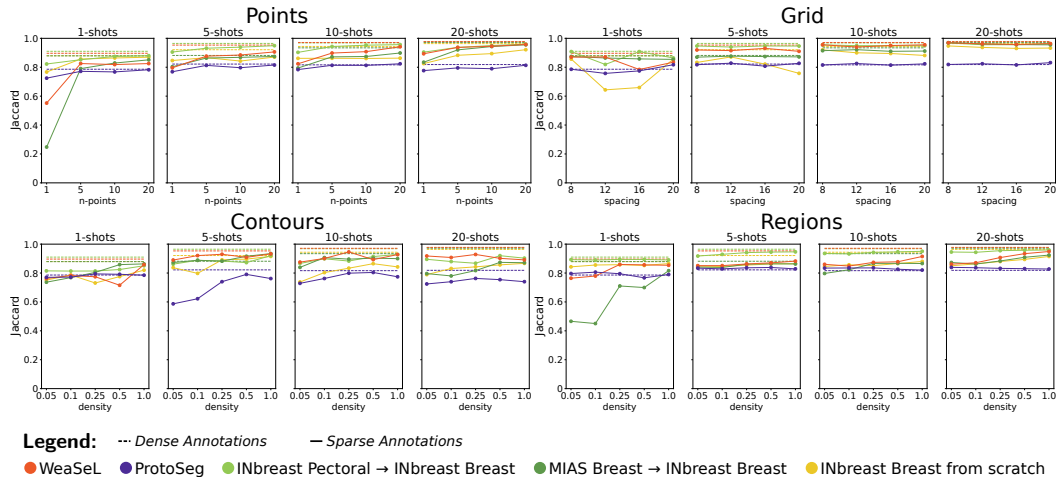Figure 18: Jaccard score of experiments with *JSRT Clavicles* task.

## 3.2 Extra Remote Sensing Tasks

This section include the results of two omitted tasks of the Remote Sensing Experiments. The two tasks from the Brazilian Coffee dataset: *Guaxupé Coffee* (Figure 25), and *Guaranésia Coffee* (Figure 26).

Figure 19: Jaccard score of experiments with *Montgomery Lungs* task.



Figure 20: Jaccard score of experiments with *Shenzhen Lungs* task.

Figure 21: Jaccard score of experiments with *NIH-labeled Lungs* task.



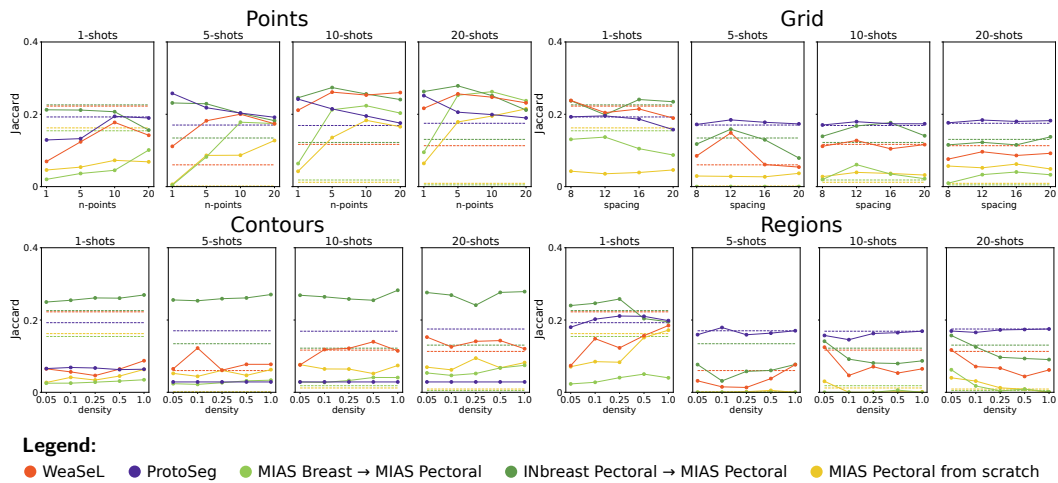Figure 22: Jaccard score of experiments with *INbreast Breast* task.

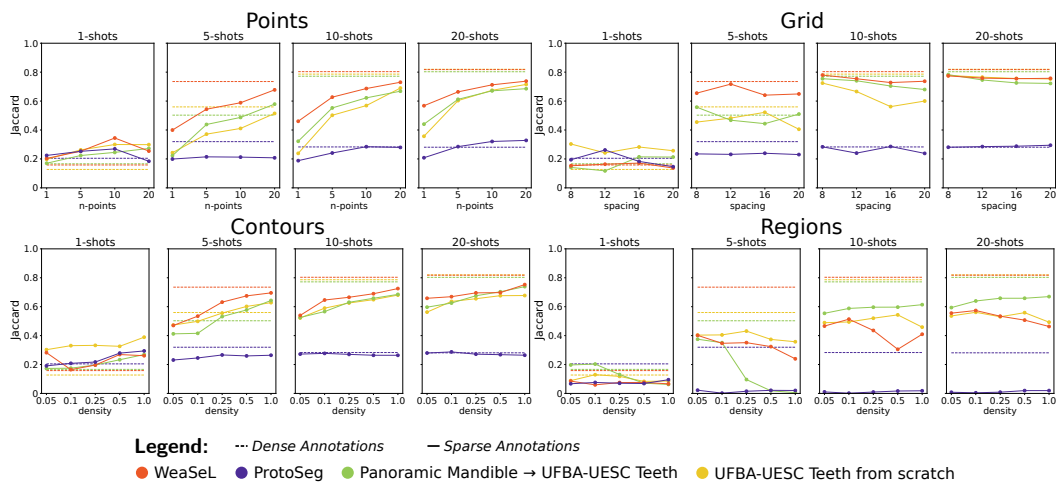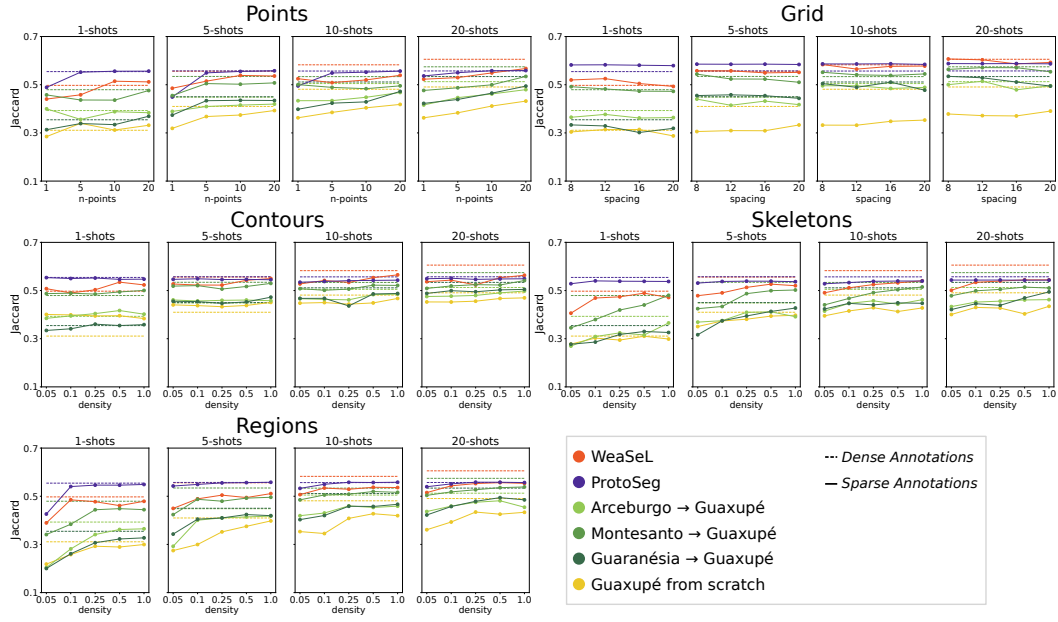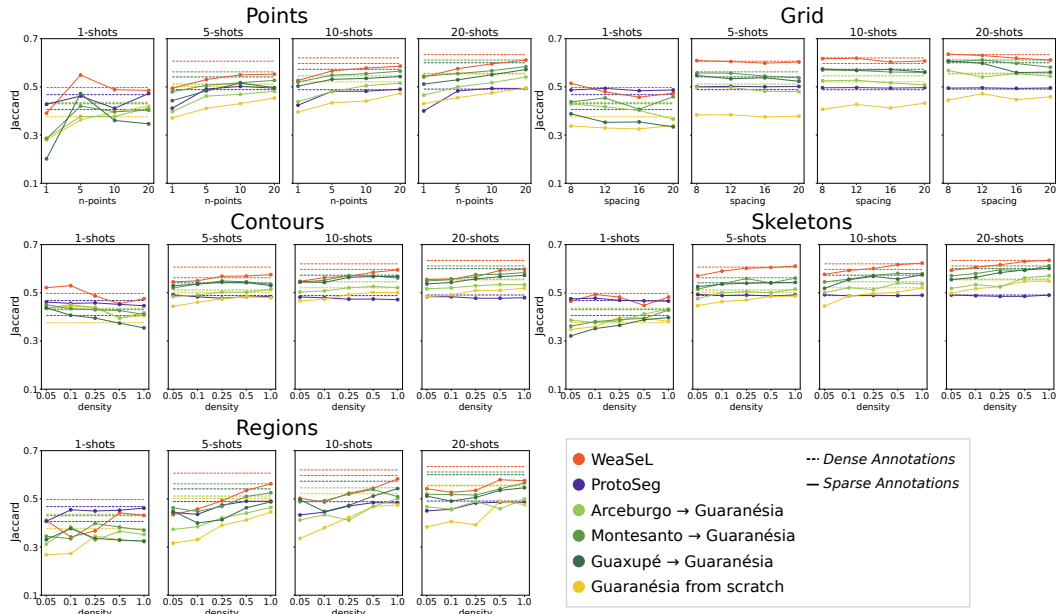Figure 23: Jaccard score of experiments with *MIAS Pectoral Muscle* task.



Figure 24: Jaccard score of experiments with *IVisionLab Teeth* task.

Figure 25: Jaccard score of experiments with *Guaxupe Coffee* task.



Figure 26: Jaccard score of experiments with *Guaranesia Coffee* task.